To: T10 Technical Committee
From: Rob Elliott, Compaq Computer Corporation (Robert.Elliott@compaq.com)
Date: 6 July 2001
Subject: SPI-4 negotiation message rewrite

Revision History

Revision 0 (16 April 2001): first revision released to T10.

Revision 1 (30 June 2001): incorporated feedback from Brian Cockburn (Adva Optical) and the May Parallel SCSI WG. Added figures for initiator-originated and target-originated WDTR and SDTR.

Revision 2 (6 July 2001): incorporated feedback from George Penokie (Tivoli/IBM). Change bars from revision 1.

Related Documents

spi4r05 – SCSI Parallel Interface – 4 revision 5

Overview

The PPR, WDTR, and SDTR message descriptions are full of duplications and contain some errors. This proposal rewrites the three sections, moving most of the text into a model section in clause 4 and simplifying the individual sections in clause 16. It attempts to eliminate duplications and fix errors. It only attempts to change behavior that needs to be changed.

Most of the errors are in the descriptions of error handling – parity errors and unexpected bus frees. The standard is not clear on when the initiator and target should maintain their negotiated settings and when they should reset them to asynchronous. This does not cause many problems in practice because errors are rare, and today's software probably issues a bus resets when errors occur, which brings all the devices back to a known state. By providing better guidance, resorting to bus resets should be less necessary.

Suggested Changes

Editor's directions: Add the new model section to chapter 4. Remove sections 16.3.10 PPR, 16.3.14 SDTR, and 16.3.16 WDTR. Add a new Negotiation section 16.x (parallel to the Link section) with subsections for PPR, SDTR, and WDTR.

The text was originally pulled from SPI-4 revision 0; some changes made through revision 5 may have been missed.

Table of contents:

4.1 Negotiation

4.1.1 Negotiation introduction

PARALLEL PROTOCOL REQUEST (PPR), SYNCHRONOUS DATA TRANSFER REQUEST (SDTR), and WIDE DATA TRANSFER REQUEST (WDTR) messages are used to alter the transfer agreement between two ports (see 3.1.76). The transfer agreement defines the protocol used during DATA phases (e.g., transfer period, REQ/ACK offset, transfer width) and agreement on features not affecting DATA phases (e.g., QAS). All other information transfer phases (COMMAND, MESSAGE, and STATUS) use eight-bit asynchronous data transfers.

The default transfer agreement is narrow and asynchronous (see 4.1.4.2).

PPR, SDTR, and WDTR are called negotiation messages. When an initiator sends one of them, the message names are PPR OUT, SDTR OUT, and WDTR OUT. When a target sends one of them, the message names are PPR IN, SDTR IN, and WDTR IN. A negotiation sequence consists of at least one matching set of negotiation messages (~e.g., PPR OUT and PPR IN).

A transfer agreement is maintained by each port for each ~~pair of ports~~other port on the SCSI bus. Each port (see 3.1.76) may be used as either a target port (see 3.1.xx) or an initiator port (see 3.1.xx). The same transfer agreement applies whether the port is being used as a target port or as an initiator port ~~All communication that occurs though a port shall use the transfer agreement regardless of whether the port is acting as a target port or an initiator port (e.g., if port 7 is used as an initiator port and port 0 is used as a target port, the same transfer agreement also applies when port 7 is used as a target port and port 0 is used as an initiator port)~~.

~~Negotiations are maintained on a port basis. All logical units in a target share the same transfer agreement.~~

4.1.2 Negotiation algorithm

An initiator port and target port exchange negotiation messages to perform negotiation. The originating port is the one that sends the first negotiation message and the responding port is the one that replies.

Ports shall not set message fields to values they do not support. The originating port should set the fields in the originating negotiation message to the maximum values (e.g., fastest transfer period, largest REQ/ACK offset, etc.) it supports. If the responding port is able to support the requested values, it shall return the same values in the responding negotiation message. If the responding port requires different values (i.e., a subset of the originating port's request), it shall return those values in the responding negotiation message (e.g., if the originating port asks for a REQ/ACK offset of 32 and the responding port only supports a REQ/ACK offset of 16, the originating message requests an offset of 32 and the responding message replies with an offset of 16).

If the responding negotiation message contains values the originating port does not support, the originating port shall respond with a MESSAGE REJECT message.

The valid error-free negotiation message sequences are shown in Figure 1. A description of ~~all~~ the possible message sequences is in section 4.1.7.
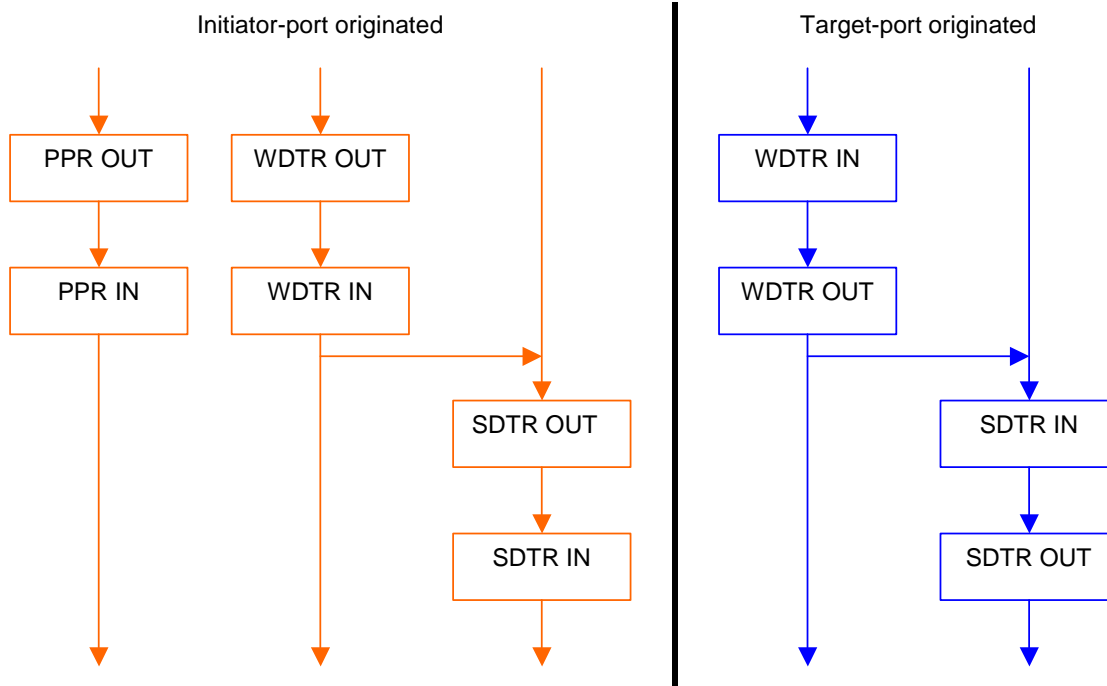
Figure 1. Error-free negotiation message sequences.

4.1.3 When to negotiate

A target port shall consider its transfer agreement invalid after:
a) a reset event (see 12.5); or
b) an error occurs while transmitting a responding negotiation message.

An initiator port shall consider its transfer agreement invalid after:
a) a reset event (see 12.5));
b) a unit attention status is received with an additional sense code whose ADDITIONAL SENSE CODE field contains 29h (e.g., POWER ON, RESET, OR BUS DEVICE RESET OCCURRED; POWER ON OCCURRED; SCSI BUS RESET OCCURRED; BUS DEVICE RESET FUNCTION OCCURRED; DEVICE INTERNAL RESET; TRANSCEIVER MODE CHANGED TO SINGLE-ENDED; or TRANSCEIVER MODE CHANGED TO LVD);
> NOTE nn: These additional sense codes are never reported in a status information unit because SCSI devices default to information units disabled.

c) an unexpected COMMAND phase occurs when selecting without using attention condition (i.e., when selecting a target with information units enabled),)the initiator port detects an unexpected COMMAND phase (see 10.6.3.1). This may occur if the target has been hot-swapped; or
d) an error occurs while transmitting a responding negotiation message.

A logical unit reset (see 16.5.6) has no effect on a transfer agreement.

An initiator port shall originate negotiation before sending a command whenever it has an invalid transfer agreement. A target port shall originate negotiation before accepting a command whenever it has an invalid transfer agreement.

A port may originate negotiation even if it has a valid transfer agreement (e.g., to change the settings or as part of integrity checking procedures). Negotiation should not be originated after every selection as this may impact performance. Because ports remember their transfer agreements between connections, negotiation is unnecessary and performance impact from extra negotiations is likely.

NOTE nn: Target ports may have had their support for originating negotiation after power on disabled to support broken initiator software. If an initiator port sends a command to a target that has been powered on (e.g., after a hot plug) that results in a unit attention condition, the initiator port realizes the transfer agreement is invalid and originates negotiation before the next command. However, if the command is INQUIRY, REPORT LUNS, or REQUEST SENSE, a unit attention condition is not created. An invalid data phase may occur if the target port does not originate negotiation. If the initiator port always originates negotiation before sending those commands, the data phase runs correctly. When information units are disabled, an initiator port should may originate negotiation with its currently negotiated settings before each INQUIRY, REPORT LUNS, or REQUEST SENSE command to avoid this problem. When information units are enabled, the selection without attention results in an unexpected COMMAND phase that notifies the initiator port that its transfer agreement is invalid, so extra negotiation is not needed.

4.1.4 Negotiable fields

4.1.4.1 Negotiable fields introduction

Table 1 lists the fields that may be negotiated and the effects of successful negotiation on those fields by each of the different negotiation messages. Ports shall implement a given message if they implement fields that are only negotiable with that message.

Table 1. Negotiable fields and effects of successful negotiation

Field Name		Negotiation message pair		
		PPR	WDTR	SDTR
TRANSFER PERIOD FACTOR		negotiated (valid values: 08h-FFh)	No requirement	negotiated (valid values: 0Ah-FFh)
REQ/ACK OFFSET		negotiated	Resets to 00h	negotiated
TRANSFER WIDTH EXPONENT		negotiated	negotiated	unchanged
PROTOCOL OPTIONS	PCOMP_EN	negotiated	Resets to 0	Resets to 0
	RTI	negotiated	Resets to 0	Resets to 0
	RD_STRM	negotiated	Resets to 0	Resets to 0
	WR_FLOW	negotiated	Resets to 0	Resets to 0
	HOLD_MCS	negotiated	Resets to 0	Resets to 0
	QAS_REQ	negotiated	Resets to 0	Resets to 0
	DT_REQ	negotiated	Resets to 0	Resets to 0
	IU_REQ	negotiated	Resets to 0	Resets to 0

When negotiating, the responding port shall respond with values that are a subset of the values in the originating message as indicated by the "Response shall be" column in Table 2 (e.g., if the originating message requests a REQ/ACK offset of 10h, the responding message is has a REQ/ACK offset field set to 10h or lower).

Table 2. Responding message requirements

Field Name		Message	Response shall be numerically
TRANSFER PERIOD FACTOR		PPR, SDTR	Greater than or equal
REQ/ACK OFFSET		PPR, SDTR	Less than or equal
TRANSFER WIDTH EXPONENT		PPR, WDTR	Less than or equal
PROTOCOL OPTIONS	PCOMP_EN	PPR	Any value
	RTI	PPR	Less than or equal
	RD_STRM	PPR	Less than or equal
	WR_FLOW	PPR	Less than or equal
	HOLD_MCS	PPR	Less than or equal
	QAS_REQ	PPR	Less than or equal
	DT_REQ	PPR	Less than or equal
	IU_REQ	PPR	Less than or equal

4.1.4.2 Transfer agreements

The valid transfer agreements that are in effect for various combinations of field values are described in Table 3. ~~The terms are not exclusive; more than one may be in effect at the same time.~~

Table 3. Valid transfer agreements

Transfer agreement	Description
default	REQ/ACK offset set to 00h, transfer width exponent set to 00h, all protocol options set to zero~~0~~.
asynchronous	REQ/ACK offset set to 00h, all protocol options set to zero~~0~~.
synchronous	REQ/ACK offset greater than or equal to 01h, transfer period factor greater than or equal to 09h, all protocol options set to zero~~0~~.
paced	REQ/ACK offset greater than or equal to 01h, transfer width exponent set to 01h, transfer period factor set to 08h, DT_REQ set to 1.
wide	Transfer width exponent set to 01h.
narrow	Transfer width exponent set to 00h.
ST data	REQ/ACK offset greater than or equal to 01h, DT_REQ set to zero~~0~~.
DT data	REQ/ACK offset greater than or equal to 01h, DT_REQ set to one~~1~~.
data group	REQ/ACK offset greater than or equal to 01h, DT_REQ set to one, IU_REQ set to zero.
information unit	REQ/ACK offset greater than or equal to 01h, DT_REQ set to one, IU_REQ set to one.

[Editor's note: add "data group" and "information unit"?]
[Editor's note: sort the table?]

4.1.4.3 Transfer period factor

The TRANSFER PERIOD FACTOR field selects the transfer period (see 3.1.101) and determines which timing values in table~~s 31 and 32~~ 32, table 33, table 34, and table 35 shall be honored. The field values are defined in Table 4. The TRANSFER PERIOD FACTOR field is negotiated with the PPR and SDTR messages. ~~If REQ/ACK OFFSET is greater than or equal to 01h, a value of 08h indicates that a paced transfer agreement is in effect and that DT_REQ is set to 1, while a value of 09h indicates that a DT transfer agreement is in effect and that DT_REQ is set to 1.~~

Table 4. Transfer Period Factor

Value	Description	Message	Timing Values (see table ~~31 32,and~~ table ~~32~~33, table 34, and table 35)
00h - 07h	Reserved. Faster transfer periods may be defined by future standards.		
08h	Transfer period equals 6,25 ns. Only valid for paced transfers.	PPR	Fast-160
09h	Transfer period equals 12,5 ns. Only valid for DT transfers.	PPR	Fast-80
0Ah	Transfer period equals 25 ns	PPR, SDTR	Fast-40
0Bh	Transfer period equals 30,3 ns	PPR, SDTR	Fast-40
0Ch	Transfer period equals 50 ns	PPR, SDTR	Fast-20
0Dh - 18h	Transfer period equals the TRANSFER PERIOD FACTOR x 4	PPR, SDTR	Fast-20
19h - 31h	Transfer period equals the TRANSFER PERIOD FACTOR x 4	PPR, SDTR	Fast-10
32h - FFh	Transfer period equals the TRANSFER PERIOD FACTOR x 4	PPR, SDTR	Fast-5

Table 5 shows which transfer period factors may be used with different types of ~~data~~ transfer agreements.

Table 5. Transfer Period Factor relationships

Value	Data group ~~transfers~~transfer agreement	Information unit ~~transfers~~transfer agreement	Synchronous ~~data~~ transfer agreement~~s~~	Paced ~~data transfers~~transfer agreement
00h - 07h	reserved			
08h	no	yes	no	yes
09h	yes	yes	yes	no
0Ah	yes	yes	yes	no
0Bh	yes	yes	yes	no
0Ch	yes	yes	yes	no
0Dh - 18h	yes	yes	yes	no
19h - 31h	yes	yes	yes	no
32h - FFh	yes	yes	yes	no

Table 9 defines valid combinations of transfer period factor and other fields.

4.1.4.4 REQ/ACK offset

The REQ/ACK OFFSET field determines the maximum number of REQs allowed to be outstanding before a corresponding ACK is received at the target. The REQ/ACK OFFSET field is negotiated with the PPR and SDTR messages.

For ST data transfers (i.e., DT_REQ negotiated to zero) the REQ/ACK OFFSET is the maximum number of REQ assertions allowed to be outstanding before a corresponding ACK assertion is received at the target. The REQ/ACK offset represents the number of bytes if the transfer width is one byte or twice the number of bytes if the transfer width is two bytes.

For DT synchronous data transfers (i.e., DT_REQ negotiated to one and IU_REQ negotiated to zero) the REQ/ACK OFFSET is the maximum number of REQ transitions allowed to be outstanding before a

corresponding ACK transition is received at the target. The REQ/ACK offset represents twice the number of bytes, since DT data transfers always use a transfer width of two bytes.

For paced DT DATA IN transfers the REQ/ACK OFFSET is the maximum number of data valid state REQ assertions (see 10.8.4.3) allowed to be outstanding before a corresponding ACK assertion is received at the target. The REQ/ACK OFFSET represents four times the number of bytes.

For paced DT DATA OUT transfers the REQ/ACK OFFSET is the maximum number of data valid state REQ assertions allowed to be outstanding before a corresponding data valid state ACK assertion is received at the target. The REQ/ACK OFFSET represents four times the number of bytes.

[Editor's note: does this table work better than those paragraphs?]

Transfer agreements in effect	Definition of REQ/ACK offset	REQ/ACK offset represents
a) narrow transfer agreement; b) synchronous transfer agreement; and c) ST data transfer agreement	Maximum number of REQ assertions allowed to be outstanding before a corresponding ACK assertion is received at the target.	the number of bytes
a) wide transfer agreement; b) synchronous transfer agreement; and ST data transfer agreement	Maximum number of REQ assertions allowed to be outstanding before a corresponding ACK assertion is received at the target.	2 times the number of bytes
a) synchronous transfer agreement; and b) DT data transfer agreement	Maximum number of REQ transitions allowed to be outstanding before a corresponding ACK transition is received at the target.	2 times the number of bytes
paced transfer agreement (DT DATA IN phase)	Maximum number of data valid state REQ assertions (see 10.8.4.3) allowed to be outstanding before a corresponding ACK assertion is received at the target	4 times the number of bytes
paced transfer agreement (DT DATA OUT phase)	Maximum number of data valid state REQ assertions allowed to be outstanding before a corresponding data valid state ACK assertion is received at the target	4 times the number of bytes

See 4.8 for an explanation of the differences between ST and DT data transfers.

The REQ/ACK OFFSET value is chosen to prevent overflow conditions in the port's receive buffer and offset counter. The values are defined in Table 6. Table 6 also indicates which timing values in table 31 32, table 33, table 34, and table 35 shall be honored. A REQ/ACK OFFSET value of zero indicates asynchronous data transfer mode and that the PERIOD FACTOR field and the PROTOCOL OPTIONS field shall be ignored; a value of FFh indicates unlimited REQ/ACK offset. If the REQ/ACK offset is not zero either a synchronous or paced transfer agreement is in effect.

Table 6. REQ/ACK offset

Value	Description	Timing values (See table ~~31~~ 32, ~~and~~ table 3~~3~~2, table 34, and table 35)
00h	Asynchronous transfer agreement. Transfer period factor and protocol options shall be ignored.	Asynch
01h-FEh	Synchronous or paced transfer agreement. Specified offset.	Determined by ~~T~~transfer ~~Period~~ period ~~Factor~~factor. See Table 4.
FFh	Synchronous or paced transfer agreement. Unlimited offset.	Determined by ~~Transfer~~ transfer ~~P~~period ~~Factor~~factor. See Table 4.

Table 9 defines valid combinations of REQ/ACK offset and other fields.

4.1.4.5 Transfer width exponent

The TRANSFER WIDTH EXPONENT field defines the transfer width to be used during DATA IN and DATA OUT phases during data transfers. The values are defined in Table 7. The TRANSFER WIDTH EXPONENT field is negotiated with the PPR and WDTR messages. If any of the protocol options bits are set to one, the only valid transfer width is 16 bits (01h). If all the protocol options bits are set to zero, a valid transfer width is 8 bits (00h) or 16 bits (01h). ~~A TRANSFER WIDTH EXPONENT field value of 02h is obsolete and values greater than 02h are reserved.~~

~~If the transfer width is 8 bits a narrow transfer agreement is in effect. If the transfer width is 16 bits a wide transfer agreement is in effect.~~

Table 7. Transfer Width Exponent

Value	Description
00h	8 bit data bus ~~(narrow)~~. Narrow transfer agreement.
01h	16 bit data bus ~~(wide)~~. Wide transfer agreement.
02h	Obsolete
03h-FFh	Reserved

Table 9 defines valid combinations of transfer width exponent and other fields.

4.1.4.6 Protocol options

4.1.4.6.1 Protocol options introduction

The protocol options fields affect the protocol used between the ports. They may only be negotiated through PPR messages ~~(which are originated only by initiator ports)~~, and are set to zero by WDTR and SDTR messages.

The target port uses the protocol options bits to indicate to the initiator port if it agrees to enable the requested protocol options. Except for the PCOMP_EN bit, the target shall not enable any protocol options that were not enabled in the negotiation message received from the initiator.

Table 8 describes the protocol options bits.

Table 8. Protocol options bits

Name	Description
PCOMP_EN	Precompensation enable
RTI	Retain training information
RD_STRM	Read streaming and read flow control enable
WR_FLOW	Write flow control enable
HOLD_MCS	Hold margin control settings
QAS_REQ	~~Quick arbitration and selection (~~QAS~~)~~ enable request
DT_REQ	~~Dual transition (~~DT~~)~~ clocking enable request
IU_REQ	Information units ~~(IU)~~ enable request

4.1.4.6.2 IU_REQ

The initiator port shall set IU_REQ to one in the PPR OUT message to request that information unit transfers be enabled. In response, the target port shall set its IU_REQ to one if it agrees to use information unit transfers or zero if it does not.

The initiator port shall set IU_REQ to zero in the PPR OUT message to request that information unit transfers be disabled. In response, the target port shall set IU_REQ to zero in the PPR IN message.

If IU_REQ is one, an information unit transfer agreement is in effect. If IU_REQ is zero, an asynchronous, synchronous or data group transfer agreement is in effect.

Table 9 defines valid combinations of IU_REQ and other fields.

Each time a negotiation results in the IU_REQ bit being changed from the previous agreement (i.e., zero to one or one to zero) the target shall go to a BUS FREE phase on completion of the negotiation. Additional requirements (see 14.1) shall be met if the IU_REQ bit is changed as a result of the negotiation.

4.1.4.6.3 DT_REQ

The initiator port shall set DT_REQ to one to request that DT DATA phases be enabled. In response, the target port shall set DT_REQ to one if it agrees to use DT DATA phases or zero if it does not.

The initiator port shall set DT_REQ to zero to request that information unit transfers be disabled. In response, the target port shall set DT_REQ to zero in the PPR IN message.

If DT_REQ is one, a DT data transfer agreement is in effect. If DT_REQ is zero, an ST data transfer agreement is in effect.

Table 9 defines valid combinations of DT_REQ and other fields.

4.1.4.6.4 QAS_REQ

The initiator port shall set QAS_REQ to one to request that QAS be enabled. In response, the target port shall set QAS_REQ to one if it supports QAS or zero if it does not.

The initiator port shall set QAS_REQ to zero to request that information unit transfers be disabled. In response, the target port shall set QAS_REQ to zero in the PPR IN message.

Table 9 defines valid combinations of QAS_REQ and other fields.

When QAS is enabled, the port shall participate in QAS arbitrations when attempting to connect to a port that has enabled QAS. When QAS is enabled and information unit transfers are enabled for a connected target port, that target port may issue a QAS REQUEST message to release the bus after a DT DATA

phase. When QAS is enabled and information unit transfers are disabled for a connected target port, that target port shall not issue QAS REQUEST messages.

4.1.4.6.5 HOLD_MCS

The initiator port shall set HOLD_MCS to one to indicate that the target should hold any margin control settings set with the margin control subpage of the port control mode page (see 18.1.4). In response, the target port shall set HOLD_MCS to one if it is capable of retaining the settings and zero if it is not.

The initiator port shall set HOLD_MCS to zero to indicate that the target shall reset to their default values any margin control settings set with the margin control subpage of the port control mode page (see 18.1.4). In response, the target port shall set HOLD_MCS to zero.

Table 9 defines valid combinations of HOLD_MCS and other fields.

4.1.4.6.6 WR_FLOW

The initiator port shall set WR_FLOW to one to indicate that the target should enable write flow control during write streaming (see Table 29, 4.10.3.3 and 8.2). In response, the target port shall set WR_FLOW to one if it is capable of write flow control and zero if it is not.

The initiator port shall set WR_FLOW to zero to indicate that the target shall disable write flow control during write streaming. In response, the target port shall set WR_FLOW to zero.

Write streaming and write flow control only occurs during information unit transfers.

Table 9 defines valid combinations of WR_FLOW and other fields.

4.1.4.6.7 RD_STRM

The initiator port shall set RD_STRM to one to indicate that the target should enable read streaming and read flow control (see Table 29, 4.10.3.3, 8.2, and 14.3.4). In response, the target port shall set RD_STRM to one if it is ~~agrees~~ capable of read streaming and read flow control and zero if it is not.

The initiator port shall set RD_STRM to zero to indicate that the target shall disable read streaming and read flow control. In response, the target port shall set RD_STRM to zero.

Read streaming and read flow control only occur during information unit transfers.

Table 9 defines valid combinations of RD_STRM and other fields.

4.1.4.6.8 RTI

The initiator port shall set RTI to one to indicate it is capable of ~~request~~ saving paced data transfer training information (see 10.8.4.2.1) and to indicate that the target should not retrain. In response, the target port shall set RTI to one if it is capable of saving paced data transfer training information and zero if it is not.

The initiator port shall set RTI to zero to if it does not support reword saving paced data transfer training information to indicate that the target shall retrain. In response, the target port shall set RTI to zero.

Table 9 defines valid combinations of RTI and other fields.

4.1.4.6.9 PCOMP_EN

The initiator port shall set PCOMP_EN to one to indicate that the target shall enable precompensation on all signals transmitted during DT DATA phases (see 4.9, 7.3.2, and 10.8.4.1). The initiator port shall set PCOMP_EN to zero to indicate that the target shall disable precompensation.

The target port shall set PCOMP_EN to one to indicate that the initiator port shall enable precompensation on all signals transmitted during DT DATA phases (see 4.9, 7.3.2, and 10.8.4.1). The target port shall set PCOMP_EN to zero to indicate that the initiator port shall disable precompensation.

Table 9 defines valid combinations of PCOMP_EN and other fields. Ports that support fast-160 shall support enabling and disabling precompensation of their drivers.

> NOTE nn: Unlike other fields and bits in the PPR message the PCOMP_EN bit is not a negotiated value; instead, it instructs the receiving SCSI device as to whether or not precompensation is to be disabled or enabled. Because of this, precompensation may be enabled on one of the SCSI devices and disabled on the other SCSI device at the completion of a successful PPR negotiation.

4.1.5 Negotiable field combinations

Not all combinations of the negotiable fields are valid. Only the combinations defined in Table 9 shall be allowed. All other combinations of the listed fields are reserved.

Table 9. Valid negotiable field combinations

Transfer period factor	REQ/ACK Offset	Transfer width exponent	Protocol options PCOMP_EN	RTI	RD_STRM	WR_FLOW	HOLD_MCS	QAS_REQ	DT_REQ	IU_REQ	Description
0Ah – FFh	00h	00h or 01h	0	0	0	0	0	0	0	0	Use ST DATA IN and ST DATA OUT phases to transfer data with asynchronous data transfers
0Ah - FFh	01h - FFh	00h or 01h	0	0	0	0	0	0	0	0	Use ST DATA IN and ST DATA OUT phases to transfer data with synchronous data transfers
09h - FFh	01h - FFh	01h	0	0	0	0	0	0	1	0	Use DT DATA IN and DT DATA OUT phases with data group transfers
09h - FFh	01h - FFh	01h	0	0	0	0	0	1	1	0	Use DT DATA IN and DT DATA OUT phases with data group transfers, and participate in QAS arbitrations
09h – FFh	01h – FFh	01h	0	0	0 or 1	0 or 1	0	0	1	1	Use DT DATA IN and DT DATA OUT phases with information unit transfers
08h	01h - FFh	01h	0 or 1	0 or 1	0 or 1	0 or 1	0 or 1	0	1	1	Use DT DATA IN and DT DATA OUT phases with information unit transfers
09h – FFh	01h - FFh	01h	0	0	0 or 1	0 or 1	0	1	1	1	Use DT DATA IN and DT DATA OUT phases with information unit transfers, participate in QAS arbitrations, and ~~use QAS for arbitration~~issue QAS_REQUEST messages to initiate QAS arbitrations
08h	01h - FFh	01h	0 or 1	0 or 1	0 or 1	0 or 1	0 or 1	1	1	1	Use DT DATA IN and DT DATA OUT phases with information unit transfers, participate in QAS arbitrations, and issue QAS_REQUEST messages to initiate QAS arbitrations~~use QAS for arbitration~~

4.1.6 Message restrictions

PPR may be originated by initiator ports but shall not be originated by target ports. Initiator ports should only use PPR when requesting values not attainable via WDTR and SDTR (e.g., ~~selecting a transfer period factor less than 0Ah or~~ setting any protocol option bits to 1). If a target port responds to PPR with values that are attainable via WDTR and SDTR, the initiator port should repeat negotiation starting with a WDTR and SDTR negotiation sequence. This ensures that bus expanders that do not support PPR are still able to handle the data phase correctly.

WDTR and SDTR may be originated by either target ports or initiator ports. Since WDTR resets all the values that SDTR sets (see 4.1.4.1), it shall be sent first if both are needed.

4.1.7 Negotiation message sequences

4.1.7.1 Negotiation message sequences overview

An initiator originated negotiation sequence contains up to four steps:
1) Initiator port's originating message;
2) Target port response;
3) Initiator port response; and
4) Target port second response.

A target originated negotiation sequence contains up to four steps:
1) Target port's originating message;
2) Initiator port response;
3) Target port response; and
4) Initiator port second response.

4.1.7.2 Initiator originated PPR negotiation

Figure 2 shows how the initiator port shall respond to various target port responses to an originating PPR OUT. The initiator port shall maintain the previous transfer agreement unless otherwise indicated.

1. Initiator port's originating message	2. Target port response	3. Initiator port response
PPR OUT	**MESSAGE IN Phase**	
	PPR IN	Reason: Successful negotiation message exchange. The initiator port shall update it transfer agreement. If any values are illegal, the initiator port shall issue a MESSAGE REJECT OUT and invalidate its transfer agreement. The target port then originates negotiation.
	MESSAGE REJECT IN	Reason: Target port does not support PPR. Initiator port shall invalidate its transfer agreement, create an attention condition and originate a WDTR OUT. [don't try to preserve any existing WDTR/SDTR compatible settings; renegotiate everything]
	a) first 3 bytes indicate PPR IN, then bad parity; b) bad parity in first 3 bytes; or c) first 3 bytes indicate a message other than PPR IN or MESSAGE REJECT IN, then bad parity	Reasons: a) Target port understands PPR, but responding message was corrupted; b) Target port response cannot be determined; or c) Target port faulty. Initiator port shall create an attention condition and send MESSAGE PARITY ERROR OUT. Target port retries at least once and creates a bus free condition if it gives up. Retry outcome: see other responses.
	A message other than PPR IN or MESSAGE REJECT IN	Reason: Target port faulty. Initiator port shall invalidate its transfer agreement, create an attention condition and originate a WDTR OUT.
	BUS FREE phase	Reason: Target port detected an unrecoverable parity error, or target faulty. Initiator port shall invalidate its transfer agreement, rearbitrate and select the target port, create an attention condition and originate a WDTR.
	Another bus phase	Reason: Target port faulty. Initiator port shall invalidate its transfer agreement, create an attention condition and originate a WDTR OUT.
	Hang	Reason: Target port faulty. Initiator port may toggle ATN and ACK to try to wake up the target port. Initiator port should create a bus reset condition.

Figure 2. Initiator originated PPR negotiation: initiator response

Figure 3 shows how the target port shall respond to various initiator port responses to a responding PPR IN. The target port shall maintain the previous transfer agreement unless otherwise indicated.

2. Target port's responding message	3. Initiator port response	4. Target port second response
PPR IN	**Attention condition; MESSAGE OUT phase** — MESSAGE REJECT OUT	Reason: Initiator port claims target provided incompatible values. Target port shall invalidate its transfer agreement and originate a WDTR IN.
	MESSAGE PARITY ERROR OUT	Reason: Initiator port detected a parity error. If the attention condition was created on one of the first 3 bytes, then the initiator port does not know which message was being sent. If the attention condition was created after the first 3 bytes, the initiator port knows a PPR IN was being sent. Target port shall retry the PPR IN at least once. If parity errors persist, target port shall create a bus free condition (unexpected). Retry outcome: Successful retry: see other responses Bus free condition: Target port shall invalidate its transfer agreement and originate a WDTR IN on the next connection if the initiator does not create an attention condition and originate negotiation itself.
	a) Bad parity in first byte; or b) First byte indicates a message other than MESSAGE REJECT OUT or MESSAGE PARITY ERROR OUT, then bad parity	Reasons: a) Initiator port response cannot be determined; or b) Successful negotiation, but hung. Target port shall repeat the MESSAGE OUT phase at least once and create a bus free condition if it gives up. Retry outcome: Successful retry: see other responses Bus free condition: Target port shall invalidate its transfer agreement and originate a WDTR IN.
	Another message	Reason: Successful negotiation. Target port shall update its transfer agreement. If IU_REQ was changed, target port shall ignore the attention condition and create a bus free condition.
	No attention condition	Reason: Successful negotiation. Target port shall update its transfer agreement. If IU_REQ was changed, target port shall create a bus free condition.

Figure 3. Initiator originated PPR negotiation: target response

4.1.7.3 Initiator originated WDTR negotiation

Figure 4 shows how the initiator port shall respond to various target port responses to an originating WDTR OUT. The initiator port shall maintain the previous transfer agreement unless otherwise indicated.

1. Initiator port's originating message	2. Target port response	3. Initiator port response
WDTR OUT	**MESSAGE IN Phase** WDTR IN	Reason: Successful negotiation message exchange. The initiator port shall update it transfer agreement. If any values are illegal, the initiator port shall issue a MESSAGE REJECT OUT and invalidate its transfer agreement. The target port then originates negotiation.
	MESSAGE REJECT IN	Reason: Target port does not support WDTR. Initiator port shall invalidate its transfer agreement, create an attention condition and originate an SDTR OUT. [don't try to preserve any SDTR compatible settings; renegotiate]
	a) first 3 bytes indicate WDTR IN, then bad parity; b) bad parity in first 3 bytes; or c) first 3 bytes indicate a message other than WDTR IN or MESSAGE REJECT IN, then bad parity	Reasons: a) Target port understands WDTR, but responding message was corrupted; b) Target port response cannot be determined; or c) Target port faulty. Initiator port shall create an attention condition and send MESSAGE PARITY ERROR OUT. Target port retries at least once and creates a bus free condition when it gives up. Retry outcome: see other responses.
	A message other than WDTR IN or MESSAGE REJECT IN	Reason: Target port faulty. Initiator port shall invalidate its transfer agreement, create an attention condition and originate an SDTR OUT.
	BUS FREE phase	Reason: Target port detected an unrecoverable parity error, or target port faulty. Initiator port shall invalidate its transfer agreement, rearbitrate and select the target, create an attention condition and originate an SDTR OUT.
	Another bus phase	Reason: Target port faulty. Initiator port shall invalidate its transfer agreement, create an attention condition and originate an SDTR OUT.
	Hang	Reason: Target port faulty. Initiator port may toggle ATN and ACK to try to wake up the target port. Initiator port should create a bus reset condition.

Figure 4. Initiator originated WDTR negotiation: initiator response

Figure 5 shows how the target port shall respond to various initiator port responses to a responding WDTR IN. The target port shall maintain the previous transfer agreement unless otherwise indicated.

2. Target port's responding message	3. Initiator port response	4. Target port second response
WDTR IN	**Attention condition; MESSAGE OUT phase** MESSAGE REJECT OUT	Reason: Initiator port claims target provided incompatible values. Target port shall invalidate its transfer agreement and originate an SDTR IN.
	MESSAGE PARITY ERROR OUT	Reason: Initiator port detected a parity error. If the attention condition was created on one of the first 3 bytes, then the initiator does not know which message was being sent. If the attention condition was created after the first 3 bytes, the initiator port knows a WDTR IN was being sent. Target port shall retry the WDTR IN at least once. If parity errors persist, target port shall create a bus free condition (unexpected). Retry outcome: Successful retry: see other responses Bus free condition: Target port shall invalidate its transfer agreement and originate an SDTR IN on the next connection if the initiator does not create an attention condition and originate negotiation itself.
	a) Bad parity in first byte; or b) First byte indicates a message other than MESSAGE REJECT OUT or MESSAGE PARITY ERROR OUT, then bad parity	Reasons: a) Initiator response cannot be determined; or b) Successful negotiation, but hung. Target port shall repeat the MESSAGE OUT phase at least once and create a bus free condition if it gives up. Retry outcome: Successful retry: see other responses Bus free condition: Target port shall invalidate agreement and originate an SDTR IN.
	Another message	Reason: Successful negotiation. Target port shall update its transfer agreement. If IU_REQ was changed, target port shall ignore the attention condition and create a bus free condition.
	No attention condition	Reason: Successful negotiation. Target port shall update its transfer agreement. If IU_REQ was changed, target port shall create a bus free condition.

Figure 5. Initiator originated WDTR negotiation: target response

4.1.7.4 Initiator originated SDTR negotiation

Figure 6 shows how the initiator port shall respond to various target port responses to an originating SDTR OUT. The initiator port shall maintain the previous transfer agreement unless otherwise indicated.

1. Initiator port's originating message	2. Target port response	3. Initiator port response
SDTR OUT	**MESSAGE IN Phase**	
	SDTR IN	Reason: Successful negotiation message exchange. The initiator port shall update it transfer agreement. If any values are illegal, the initiator port shall issue a MESSAGE REJECT OUT and invalidate its transfer agreement. The target port then originates negotiation.
	MESSAGE REJECT IN	Reason: Target port does not support SDTR. The initiator port shall set its transfer agreement to the default transfer agreement.
	a) first 3 bytes indicate SDTR IN, then bad parity; b) bad parity in first 3 bytes; or c) first 3 bytes indicate a message other than SDTR IN or MESSAGE REJECT IN, then bad parity	Reasons: a) Target port understands SDTR, but responding message was corrupted; b) Target port response cannot be determined; or c) Target port faulty. Initiator port shall create an attention condition and send MESSAGE PARITY ERROR OUT. Target port retries at least once and creates a bus free condition when it gives up. Retry outcome: see other responses.
	A message other than SDTR IN or MESSAGE REJECT IN	Reason: Target port faulty. Initiator port shall set its transfer agreement to the default transfer agreement.
	BUS FREE phase	Reason: Target port detected an unrecoverable parity error, or target port faulty. Initiator port shall set its transfer agreement to the default transfer agreement. [parity error => keep, fauly => default]
	Another bus phase	Reason: Target port faulty. Initiator port shall set its transfer agreement to the default transfer agreement.
	Hang	Reason: Target port faulty. Initiator port may toggle ATN and ACK to try to wake up the target port. Initiator port should create a bus reset condition.

Figure 6. Initiator originated SDTR negotiation: initiator response

Figure 7 shows how the target port shall respond to various initiator port responses to a responding SDTR IN. The target port shall maintain the previous transfer agreement unless otherwise indicated.

2. Target port's responding message	3. Initiator port response	4. Target port second response
SDTR IN	**Attention condition; MESSAGE OUT phase**	
	MESSAGE REJECT OUT	Reason: Initiator port claims target provided incompatible values. If any value was not set to zero, the target port shall originate an SDTR IN with every value set to zero. If every value was zero, the target port shall set its transfer agreement to the default transfer agreement.
	MESSAGE PARITY ERROR OUT	Reason: Initiator port detected a parity error. If the attention condition was created on one of the first 3 bytes, then the initiator port does not know which message was being sent. If the attention condition was created after the first 3 bytes, the initiator port knows a SDTR IN was being sent. Target port shall retry the SDTR IN at least once. If parity errors persist, target port shall create a bus free condition (unexpected). Retry outcome: Successful retry: see other responses Bus free condition: Target port shall invalidate its transfer agreement and originate an SDTR negotiation on the next connection if the initiator does not create an attention condition and originate negotiation itself.
	a) Bad parity in first byte; or b) First byte indicates a message other than MESSAGE REJECT OUT or MESSAGE PARITY ERROR OUT, then bad parity	Reasons: a) Initiator port response cannot be determined; or b) Successful negotiation, but hung. Target port shall repeat the MESSAGE OUT phase at least once and create a bus free condition if it gives up. [a => invalidate/default, b => update… too complex?] Retry outcome: Successful retry: see other responses Bus free condition: Target port shall set its transfer agreement to the default transfer agreement.
	Another message	Reason: Successful negotiation. Target port shall update its transfer agreement. If IU_REQ was changed, target port shall ignore the attention condition and create a bus free condition.
	No attention condition	Reason: Successful negotiation. Target port shall update its transfer agreement. If IU_REQ was changed, target port shall create a bus free condition.

Figure 7. Initiator originated SDTR negotiation: target response

4.1.7.5 Target originated WDTR negotiation

Figure 8 shows how the target port shall respond to various initiator port responses to an originating WDTR IN. The target port shall maintain the previous transfer agreement unless otherwise indicated.
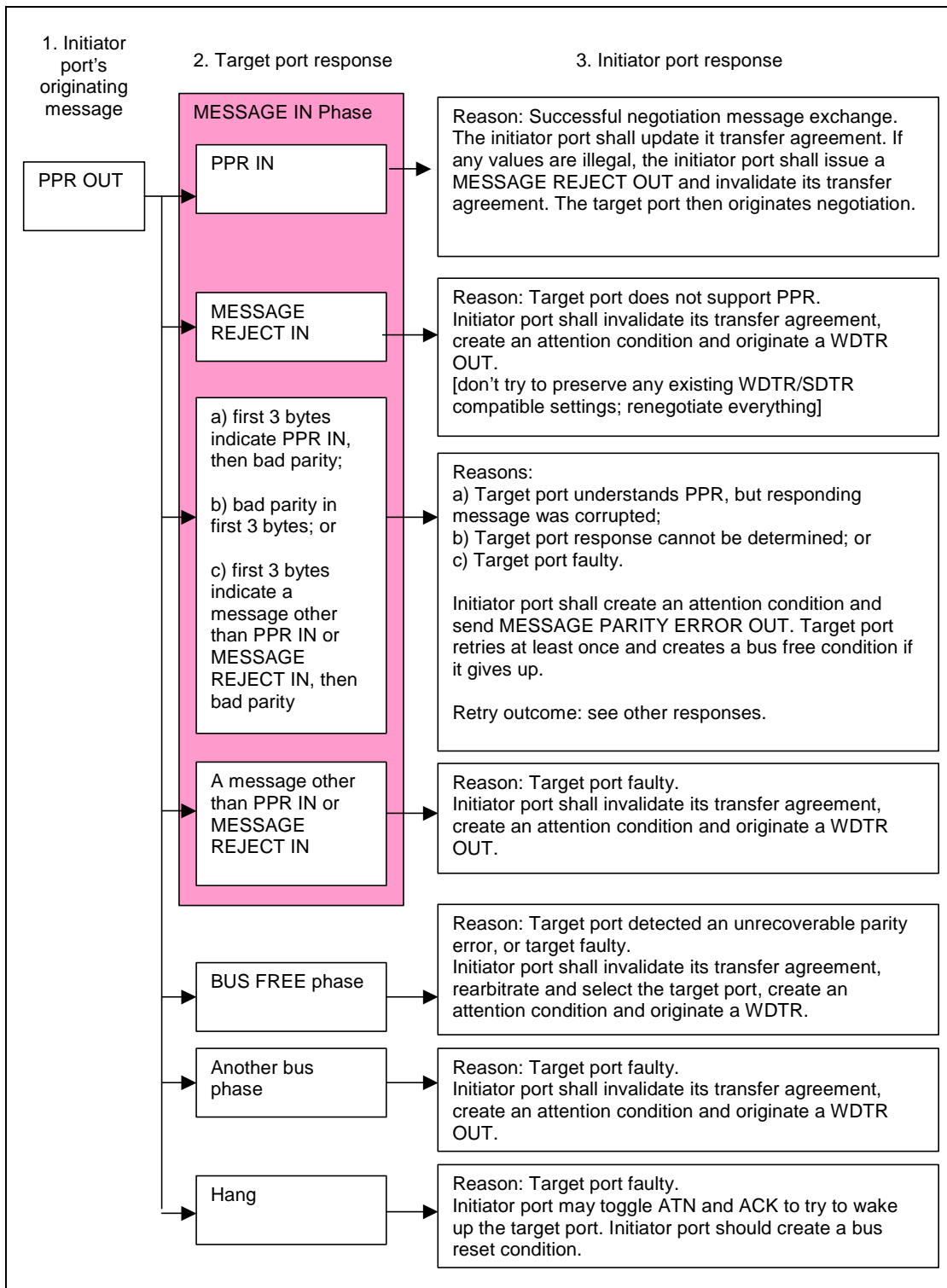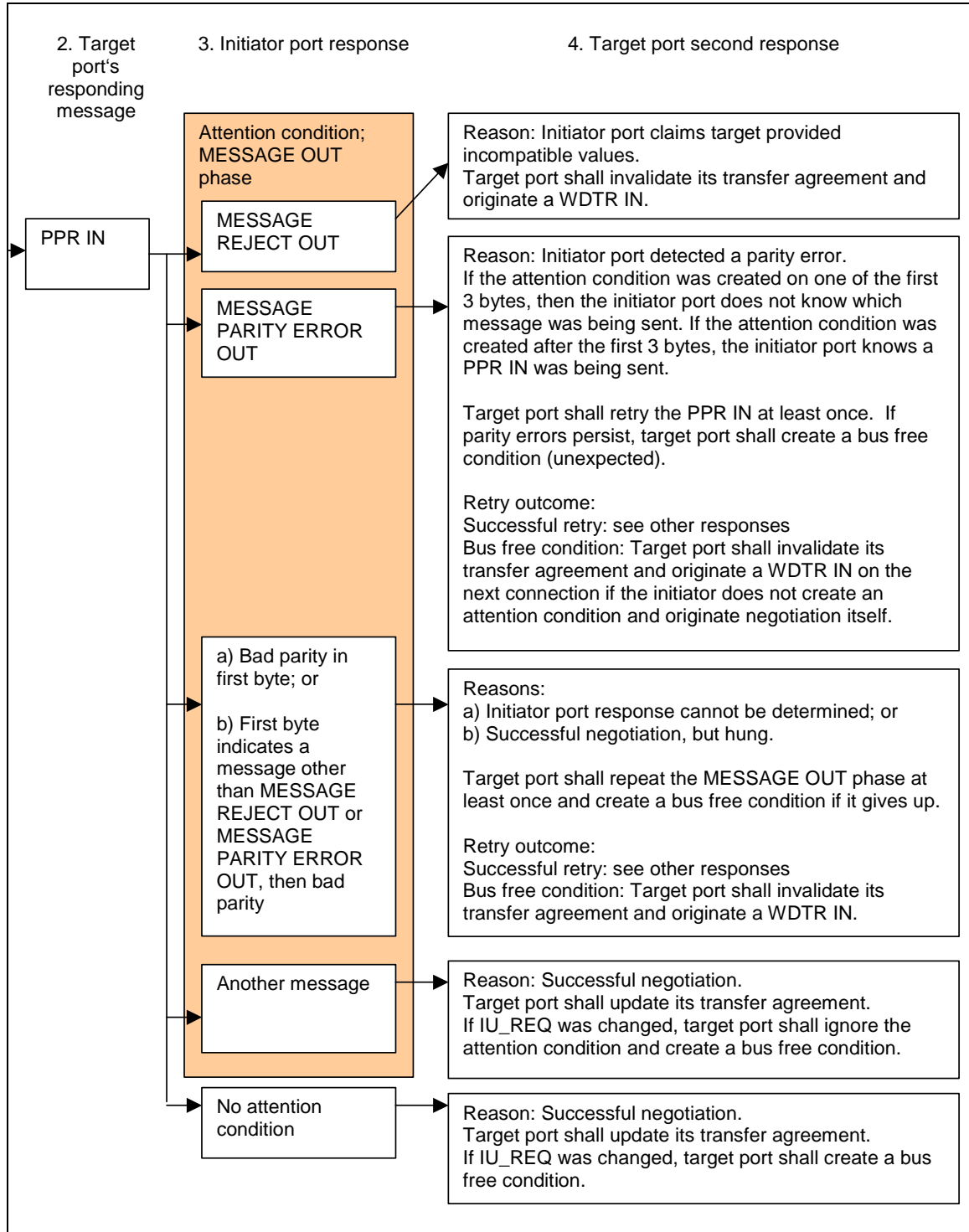
1. Target port's originating message	2. Initiator port response	3. Target port response
WDTR IN	Attention condition; MESSAGE OUT phase — WDTR OUT	Reason: Successful negotiation message exchange. If any values are illegal, target port shall issue a MESSAGE REJECT IN and invalidate its transfer agreement and originate an SDTR IN if initiator does not originate negotiation first. Otherwise, target port shall update its transfer agreement. If IU_REQ was changed, target port shall create a bus free condition.
	MESSAGE REJECT OUT	Reason: Initiator port does not support WDTR. Target port shall invalidate its transfer agreement and originate an SDTR IN. [don't try to preserve SDTR settings; renegotiate]
	MESSAGE PARITY ERROR OUT	Reason: Initiator port detected a parity error. Target port shall retry the WDTR IN at least once. If parity errors persist, target port shall create a bus free condition (unexpected). Note: If the attention condition was created on one of the first 3 bytes, then the initiator does not know which message was being sent. If the attention condition was created after the first 3 bytes, the initiator port knows a WDTR IN was being sent. Retry outcome: Successful retry: see other responses Bus free condition: Target port shall invalidate its transfer agreement and originate an SDTR on the next connection if the initiator does not create an attention condition and do so first.
	a) Bad parity in first byte; or b) First byte indicates a message other than WDTR OUT, MESSAGE REJECT OUT or MESSAGE PARITY ERROR OUT, then bad parity	Reasons: a) Initiator response cannot be determined; or b) Initiator port faulty. Target port shall repeat the MESSAGE OUT phase at least once and create a bus free condition if it gives up. Retry outcome: Successful retry: see other responses Bus free condition: Target port shall invalidate its transfer agreement and originate an SDTR IN.
	No attention condition	Reason: Initiator port faulty. Target port shall invalidate its transfer agreement and originate an SDTR if the initiator does not create an attention condition and do so first.

Figure 8. Target originated WDTR negotiation: target response

Figure 9 shows how the initiator port shall respond to various target port responses to a responding WDTR OUT. The initiator port shall maintain the previous transfer agreement unless otherwise indicated.
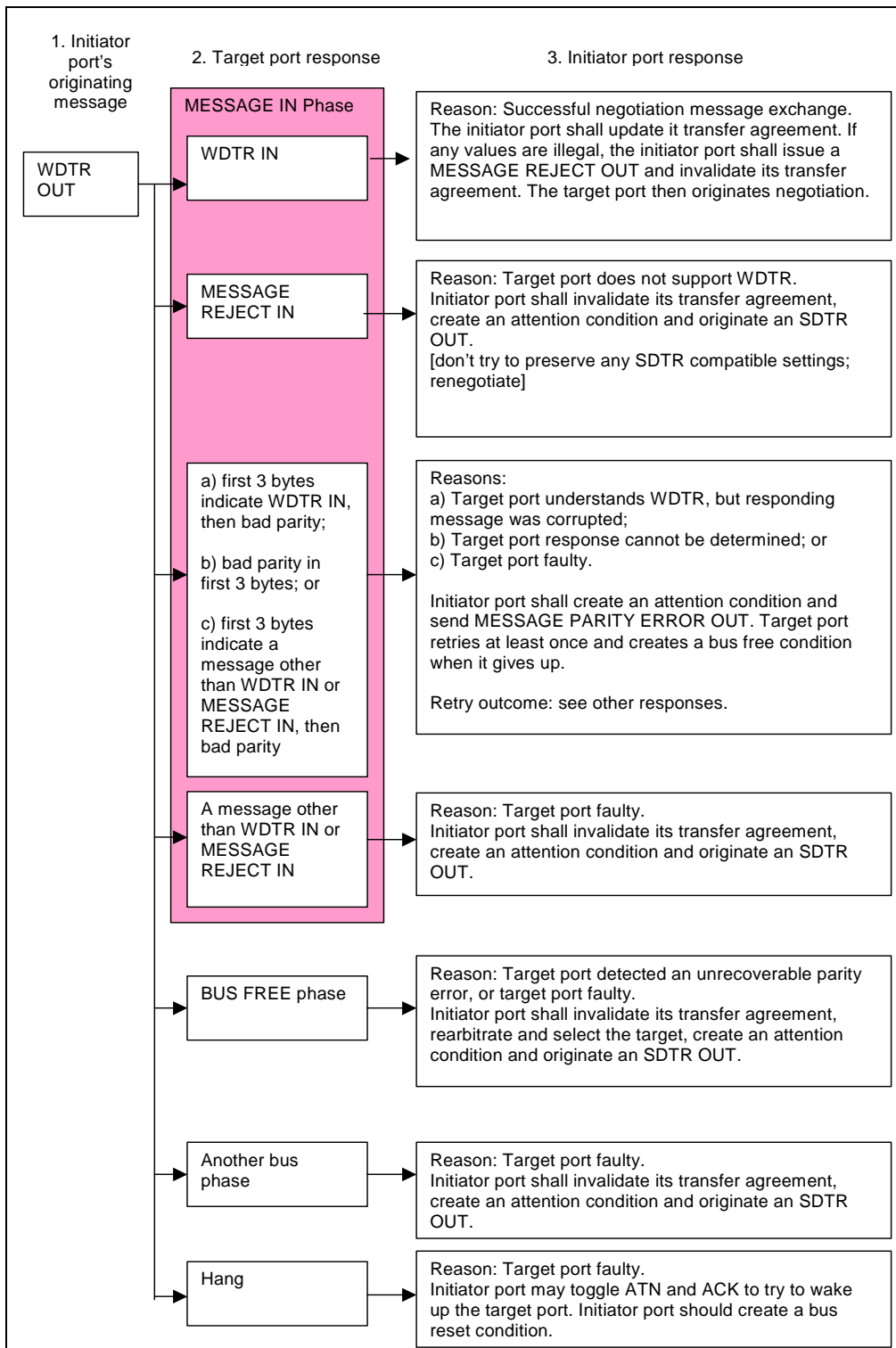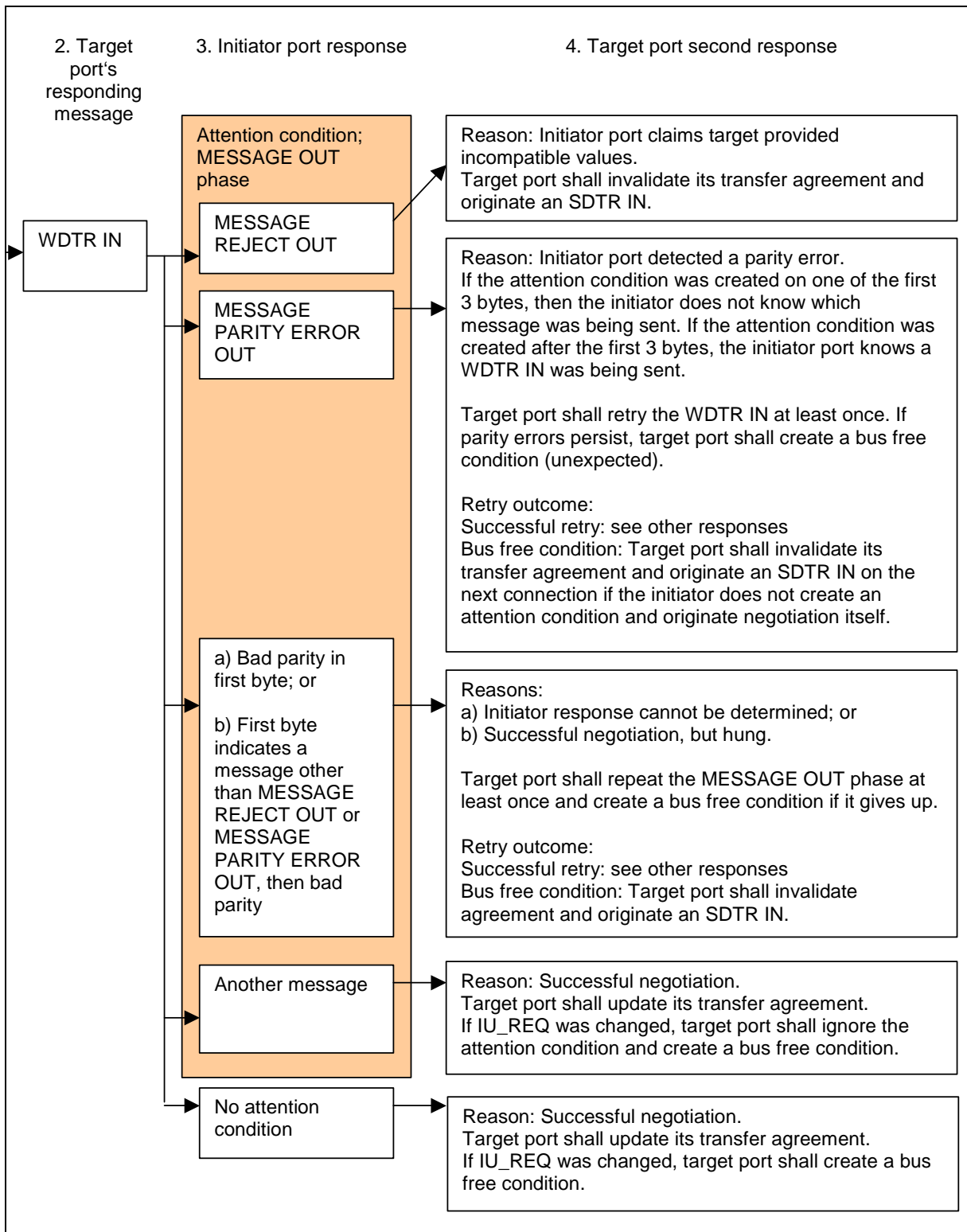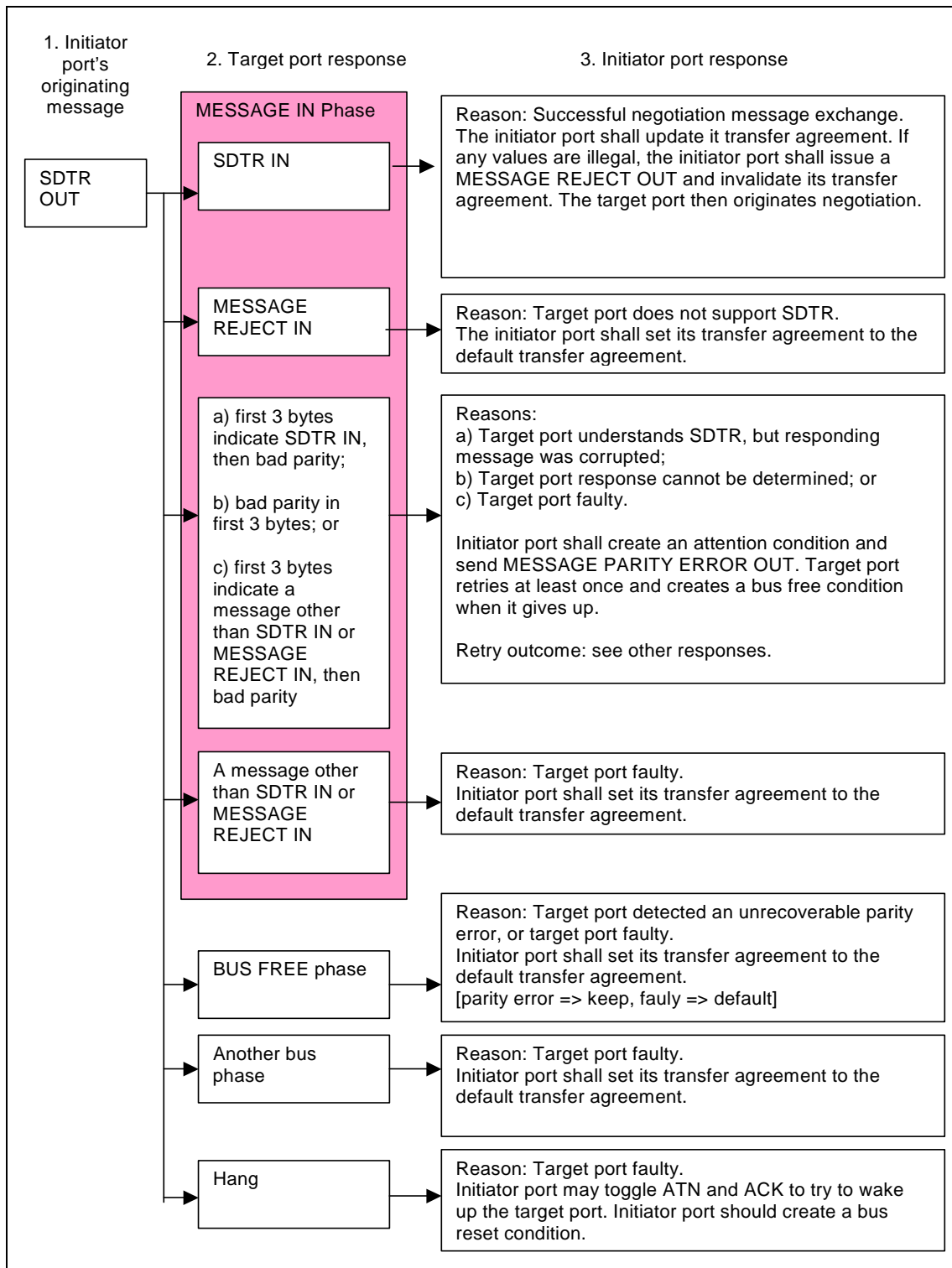
2. Initiator port's responding message	3. Target port response	4. Initiator port second response
WDTR OUT	**MESSAGE IN Phase**	
	MESSAGE REJECT IN	Reason: Target port claims initiator port provided incompatible values. Initiator port shall invalidate its transfer agreement, create an attention condition and originate an SDTR OUT.
	Another message	Reason: Successful negotiation message exchange. If any values are illegal, initiator port shall issue a MESSAGE REJECT OUT and invalidate its transfer agreement. The initiator port shall then originates negotiation. Otherwise, initiator port shall update its transfer agreement.
	a) first 3 bytes indicate WDTR IN, then bad parity; b) bad parity in first 3 bytes; or c) first 3 bytes indicate a message other than MESSAGE REJECT IN, then bad parity	Reasons: a) Target port understands WDTR, but responding message was corrupted; b) Target port response cannot be determined; or c) Target port faulty. Initiator port shall create an attention condition and send MESSAGE PARITY ERROR OUT. Target port retries at least once and creates a bus free condition when it gives up. Target port retry outcome: see other responses.
	BUS FREE phase	Reason: Target port detected an unrecoverable parity error, IU_REQ changed, or target faulty. Initiator port shall invalidate its transfer agreement, rearbitrate and select the target, create an attention condition and originate an SDTR.
	Another bus phase	Reason: Successful negotiation. Initiator port shall update its transfer agreement.
	Hang	Reason: Target port faulty. Initiator port should toggle ATN and ACK to try to wake up the target port. Finally, initiator port should create a bus reset condition.

Figure 9. Target originated WDTR negotiation: initiator response

4.1.7.6 Target originated SDTR negotiation

Figure 10 shows how the target port shall respond to various initiator port responses to an originating SDTR IN. The target port shall maintain the previous transfer agreement unless otherwise indicated.
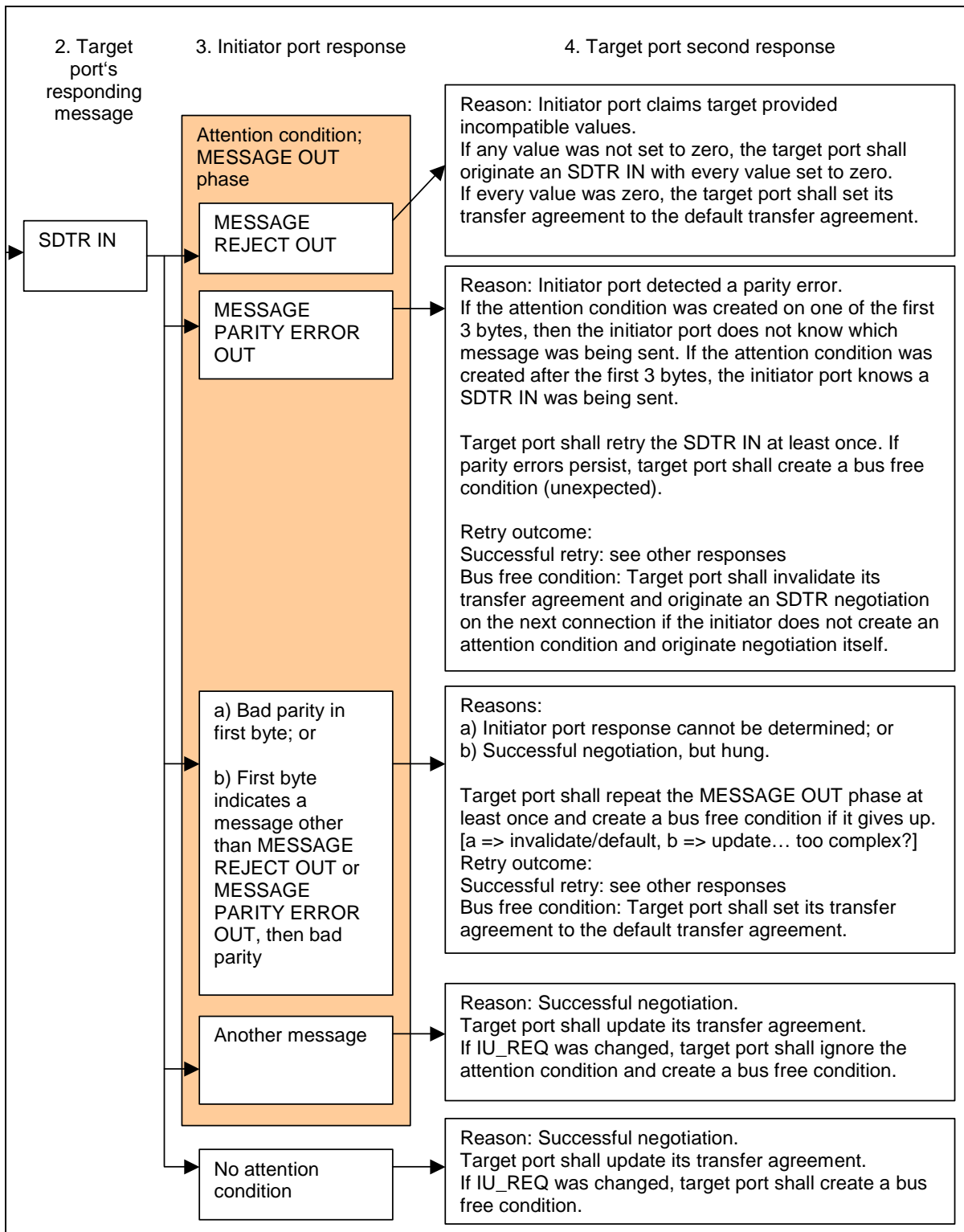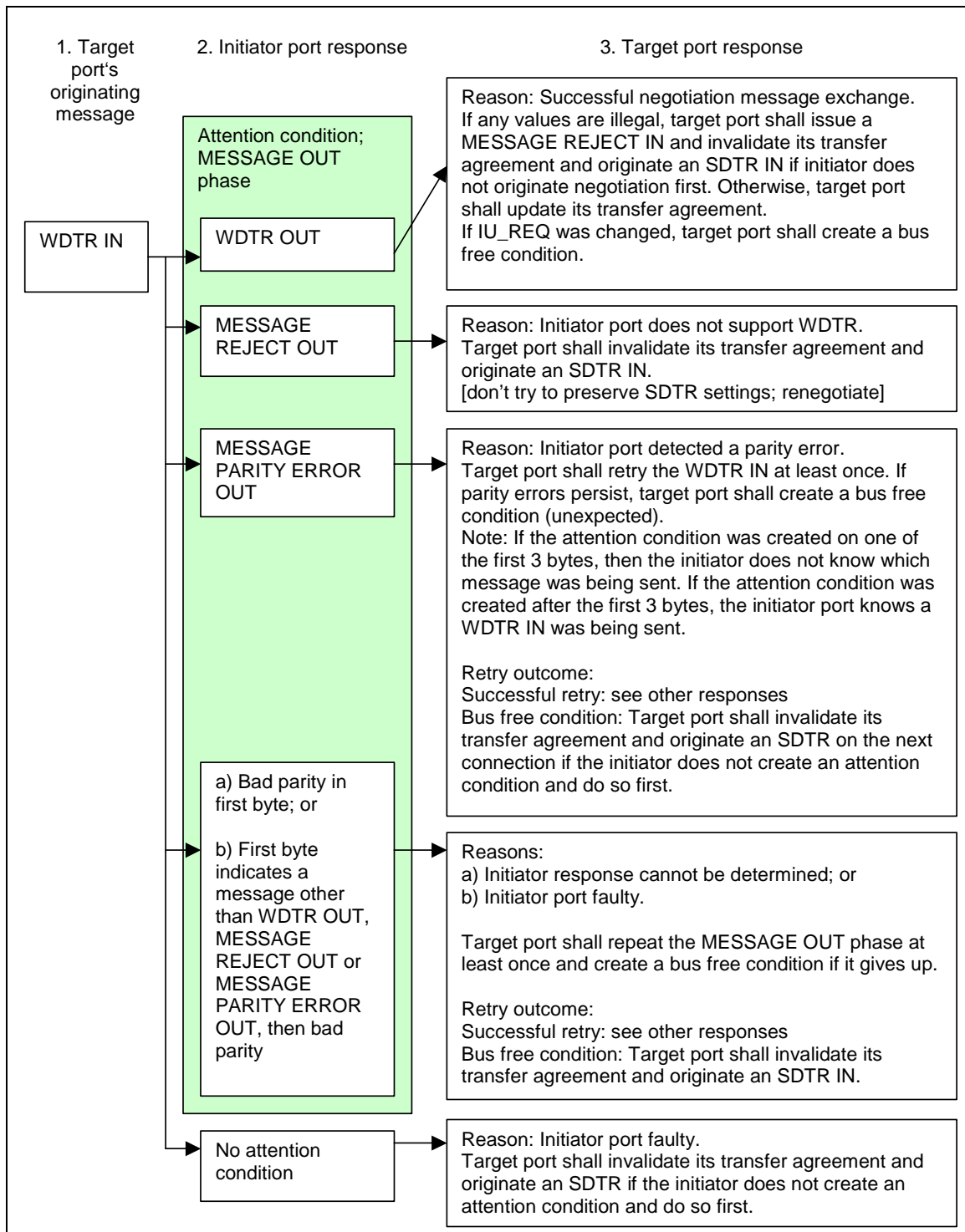
1. Target port's originating message	2. Initiator port response	3. Target port response
SDTR IN	Attention condition; MESSAGE OUT phase SDTR OUT	Reason: Successful negotiation message exchange. If any values are illegal, target port shall issue a MESSAGE REJECT IN and set its transfer agreement to the default transfer agreement. Otherwise, target port shall update its transfer agreement. If IU_REQ was changed, target port shall create a bus free condition.
	MESSAGE REJECT OUT	Reason: Initiator port does not support SDTR. Target port shall set its transfer agreement to the default transfer agreement.
	MESSAGE PARITY ERROR OUT	Reason: Initiator port detected a parity error. Target port shall retry the SDTR IN at least once. If parity errors persist, target port shall create a bus free condition (unexpected). If the attention condition was created on one of the first 3 bytes, then the initiator port does not know which message was being sent. If the attention condition was created after the first 3 bytes, the initiator port knows an SDTR IN was being sent. Retry outcome: Successful retry: see other responses Bus free condition: Target port shall set its transfer agreement to the default transfer agreement.
	a) Bad parity in first byte; or b) First byte indicates a message other than SDTR OUT, MESSAGE REJECT OUT or MESSAGE PARITY ERROR OUT, then bad parity	Reasons: a) Initiator port response cannot be determined; or b) Initiator port faulty. Target port shall repeat the MESSAGE OUT phase at least once and create a bus free condition if it gives up. Retry outcome: Successful retry: see other responses Bus free condition: Target port shall set its transfer agreement to the default transfer agreement.
	No attention condition	Reason: Initiator port faulty. Target port shall set its transfer agreement to the default transfer agreement.

Figure 10. Target originated SDTR negotiation: target response

Figure 11 shows how the initiator port shall respond to various target port responses to a responding SDTR OUT. The initiator port shall maintain the previous transfer agreement unless otherwise indicated.
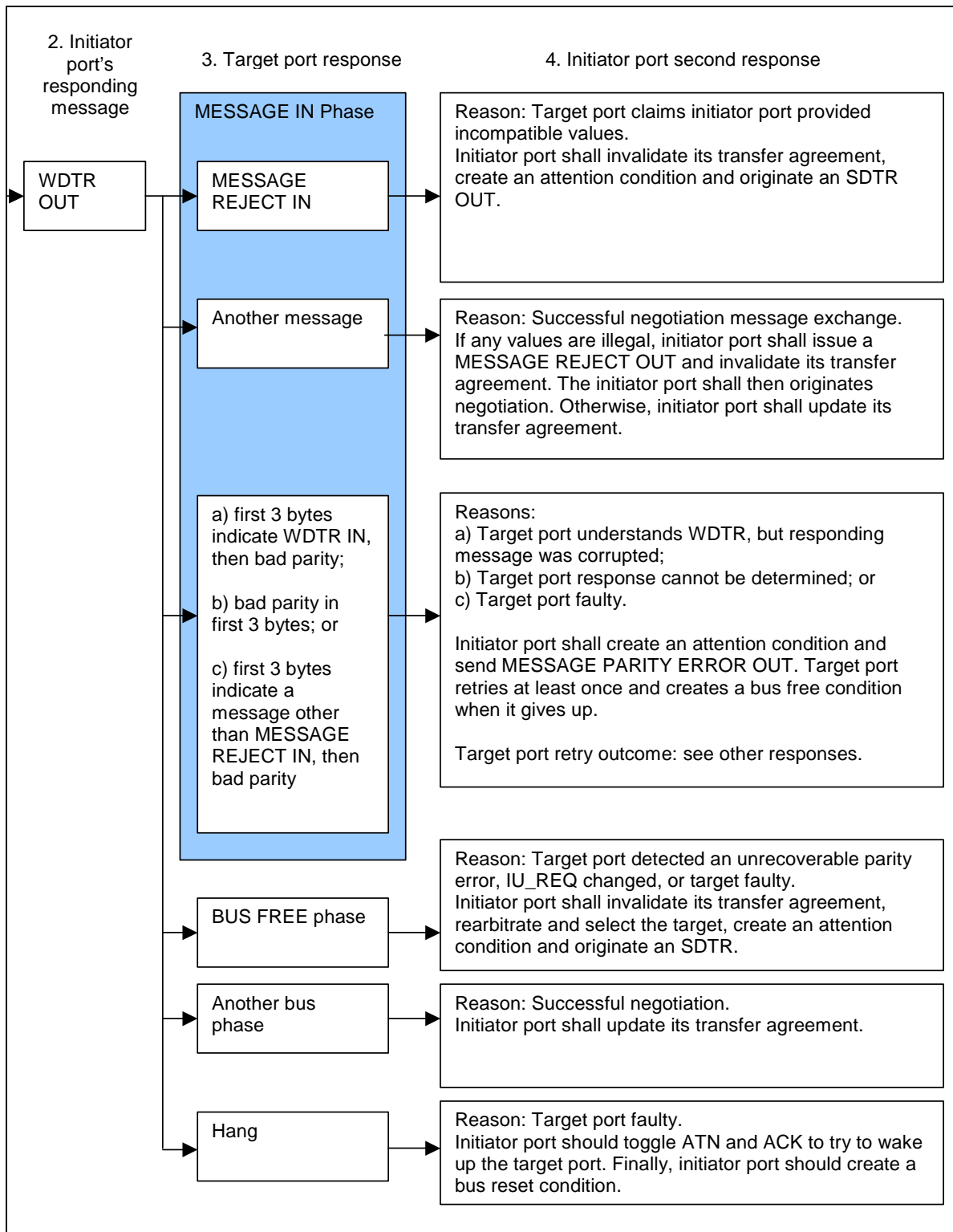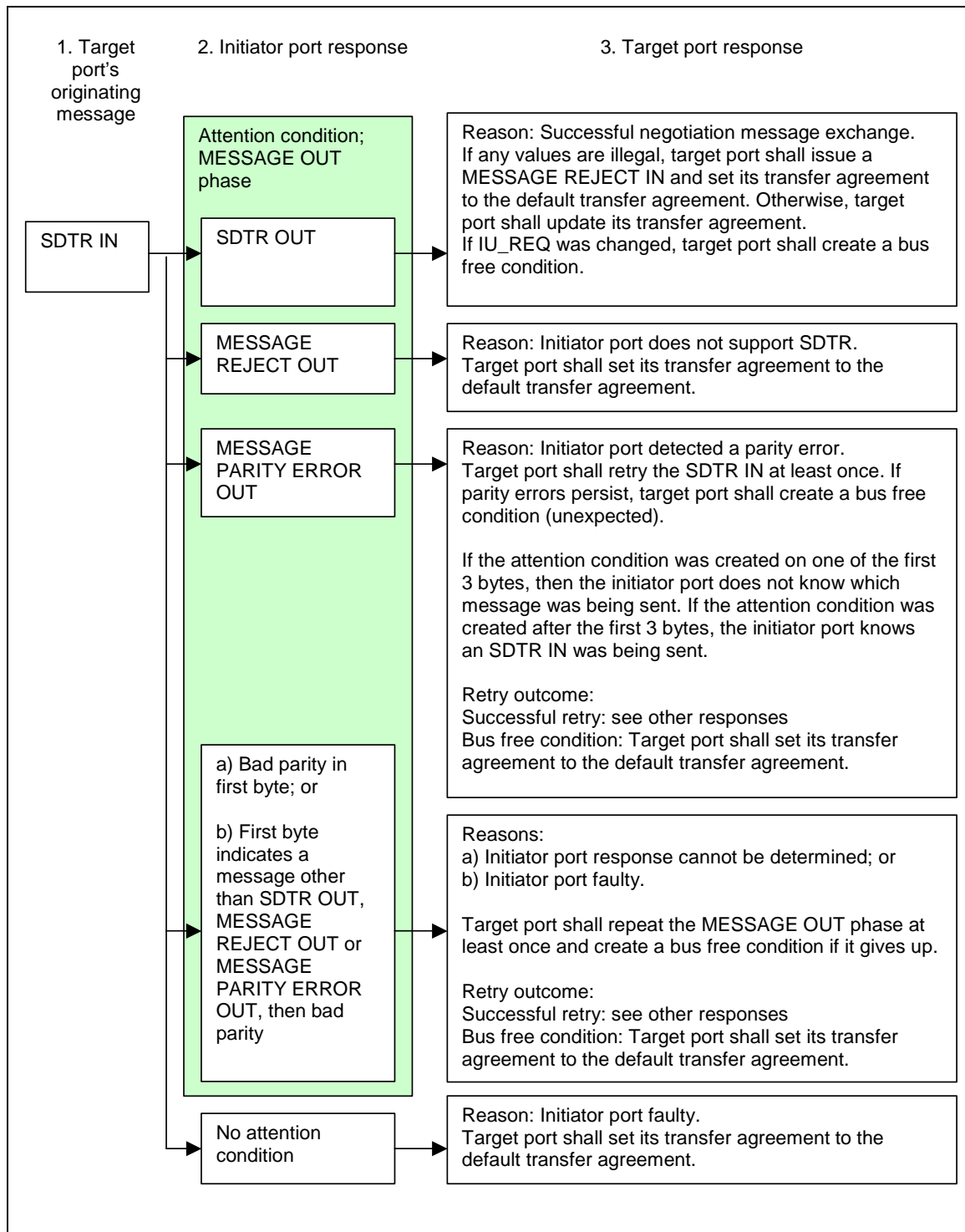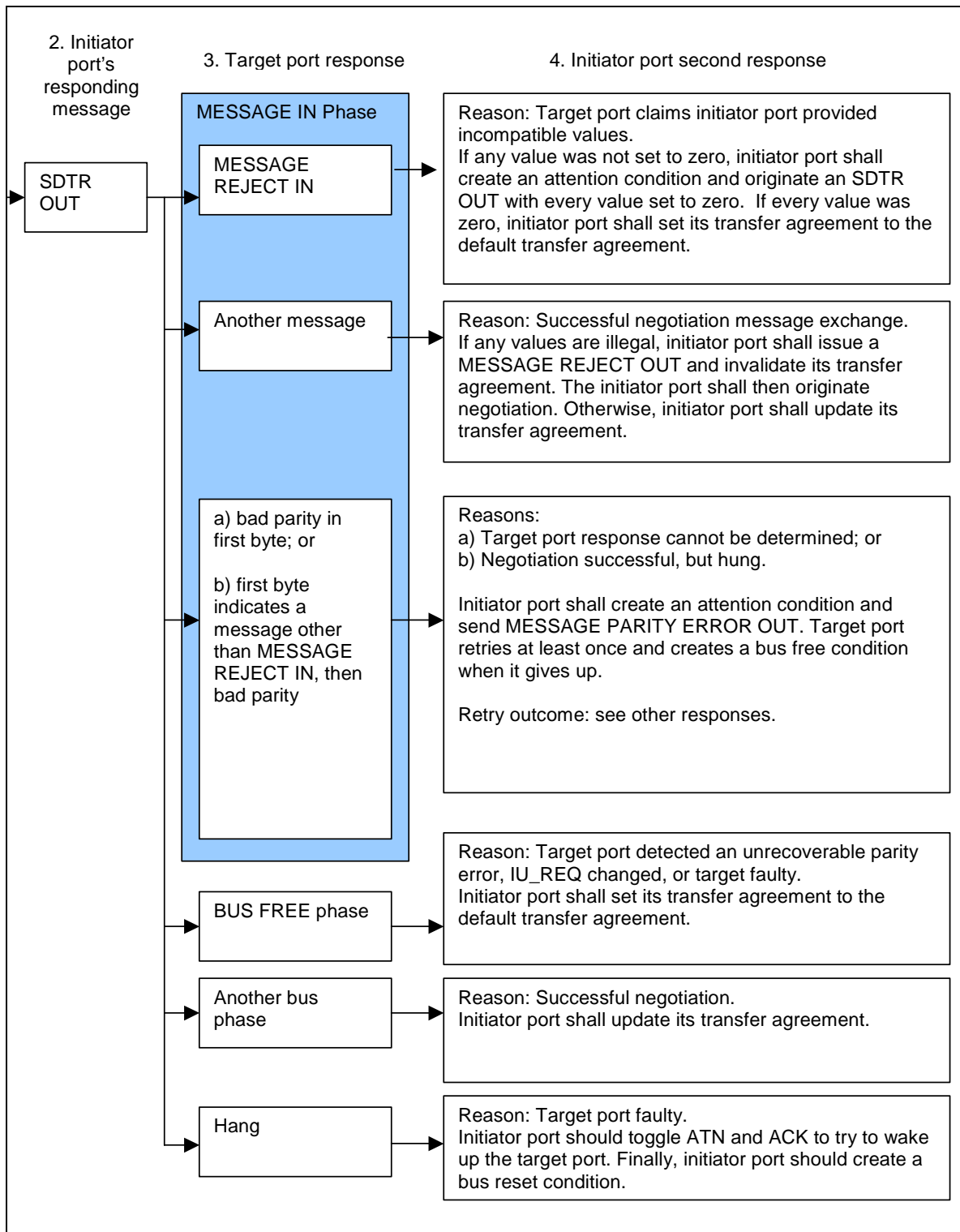
2. Initiator port's responding message	3. Target port response	4. Initiator port second response
SDTR OUT	**MESSAGE IN Phase** MESSAGE REJECT IN	Reason: Target port claims initiator port provided incompatible values. If any value was not set to zero, initiator port shall create an attention condition and originate an SDTR OUT with every value set to zero. If every value was zero, initiator port shall set its transfer agreement to the default transfer agreement.
	Another message	Reason: Successful negotiation message exchange. If any values are illegal, initiator port shall issue a MESSAGE REJECT OUT and invalidate its transfer agreement. The initiator port shall then originate negotiation. Otherwise, initiator port shall update its transfer agreement.
	a) bad parity in first byte; or b) first byte indicates a message other than MESSAGE REJECT IN, then bad parity	Reasons: a) Target port response cannot be determined; or b) Negotiation successful, but hung. Initiator port shall create an attention condition and send MESSAGE PARITY ERROR OUT. Target port retries at least once and creates a bus free condition when it gives up. Retry outcome: see other responses.
	BUS FREE phase	Reason: Target port detected an unrecoverable parity error, IU_REQ changed, or target faulty. Initiator port shall set its transfer agreement to the default transfer agreement.
	Another bus phase	Reason: Successful negotiation. Initiator port shall update its transfer agreement.
	Hang	Reason: Target port faulty. Initiator port should toggle ATN and ACK to try to wake up the target port. Finally, initiator port should create a bus reset condition.

Figure 11. Target originated SDTR negotiation: initiator response

4.1.7.7 Asynchronous transfers

SCSI parallel interface devices default to 8-bit asynchronous transfers.

The 8-bit asynchronous transfers are used for all information transfers except DATA phases. ST DATA phases use 16-bit asynchronous transfers when a wide transfer agreement (see 10.8.5) is in effect. Asynchronous transfers are not permitted when DT DATA phases are enabled.

4.1.7.8 Synchronous transfers

Synchronous transfers shall only be used in DATA phases when a synchronous transfer agreement is in effect (see 10.8.5).

ST DATA phases shall transfer data using synchronous transfers when a ST DATA phase enabled agreement is in effect. ST DATA phases use 16-bit synchronous transfers when a wide transfer agreement (see 10.8.5) is in effect.

DT DATA phases shall transfer data using synchronous transfers when a DT DATA phase enabled agreement is in effect (see 10.8.5). DT DATA phases shall only use 16-bit synchronous transfers.

4.1.7.9 Paced transfers

Paced transfers shall only be used in DT DATA phases when a fast-160 agreement is in effect (see 10.8.5). DT DATA phases shall only use 16-bit paced transfers.

10.6 SELECTION phase

10.6.3 Selection without using attention condition

10.6.3.1 Information unit transfers enabled or disabled

…
If an initiator, when selecting without using an attention condition, detects an unexpected COMMAND phase it ~~should~~ shall invalidate all prior negotiations with the selected target. In this case, the initiator shall create an attention condition and on the corresponding MESSAGE OUT phase shall issue an ABORT TASK message. On the next selection of the target that received the ABORT TASK message the initiator should do a selection using the attention condition and negotiate to enable information unit transfers.

16.1 Negotiation messages

[Editor's note: create a new section 16.x parallel to the Link control messages section.]

16.1.1 Negotiation message codes

[Editor's note: Move PPR, WDTR, and SDTR out of the Link control message codes table into a new table here. Split the OUT and IN messages into separate rows]

Table 10. Negotiation message codes

Extended Message, Length, and Code	Init	Targ	Message Name	In	Out	Clear Attention Condition
01h, 06h, 04h	O	O	PARALLEL PROTOCOL REQUEST (PPR) ~~OUT~~	In	Out	Yes
~~01h, 06h, 04h~~	~~O~~	~~O~~	~~PARALLEL PROTOCOL REQUEST (PPR) IN~~	~~In~~		
01h, 03h, 01h	O	O	SYNCHRONOUS DATA TRANSFER REQUEST (SDTR) ~~OUT~~	In	Out	Yes
~~01h, 03h, 01h~~	~~O~~	~~O~~	~~SYNCHRONOUS DATA TRANSFER REQUEST (SDTR) IN~~	~~In~~		
01h, 02h, 03h	O	O	WIDE DATA TRANSFER REQUEST (WDTR) ~~OUT~~	In	Out	Yes
~~01h, 02h, 03h~~	~~O~~	~~O~~	~~WIDE DATA TRANSFER REQUEST (WDTR) IN~~	~~In~~		
Key: M=Mandatory support, O=Optional support In=Target to initiator, Out=Initiator to target Yes=When sending the message in a MESSAGE OUT phase, the initiator shall clear the attention condition before the last ACK of the MESSAGE OUT phase. Init=initiator, Targ=target						

16.1.2 PARALLEL PROTOCOL REQUEST message description

PARALLEL PROTOCOL REQUEST (PPR) messages (see Table 11) are used to negotiate the transfer period factor, REQ/ACK offset, transfer width exponent, and protocol options between two SCSI devices.

Table 11. PARALLEL PROTOCOL REQUEST message format

Bit Byte	7	6	5	4	3	2	1	0
0	Extended Message (01h)							
1	EXTENDED MESSAGE LENGTH (06h)							
2	Parallel Protocol Request (04h)							
3	TRANSFER PERIOD FACTOR							
4	Reserved							
5	REQ/ACK OFFSET							
6	TRANSFER WIDTH EXPONENT *["m" removed]*							
7	PROTOCOL OPTIONS							
	PCOMP_EN	RTI	RD_STRM	WR_FLOW	HOLD_MCS	QAS_REQ	DT_REQ	IU_REQ

PPR messages shall be supported by ports supporting transfer period factors less than 0Ah or supporting any of the protocol options. PPR messages shall be supported by target ports with a CLOCKING field indicating DT support, IUS set to 1, or QAS set to 1 in the INQUIRY page of all their logical units (see SPC-2).

Usage of this message is defined in 4.1. Fields are defined in 4.1.4.

16.1.3 SYNCHRONOUS DATA TRANSFER REQUEST message description

SYNCHRONOUS DATA TRANSFER REQUEST (SDTR) messages (see Table 12) are used to negotiate the transfer period factor and REQ/ACK offset between two SCSI devices.

Table 12. SYNCHRONOUS DATA TRANSFER REQUEST message format

Byte	
0	Extended Message (01h)
1	EXTENDED MESSAGE LENGTH (03h)
2	Synchronous Data Transfer Request (01h)
3	TRANSFER PERIOD FACTOR
4	REQ/ACK OFFSET

SDTR messages shall be supported by devices supporting synchronous data transfers (i.e., non-zero REQ/ACK offsets). SDTR messages shall be supported by target ports with SYNC set to 1 in the INQUIRY page of all their logical units (see SPC-2).

~~Only transfer period factors greater than or equal to 0Ah shall be negotiated with SDTR. PPR shall be used for transfer period factors less than 0Ah.~~

Usage of this message is defined in 4.1. Fields are defined in 4.1.4.

16.1.4 WIDE DATA TRANSFER REQUEST message description

WIDE DATA TRANSFER REQUEST (WDTR) messages (see Table 13) are used to negotiate the transfer width exponent between two SCSI devices.

Table 13. WIDE DATA TRANSFER REQUEST message format

Byte	
0	Extended Message (01h)
1	EXTENDED MESSAGE LENGTH (02h)
2	Wide Data Transfer Request (03h)
3	TRANSFER WIDTH EXPONENT *["m" removed]*

WDTR messages shall be supported by ports supporting wide data transfers (i.e., non-zero transfer width exponents). WDTR messages shall be supported by target ports with WBUS16 set to 1 in the INQUIRY page of all their logical unit (see SPC-2).

Usage of this message is defined in 4.1. Fields are defined in 4.1.4.

Additional changes

References to sections 16.3.12 (PPR), 16.3.16 (SDTR), and 16.3.18 (WDTR) need to be redirected to the new section numbers (probably always safe since they refer to 4.x) or the 4.x model section (preferable if a specific field is being referenced).

All references to the various field names need to be reviewed. (e.g., phrases like "if information units are enabled" may need to change to "if an information unit transfer agreement is in effect")

All references to "transfer agreement" need to be reviewed. (e.g. make sure terminology is consistent for phrases like "paced data transfer agreement" vs. "paced transfer agreement", and that we have all the "transfer agreements" used in the main text defined in this section.)