Packetized Serial Protocol Specification

Document Number: T10/02-013r0

December 7, 2001

Table of Contents

Packetized Serial Protocol 3
1.0 Introduction 3
2.0 Frame Formats 4
2.1 Payload 4
2.2 CRC 4
3.0 Payload 5
3.1 Payload Fields 5
3.1.1 Device 5
3.1.2 Control 5
3.1.3 Source Address 5
3.1.4 Destination Address 5
3.1.5 Frame Sequence 5
3.1.6 Frame ACK Sequence 5
3.1.7 Length 5
3.1.8 Application Data 5
3.2 Supervisory Frames 6
3.2.2 REJ (Reject) 6
3.2.3 RNR (Receive Not Ready) 6
3.3 Unnumbered Frames 7
3.3.1 SRM Command (Set Response Mode) 7
3.3.2 RST Command (Reset) 8
3.3.3 ACK Response (Acknowledgement) 8
3.3.4 Test Command 8
3.3.5 Test ACK Response 8
3.3.6 FRMR Response (Frame Reject) 9
3.4 Information Frames 10
4.0 Application Data Transfer Process 10
4.1 Premature Termination (Abort) 10
4.2 Maximum Retry / REJ 11
4.3 Sending Phase 11

4.3.1 SRM 11
4.3.2 RR (Receiver Ready) 11
4.3.3 Time-out 11
4.3.4 REJ (Reject) 12
4.3.5 RNR (Receiver Not Ready) 12
4.3.6 FRMR (Frame Reject) 12
4.4 Receiving Phase 12
4.4.1 Frame Validation Fails 13
4.4.2 Sequence Validation Failures 13
4.4.3 Receive Time-out 13
4.4.4 Information Frame 13
4.4.5 Test Frame 13
4.4.6 RST (Reset) 13
4.4.7 SRM (Set Response Mode) 13
4.4.8 FRMR (Frame Reject) 14
4.4.9 Abort Sequence 14
4.5 Frame Sequences 15
4.5.1 Link Initialization 15
4.5.2 Normal 16
4.5.3 Retry Sequence 17

Appendix A: CRC Algorithm 19
1.0 Overview 19
2.0 Polynomial Divisor 20
3.0 Dividend 20
4.0 Hardware Circuit 20
5.0 Remainder 21
6.0 Software Implementation 21
7.0 CCITT\SDLC Standard 21

Packetized Serial Protocol

This document describes the Packetized Serial Protocol.

1.0 Introduction

The Packetized Serial Protocol (PSP) is a point to point protocol. The PSP has been developed for the purpose of reliable communication between a tape drive and an autoloader or library device. Previous protocols used did not provide any error detection and were based on a master / slave protocol. The PSP protocol allows for messages to be initiated by either device. In addition, each message is encapsulated within a frame and includes a CRC16 for error detection.

The protocol involves application data frames and PSP acknowledgement frames. An acknowledgement frame is sent for every application data frame received. The application data is segmented and transmitted in a sequence of frames called Information Frames. These frames are acknowledged with Supervisory Frames. A CRC16 is placed over each frame for error detection. Timeout and retries are designed into the PSP protocol for Information Frames.

All communication is done using frames. There are three frame types, Unnumbered, Supervisory, and Information. The Unnumbered frames are used by the protocol for establishing and testing the communication link. The Supervisory frames are used for acknowledging Information frames, and Information frames carry the application data.

2.0 Frame Formats

The fundamental frame consists of four fields. Each frame starts with a Start Of Frame (SOF) and ends with a End Of Frame (EOF). The Payload is encapsulated within the SOF and EOF. A CRC16 for error detection is added.

The usage of each field is described in Table 1 on page 4.

SOF	Payload	CRC16	EOF

Table 1: Flag Descriptions

Flag	Description
SOF: Start of Frame [0xC0]	The Start of Frame flag indicates the beginning position of the Payload and initiates error checking.
EOF: End of Frame [0xC1]	The End of Frame flag delimits the end of the CRC field and terminates the frame.
CE: Control Escape Flag [0x7D]	The content of the Payload is unrestricted. Any data having the same value as one of the flags (0xC0, 0xC1, and 0x7D) must be differentiated from that flag. Each byte between the SOF and EOF flags is checked. For each data byte that is encountered having the same value as a flag the following encoding scheme is used: • Insert a 0x7D byte before the data byte. • Complement bit five of the data byte (i.e. XOR with 0x20).The CRC16 is calculated on the payload, before the encoding scheme is applied. The Control Escape Flag 0x7d byte doesn't impact the maximum user data size. The receiving device discards the encoded bytes and decodes the data bytes on the fly. Therefore, the maximum data block count is not affected.

2.1 Payload

The layout of the Payload varies depending on the type of frame. In general, it includes header information used by the protocol and the data to be transferred. The various types of frames and the corresponding layout of this field are discussed in the Payload section.

2.2 CRC

The purpose of the CRC16 field is to check the Payload for errors. The CRC16 is computed on the entire Payload, excluding any Control Escape Flag bytes. The CRC16 is a 16-bit cyclic redundancy check using the CRC-CCITT algorithm. This CRC16 is adequate for 2^16 and therefore is adequate for the maximum raw data block. Appendix A details a description of the CRC16 algorithm.

The one's complement of the CRC16 is transmitted instead of the CRC16 itself. This allows detection of slippage errors. Any device detecting a CRC16 error will throw out that frame and cause the frame to be re-sent on a time-out condition.

3.0 Payload

As mentioned previously, there are various types of frames and the Payload for each is unique. The frame type is identified by the Control field.

Reserved fields serve as placeholders to accommodate future implementation of the protocol. They are not guaranteed to be either one or zero. Since new features may be added at any time, it is suggested that no attempt be made to check reserved fields.

3.1 Payload Fields

Each frame is made up of a combination of the fields defined in the following sub-sections.

3.1.1 Device

The first field of the Payload of all types of frames is the Device. It is 1 byte in length. The upper 4 bits of this field identify the implementation version of this protocol. Implementation of this version of the protocol shall be reported as 1 in the Device field. The lower 4 bits are reserved.

3.1.2 Control

The Control field is 1 byte and defines the type of frame. This field is defined in more detail in the following sections that describe the various frame types.

3.1.3 Source Address

This field is 1 byte and is used to identify the sender of the message. This field contains the unique source address of the application that initiated the data transfer. Source addresses must be unique and within the range of 0x1-0xfe.

3.1.4 Destination Address

This field is 1 byte and is used for the routing of Information frames for processing. This field contains the destination address of the application that the data portion of the Payload is intended for.

3.1.5 Frame Sequence

This is a 1-byte field that contains a number to indicate the sequence of frames. The frame numbers start at 1 and are incremented by 1, for each Information frame transmitted. At 255 it will wrap to 0 and continue incrementing.

3.1.6 Frame ACK Sequence

This is a 1-byte field that acknowledges a received Information frame. This field contains the next Frame Sequence expected.

3.1.7 Length

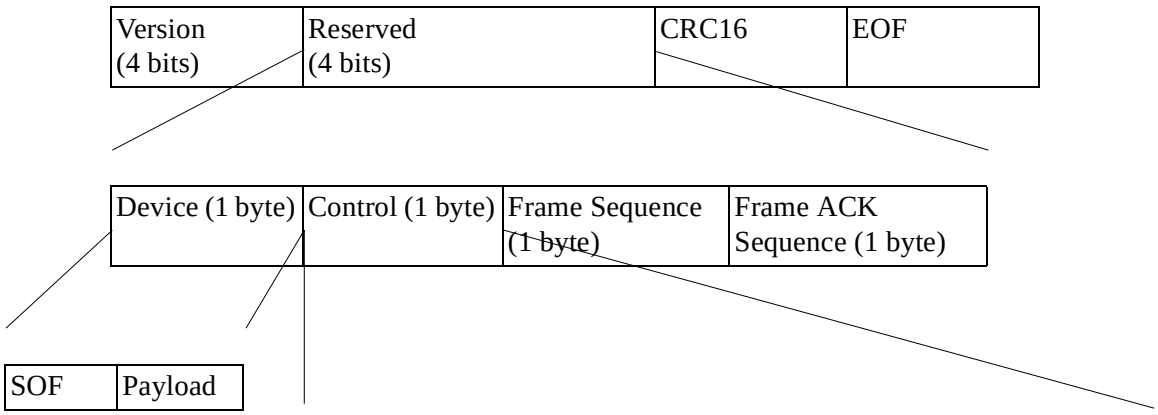
This field is 2 bytes and contains the number of bytes in the Application Data field.

3.1.8 Application Data

This field contains the raw data. The size of this field varies and is reported in the Length field.

3.2 Supervisory Frames

Supervisory frames convey an acknowledgment of an Information frame or error condition. The Supervisory Receiver Ready (RR) indicates that the frame was received and successfully validated. The Supervisory Receiver Not Ready (RNR) indicates that the frame was received and successfully validated, but the receiver is busy. The Supervisory Reject (REJ) indicates a bad sequence number in the Information Frame. The frame was not delivered and must be re-received. A RR Supervisory Frame confirms the sequence and data integrity of the last sent



7 - 4	3	2	1	0	Bit/Type
0	0	0	0	1	RR command / response [0x01]
0	1	0	0	1	REJ command / response [0x09]
0	0	1	0	1	RNR command / response [0x05]

Figure 3.1 Supervisory Frames

Information Frame.The Frame ACK Sequence field shall be one greater than the Frame Sequence field in the received Information frame. In other words, it is the next Frame Sequence it expects to receive. A RR frame also indicates that it is ready to receive additional frames.

3.2.1 Receiver Ready

A RR Supervisory Frame confirms the sequence and data integrity of the last sent Information frame. The Frame ACK Sequence field shall be one greater than the Frame Sequence field in the received Information frames. In other words, it is the next Frame Sequence it expects to receive. A RR frame indicates that it is ready to receive additional frames.

3.2.2 REJ (Reject)

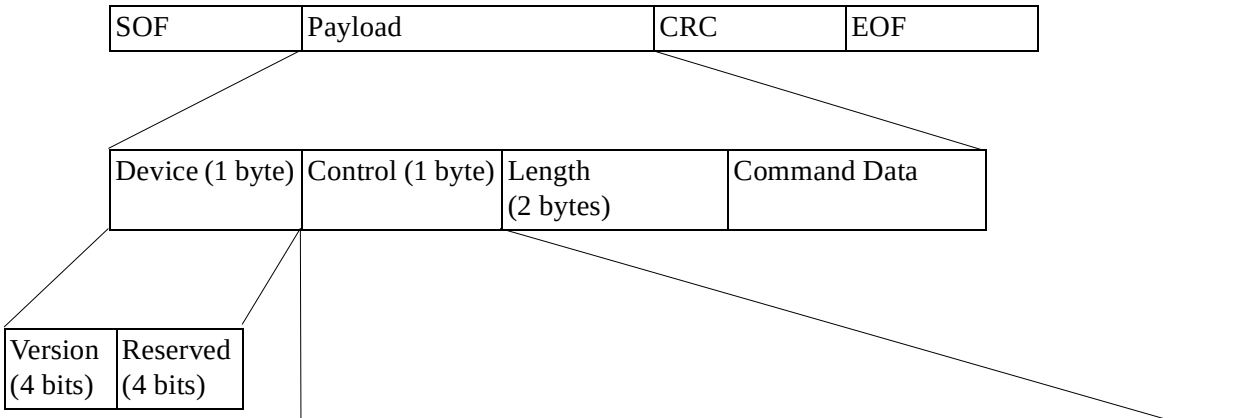
A REJ Supervisory Frame requests retransmission of the frame identified in the Frame ACK Sequence field. This is only used when the Frame Sequence number is unexpected. The frame is not delivered and must be re-sent with the proper sequence number. Frames that contain bad CRC16 errors are ignored, causing the sender to time out and re-send.

3.2.3 RNR (Receiver Not Ready)

A RNR Supervisory Frame indicates a temporary busy condition caused by internal constraints, which prevents the reception of additional Information frames. When this condition has cleared, a RR frame is sent indicating that the sending of Information frames can continue. The RNR frame also confirms the sequence and data integrity of the last sent Information Frame (Frame ACK Sequence - 1). As long as the busy condition exists, the receiver must continue to send RNR frames to prevent the sender from reaching a timeout.

3.3 Unnumbered Frames

Unnumbered Frames are used by the PSP to establish and verify the physical communication link.



7	6	5	4	3	2	1	0	Bit/Type
1	0	0	0	0	0	1	1	SRM command [0x83]
0	1	0	0	0	0	1	1	RST command [0x43]
1	1	0	0	0	0	1	1	ACK response [0xC3]
1	1	1	0	0	0	1	1	Test command [0xE3]
1	1	1	0	0	0	1	0	Test ACK response [0xE2]
1	0	0	0	0	1	1	1	FRMR response [0x87]

Figure 3.2 Unnumbered Frames

3.3.1 SRM Command (Set Response Mode)

An SRM frame is used to establish or re-establish a communication link, which synchronizes the two devices. This includes baud rate and sequence numbers. The following diagram illustrates the Data field of an SRM frame.

Command (1 byte)	Length (1-byte)	Command Data

3.3.1.1 SRM Baud Rate Data

The SRM baud rate command is 0x01 and contains a length of 1. The data byte contains the systems supported baud rates. For each baud rate that is supported, the bit shall be set.

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
153600	115200	76800	57600	38400	19200	9600	2400

For example, if the device supported 38400, 19200, and 9600, then the value would be 0x0E.

3.3.2 RST Command (Reset)

A RST frame is sent to indicate that the user has requested the communication link be reset. RST frames are not acknowledged and the communication link needs to be re-established. Both devices would initiate the initialization process by sending SRM frames.

3.3.3 ACK Response (Acknowledgement)

The ACK response frame is used to acknowledge SRM frames. The Data Field of the frame has the same format as in the SRM frame. The baud rate command has a length of 1 and the Data contains the negotiated baud rate. This is the highest baud rate that both devices published in their respective SRM frames. This can be found by doing a bit-wise or operation of the baud rates supported by each device and then identifying the highest bit that is set.

3.3.4 Test Command

Test Frames are used to establish an error free communication link between the devices. The Data field of a Test frame contains the byte sequence N+1, N+2,., where N is the Test frame number. Five Test frames shall be sent and acknowledged before the link is declared good and the link is ready to transfer Information frames.

Command (1 byte)Length (1-byte) 50

Data 01,02…32

3.3.5 Test ACK Response

Test ACK Response frames are used to acknowledge received Test frames. The Data field of a Test ACK Frame contains the data received in the Test frame. For every Test frame received one Test ACK Frame is sent.

Command (1 byte)	Length (1-byte) 50	Data [Received Test Data]

3.3.6 FRMR Response (Frame Reject)

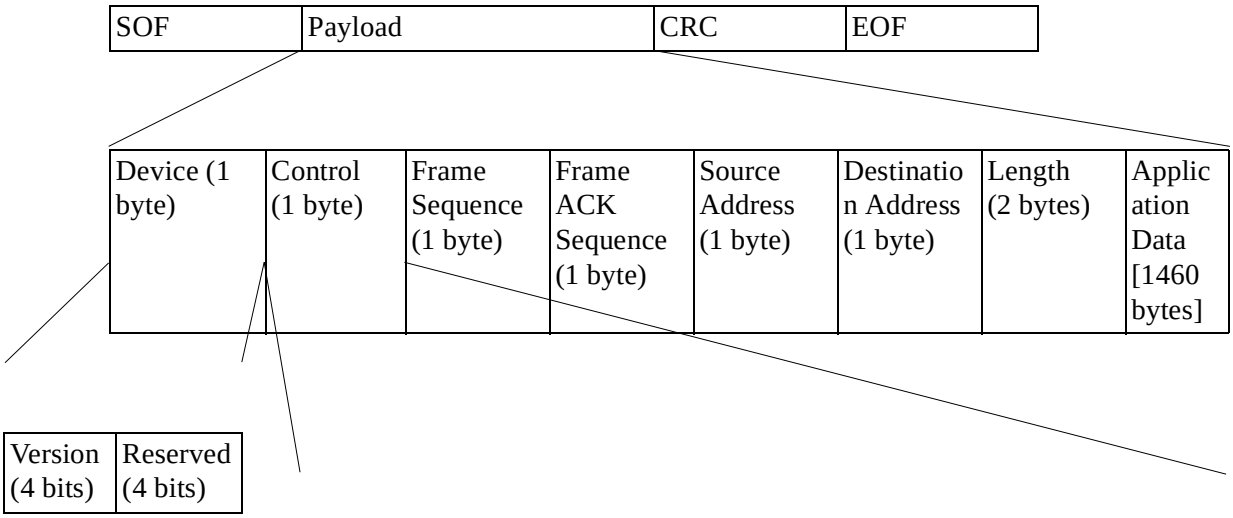
The FRMR frame shall be sent in response to the reception of a frame that cannot be delivered to the destination address. The data field of this frame type is 3 bytes in length. It includes a cause, source address and destination address.

Cause	Source Address	Destination Address

The cause value indicates the reason for the frame reject. In this version only one cause is defined, 0x03, indicating invalid destination address. The source and destination addresses are as sent in the frame being rejected.

3.4 Information Frames

Information Frames are used to transfer sequenced user data frames. All frames of this type are sequenced and acknowledged with Supervisory RR frames. If the data to be transferred is greater than the maximum user data block size (1460 bytes),it must be transferred in multiple frames. In this case, all frames except the last frame will have the Final Frame bit set to 0. Only the frame that contains the last segment of data for a transfer has the Final Frame bit set to 1.



7-5	4 (Final Frame Bit)	3-2	1	0	Bit/Type
Rsvd	1	Rsvd	1	1	Frame [0x03]
Rsvd	1	Rsvd	1	1	Final Frame [0x13]

Figure 3.3 Information Frames

4.0 Application Data Transfer Process

The data transfer process consists of sending phases and acknowledgment phases. Each sending phase is followed by an acknowledgement phase. A transmitter is associated with the sending phase and a receiver is associate with an acknowledgment phase. During the sending phase, the transmitter sends an Information frame to the receiver. The receiver validates the frame and begins an acknowledgment phase. If no errors are detected, the receiver returns a Supervisory Receiver Ready (RR) frame requesting the next frame.

4.1 Premature Termination (Abort)

The process of terminating a frame before the transmission of the frame is complete is reserved to the transmitter. The transmitter aborts transmitting a frame by sending an escape byte (0x7D) immediately followed by a flag byte (0xC0, 0xC1, or 0x7D). Both devices behave as if the frame had never been sent.

4.2 Maximum Retry / REJ

Termination also occurs if the maximum send retry count is exceeded or if the maximum number of consecutive REJ frames is reached. The retry count is three.

4.3 Sending Phase

This section describes the various frames that can be received by a device that has just sent a new Information Frame. The appropriate action that shall be taken is shown in association to the device that sent the Information frame. The table provides a summary and the details are in the following paragraphs.

Table 1: Sending Processing Phase After Information Frame Sent

Received Frame	Action by sender
SRM	Link down, re-establish link
RR	Frame sent successfully
Time-out (No Response)	Re-send frame
REJ (Reject)	Re-send requested frame
RNR (Receiver Not Ready)	Reset time out timer
FRMR (Frame Reject)	Delete frame, notify sending application

4.3.1 SRM

The receipt of an SRM frame indicates that the receiver has reset and the link is being re-established. The transmitter must reset and go through the initialization process. Any outstanding frames that were waiting to be sent are thrown away.

4.3.2 RR (Receiver Ready)

Receiving this frame indicates that the receiver has acknowledged and delivered to the registered application the last sent Information Frame.

4.3.3 Time-out

An acknowledgement frame timer shall be started once a Informational Frame is sent. The timer shall be turned off once a RR frame with the correct Frame ACK Sequence number is received. The time-out for the Information Frame shall be calculated as follows:

Maximum frame size * baud rate * 8

For a baud rate of 9600 and a maximum frame size of 1470 bytes, the time-out is 11,760 milliseconds. If the timer expires, the transmitter shall re-send the last Information Frame and the retry count shall be incremented. If the maximum number of retries is reached, a SRM shall be sent and the communication link shall be re-established.

4.3.4 REJ (Reject)

Receiving this frame indicates that an unexpected Frame Sequence number was sent. The number of the frame expected is included in the Frame ACK Sequence field of the REJ frame. The transmitter shall re-send the requested frame, if possible and if retries have not been exhausted. Otherwise, a SRM frame shall be sent and the communication link shall be re-established.

4.3.5 RNR (Receiver Not Ready)

Receiving this frame indicates that the receiver is not yet ready to receive another frame. The transmitter starts the acknowledgement frame timer and waits for the next response frame. The receiver may send another RNR before the timer expires to restart the timer. This extends the time necessary for the receiver to become ready.

4.3.6 FRMR (Frame Reject)

This frame indicates that the last frame the transmitter has sent has an unknown destination address. The transmitter shall send an indication to the application that sent this frame that it was undeliverable.

4.4 Receiving Phase

This section describes the various frames that can be received and the appropriate action that shall be taken by the receiving device. The table provides a summary and the details are in the following sections.

Table 2: Receiving Processing Phase

Received Frame	Action by Receiver
Frame validation fails	Delete frame
Sequence validation fails	Send Supervisory REJ frame
Receive Time-out	Ignore received characters
Information Frame	Send Supervisory RR or RNR
Test Frame	Send Test ACK Response
RST Frame	Reset link
SRM Frame	Re-establish link
FRMR Frame	Report to application and continue
Abort Sequence Frame	Abort current receive frame

4.4.1 Frame Validation Fails

The frame may fail the validation process as a result of any of the following:

- Unknown / unimplemented Control field.
- The data field length exceeds the maximum raw data block size.
- Serial line errors.
- CRC16 errors.

If the frame fails validation due to any of the items above, the frame is deleted (ignored). The transmitter will time-out waiting for an acknowledgment, and re-transmit the frame.

4.4.2 Sequence Validation Failures

If the Frame Sequence field does not contain the expected value, the validation fails and a REJ frame is sent.

4.4.3 Receive Time-out

A receive frame timer shall be started once the SOF flag is received. The timer shall be turned off once the EOF flag is received. The time-out for the Information Frame shall be calculated as follows:

Maximum frame size * baud rate * 4

For a baud rate of 9600 and a maximum frame size of 1470 bytes, the time-out is 5,880 milliseconds. If the timer expires, the receiver shall ignore that the frame was ever started, reset, and start looking for a SOF flag.

4.4.4 Information Frame

When an Information Frame is received that passes validation and is deliverable, the receiver shall respond with a RR or a RNR frame. Both of these frames contain a Frame ACK Sequence number to acknowledge the receipt of a valid Information Frame. The RR indicates that the receiver is ready to receive another frame whereas the RNR indicates that the receiver is busy and is not ready to receive additional frames.

4.4.5 Test Frame

This frame is transmitted during link initialization. If the frame is valid, the receiver sends a Test ACK frame in response, which contains a copy of the data bytes from the received frame.

4.4.6 RST (Reset)

This frame indicates that an application has requested the link be reset. The receiver shall perform the normal initialization process.

4.4.7 SRM (Set Response Mode)

This frame indicates the initialization of the communication link. The receiver shall respond with an ACK frame with data indicating the negotiated baud rate.

4.4.8 FRMR (Frame Reject)

Receipt of this frame indicates that the frame with the specified source and destination address was not deliverable. The receiver shall input the status to the source application.

4.4.9 Abort Sequence

The abort sequence indicates a premature termination of a data transfer on the part of the transmitter. The receiver shall terminate the current reception and behave as if the frame had never been sent.

4.5 Frame Sequences

This section describes various PSP frame flow sequences. This includes the type of PSP frames that are used throughout initialization, normal, and error conditions.

4.5.1 Link Initialization

Figure 1.1 At power up and after unrecoverable errors, each device shall go through a synchronization phase. The initialization starts with SRM and ACK Response frames. Once both sides have acknowledged SRM frames, each device starts a sequence of sending and acknowledging five Test Frames. These Test Frames will be acknowledged with Test ACK Frames, with the received test data sent back. Once this sequence of frames is transmitted successfully, the communication link is established and ready for Information frames to be transmitted. See Figure 1.1, PSP Initialization Flow for the Initialization flow.

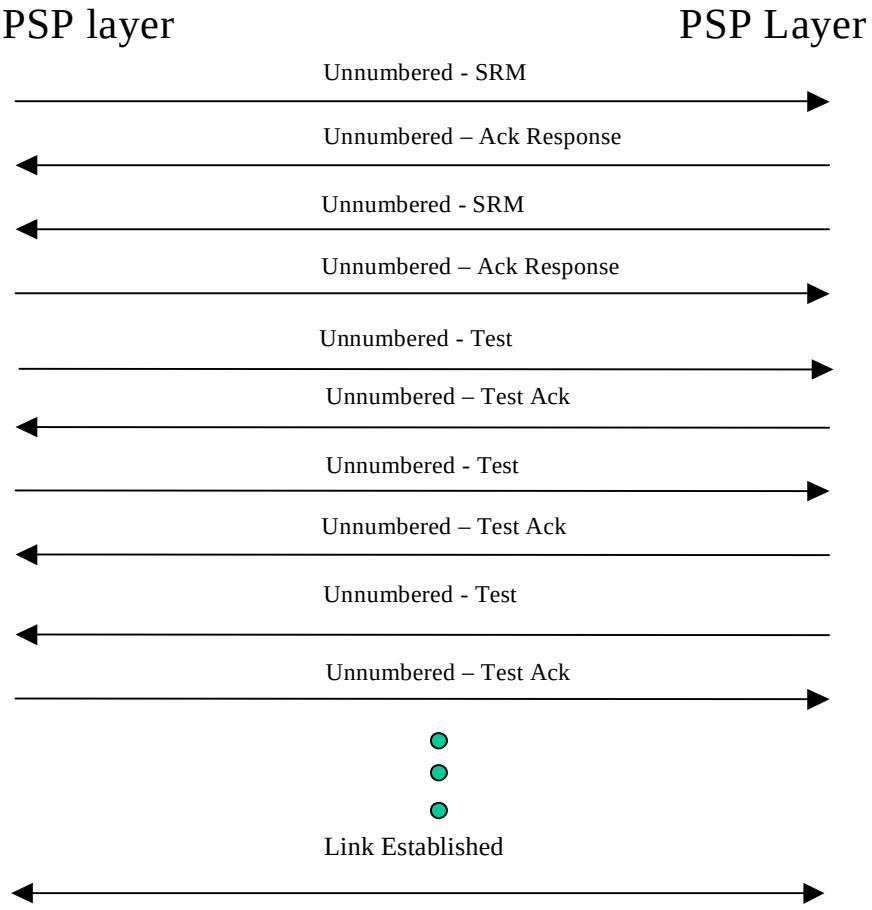


Figure 4.1 PSP Initialization Flow

4.5.2 Normal

Figure 4.2 shows a normal sequence flow of Information frames being transmitted. For each Information frame transmitted the receiving device responds with a RR Frame to indicate the successful receipt and validation of the Information frame. The main purpose of this diagram is to illustrate the interaction of the Frame Sequence and Frame ACK Sequence numbers.

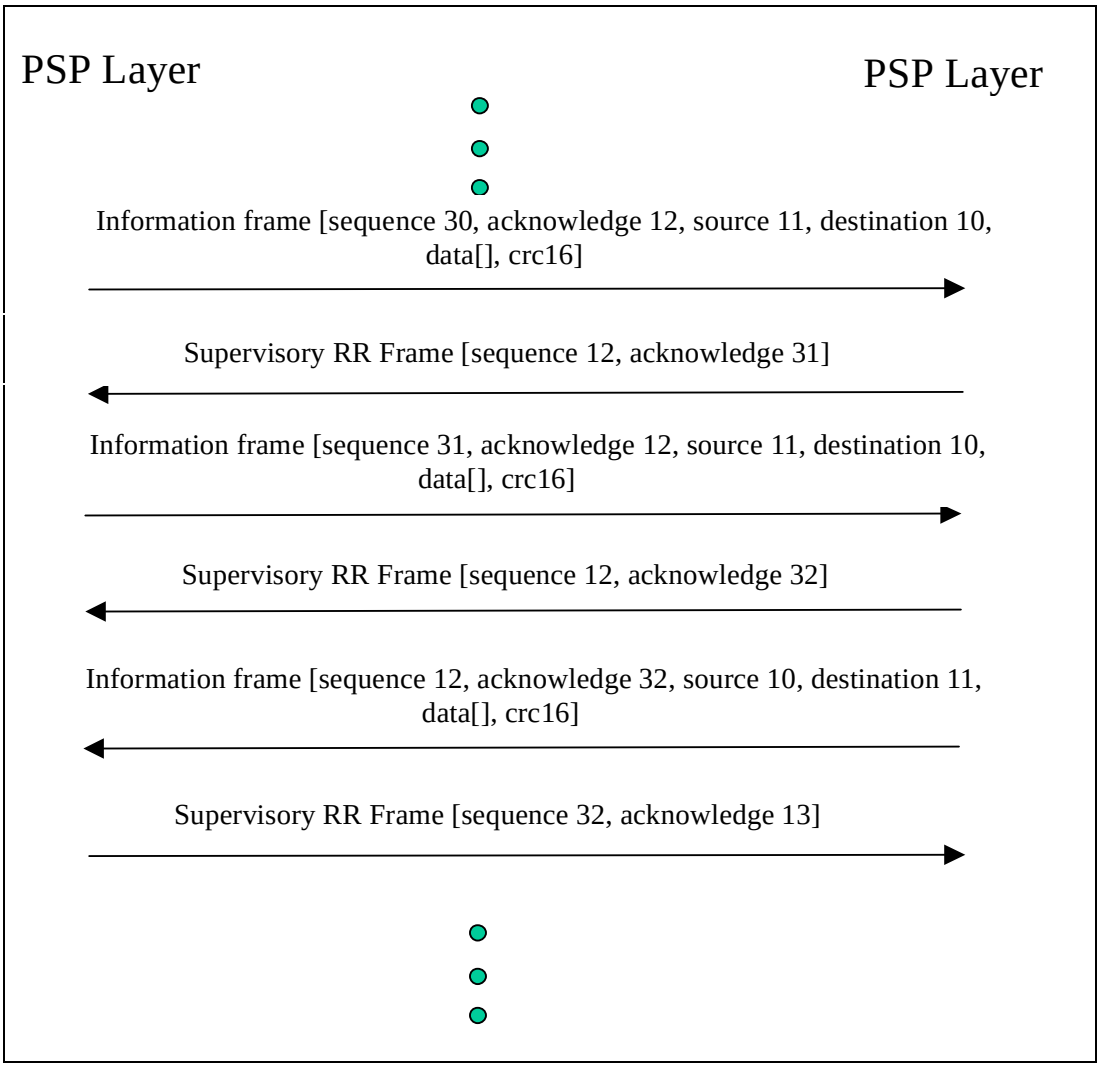


Figure 4.2 PSP Information Frame Flow

4.5.3 Retry Sequence

Figure 4.3 shows a lost Information Frame being re-transmitted. One Information frame is sent, but no Supervisory acknowledge frame for that frame is sent. The Information Frame times out and re-transmits the frame. Once the Information Frame is received, the receiver acknowledges the frame with a Receiver Ready (RR) Frame.

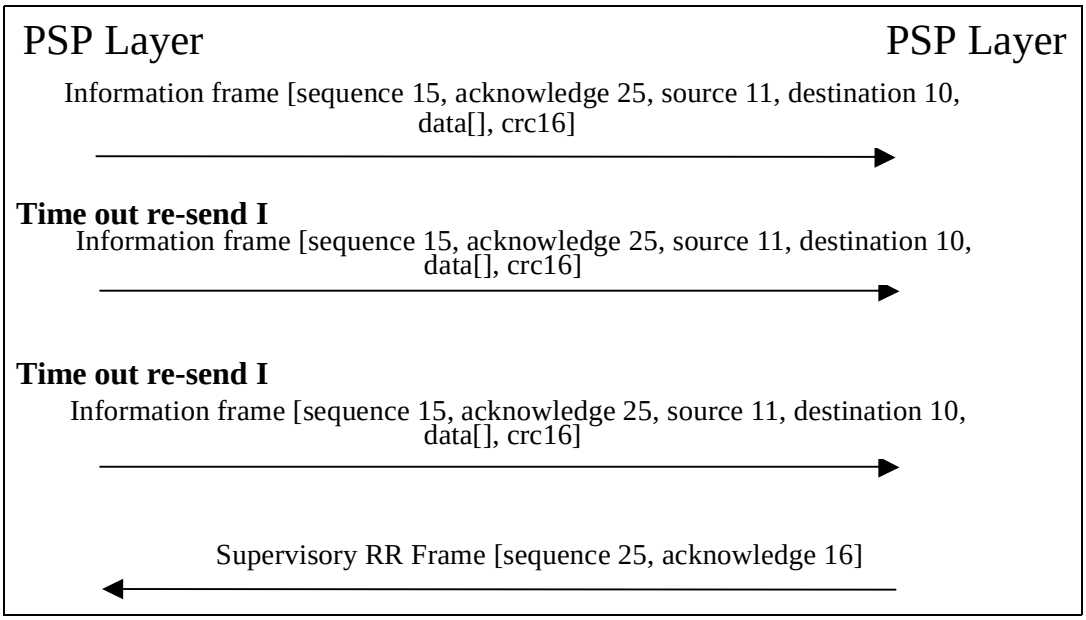


Figure 4.3 PSP Information Frame Retry Flow

A
CRC Algorithm

This chapter describes the CRC Algorithm.

1.0 Overview

The CRC algorithm employed during serial communication between the two devices is used only as an error detection mechanism. Error correction procedures are not included.

CRC values are calculated using polynomial division. A polynomial is chosen to be the divisor. The data upon which the CRC is calculated serves as the dividend. The CRC is the remainder that is left once the division process is complete. The size of the CRC is dependent upon the polynomial divisor chosen. This algorithm implements a two byte (16 bit) CRC.

One change is made to the polynomial division process. The subtraction operation is replaced with an exclusive ORing operation. This eliminates having to deal with borrows and carries.

This algorithm uses software to implement a circuit, which represents the polynomial division process.

The algorithm is based on polynomial division using the CCITT\SDLC CRC standard. The following sections describe the various parts of the algorithm:

- “Polynomial Divisor” on page A-20
- “Dividend” on page A-20
- “Hardware Circuit” on page A-20
- “Remainder” on page A-21
- “Software Implementation” on page A-21
- “CCITT\SDLC Standard” on page A-22

2.0 Polynomial Divisor

The CCITT\SDLC standard uses the following polynomial as the divisor:

```
X16 + X12 + X5 + 1
```

This polynomial is reducible by a factor of X + 1. Being reducible by a factor of X + 1 eliminates all odd bit errors.

The polynomial is divided into each data byte. Since the data is binary, the polynomial has to be converted into its binary form. This is accomplished by placing a 1 bit for each non-zero term in the polynomial and a zero bit for each zero term in the polynomial starting with the most significant term on the left. Doing this results in the following binary number:

```
1000 1000 0001 0000 1B.
```

Two more changes have to be made to the divisor to account for the way the hardware circuit functions. First, hardware shift registers shift right. The dividend is shifted to the left in polynomial division. This is accounted for by turning the divisor around. After turning the divisor around, the following binary number results:

```
1000 0100 0000 1000 1B.
```

Second, in the polynomial division, the subtraction occurs first and then the shift occurs for the next round. In the shift register, we look at the least significant bit shift then perform the exclusive OR. The divisor is shifted one bit relative to its equivalent polynomial. Shifting the divisor one bit to the right results in the following binary number:

```
1000 0100 0000 1000 B.
```

This number (8408H) serves as the feedback factor that would drive the hardware CRC generator circuit.

3.0 Dividend

The data upon which the CRC is calculated serves as the dividend.

4.0 Hardware Circuit

By placing an exclusive OR gate at the entrance to every bit that has a 1 in its feedback factor, and by placing an exclusive OR gate at the output, the hardware circuit results. If no exclusive OR gate is placed at the output, the circuit can not deal with 0. The idea, based on CRC theory, is to get the effect of the data bits scattered throughout the remainder. Figure A.1 illustrates what all the exclusive OR gates do.

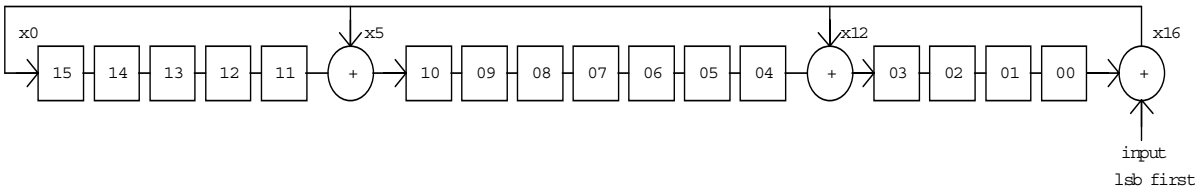


Figure A.1: Exclusive OR Gates

5.0 Remainder

The remainder is the CRC. As each bit is processed through the circuit the CRC is updated. After all the data has been processed the resulting CRC is transmitted.

6.0 Software Implementation

This algorithm employs a byte-wise CRC computation. To do this it is necessary to follow the processing of one byte through the circuit. The data is fed into every tap. Eight shifts have to occur to process one byte. The process is detailed in Table 1. At any given stage, all the terms in a column are to be exclusively ORed together. After eight shifts the following is observed:

- The high byte of the CRC shifted down into the low byte position.
- The rest of the result is dependent on the value of the data being processed. Using a few exclusive ORs in the software can arrive at this result.

The eighth shift state can be represented by the following code:

```
Data Byte = Data Byte ^ Low Byte of CRC
Data Byte = Data Byte ^ (Data Byte << 4)
CRC = (CRC >> 8) ^ (Data Byte << 8) ^ (Data Byte << 3) ^ (Data Byte >> 4)
```

Since communication is through a serial interface, only one byte can be transmitted at a time. The above code needs to be revised for a two-byte CRC. The resulting code follows:

```
d=d ^ crc1
d=d ^ (d << 4)
crc1=crc2 ^ (d << 3) ^ (d >> 4)
crc2=d ^ (d >> 5);
```

7.0 CCITT\SDLC Standard

In the SDLC standard the CRC is initialized to 0FFFFH instead of to 00000H. This is done to catch any unwanted nulls preceding a message frame. Each CRC byte (i.e. crc1, crc2) is initialized to 0FFH before the CRC is calculated.

The SDLC also sends the complement of the CRC instead of the actual CRC. The complement of each CRC byte is sent with each message frame being transmitted. During the reception process the CRC bytes are also processed through the CRC code. If there are no errors the following constant results:0F0B8H.

Table A.1: Processing One Byte Through the Shift Register

SHIFT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	C0 DO	C15	C14	C13	C12	C11 C0 D0	C10	C9	C8	C7	C6	C5	C4 C0 D0	C3	C2	C1
2	C1 D1	C0 D0	C15	C14	C13	C12 C1 D1	C11 C0 D0	C10	C9	C8	C7	C6	C5 C1 D1	C4 C0 D0	C3	C2
3	C2 D2	C1 D1	C0 D0	C15	C14	C13 C2 D2	C12 C1 D1	C11 C0 D0	C10	C9	C8	C7	C6 C2 D2	C5 C1 D1	C4 C0 D0	C3
4	C3 D3	C2 D2	C1 D1	C0 D0	C15	C14 C3 D3	C13 C2 D2	C12 C1 D1	C11 C0 D0	C10	C9	C8	C7 C3 D3	C6 C2 D2	C5 C1 D1	C4 C0 D0

Table A.1: Processing One Byte Through the Shift Register (Continued)

SHIFT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
5	C4 D4 C0 D0	C3 D3	C2 D2	C1 D1	C0 D0	C15 C4 D4 C0 D0	C14 C3 D3	C13 C2 D2	C12 C1 D1	C11 C0 D0	C10	C9	C8 C4 D4 C0 D0	C7 C3 D3	C6 C2 D2	C5 C1 D1
6	C5 D5 C1 D1	C4 D4 C0 D0	C3 D3	C2 D2	C1 D1	C0 D0 C5 D5 C1 D1	C15 C4 D4 C0 D0	C14 C3 D3	C13 C2 D2	C12 C1 D1	C11 C0 D0	C10	C9 C5 D5 C1 D1	C8 C4 D4 C0 D0	C7 C3 D3	C6 C2 D2
7	C6 D6 C2 D2	C5 D5 C1 D1	C4 D4 C0 D0	C3 D3	C2 D2	C1 D1 C6 D6 C2 D2	C0 D0 C5 D5 C1 D1	C15 C4 D4 C0 D0	C14 C3 D3	C13 C2 D2	C12 C1 D1	C11 C0 D0	C10 C6 D6 C2 D2	C9 C5 D5 C1 D1	C8 C4 D4 C0 D0	C7 C3 D3

Table A.1: Processing One Byte Through the Shift Register (Continued)

SHIFT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
8	C7D7 C3D3	C6D6 C2D2	C5D5 C1D1	C4D4 C0D0	C3D3	C2D2 C7D7 D6D6 C3D3	C1D1 C6C6 D5D5 C2D2	C0D0 C5D5 D4D4 C1D1	C15 C4D4 C0D0	C14 C3D3	C13 C2D2	C12 C1D1	C11 C7D7 C0D0 C3D3	C10 C6D6 C2D2	C9C5D5 C1D1	C8C4D4 C0D0 D1