2eSST
(SOURCE SYNCHRONOUS TRANSFER)
DRAFT STANDARD

VITA 1.5--199x

Draft 1.8
June 7, 1999
Printed on June 7, 1999 5:09 pm

This draft standard is being prepared by the VITA Standards Organization (VSO) and is unapproved.

Do not specify or claim conformance to this draft standard.

VSO is the Public Domain Administrator of this draft standard and guards the contents from change except by sanctioned meetings of the task group under due process.

VITA Standards Organization
7825 East Gelding Drive, Suite 104
Scottsdale, AZ 85260
Phone: (480) 951-8866
Fax: (480) 951-0720
URL: http://www.vita.com

Table of Contents

Do not specify or claim conformance to this draft standard

List of Figures

List of Tables

Abstract

This draft standard is an extension to the VME64 (ANSI/VITA 1-1994) and VME64x (VITA1.1-1997) standards. It defines a new transfer protocol, based upon source synchronous concepts, that permits the VMEbus to operate at rates up to 320MB/s. As technology improves, this rate may be extend to higher levels.

Foreword

Since the VMEbus was developed in the early 80's, it has been continuously enhanced. New features have been added and new technologies have been adopted to increase the utility and performance. This standard adds a new protocol to further enhance the performance of the VMEbus.

Comments, Corrections, Additions

This document was started by Drew Berding, Arizona Digital, and Jonathan Morris, Tundra Semiconductor, and is currently being edited by Mac Rush, Motorola. At the present time, all comments, corrections or additions should be addressed to:

Mac Rush
Document Editor
Motorola Computer Group
2900 S. Diablo Way
M.D. DW278
Tempe, AZ 85282
Phone: 602-438-3019
Fax: 602-438-3836
Email: rush@phx.mcd.mot.com

Draft History

Draft Number	Date	Changes/Comments
D1.0	June 6, 1997	First draft
D1.1	June 30, 1997	Total revision of Chapter 5; Remove Trademarks; addition of section 3.1.3
D1.2	June 30,1997	Re-organized document to mimic other VSO documents and expanded text on protocol.
D1.3	January 12, 1998	Add comments from the July VSO meeting and additional text.
D1.4	March 17, 1998	Add comments from January VSO meeting. Converted to FrameMaker. Added section on broadcast and broadcall.

Draft Number	Date	Changes/Comments
D1.5	April 20, 1998	Redefined the fields in address phase two and three. Increased the size of the beat count field and changed the name to cycle count. Removed broadcall. Added a bit to indicate broadcast transfers and added slave select bits for broadcast transfers. Transfer rate is now defined by a transfer rate field rather than XAM codes. Simplified protocol by allowing only the source to terminate the cycle.
D1.6	July 1, 1998	Added comments from the first task group ballot and May VSO meeting. Re-numbered rules, observations.
D1.7	November 3, 1998	Added comments from the July and September VSO meetings.
D1.8	June 7, 1999	Added comments from the second task group ballot, January VSO meeting, March VSO meeting and May VSO meeting.

Task Group Members

The 2eSST Task Group consists of the following members:

Name	Position	Company
Mac Rush	Chairman	Motorola Computer Group
Chau Pham	Vice Chairman	Motorola Computer Group
Mac Rush	Document Editor	Motorola Computer Group
Drew Berding	Document Editor	Arizona Digital
Ed Barsotti	Member	Fermilab
Bob Downing	Member	VIPA
Denis Dubois	Member	Tundra Semiconductor
Wayne Fischer	Member	Force Computers
Joe George	Member	Nanotek
Jing Kwok	Member	DY 4 Systems
Jonathan Morris	Document Editor	Tundra Semiconductor
Chris Parkman	Member	CERN
John Rynearson	Member	VITA
Robert Sullivan	Member	Hybricon
Istvan Vadasz	Member	Force Computers

Unresolved Issues

The VMEbus drivers for the 2eSST protocol continue to be an issue. I see the following possible options.

> Specify standard TTL drivers which conform to the ANSI/VITA 1-1994 standard. This applies to the DC electrical characteristics (threshold and drive). The user is still required to chose a driver which meets the AC electrical requirements such as skew and delay. Standard TTL devices are not specified very well. The switching threshold is poorly defined (.8v to 2v). Part to part and driver to driver skew is generally not specified. The poorly defined threshold will contribute to skew especially if the edge rates are slow. The wide threshold also makes the devices sensitive to the ringing that occurs when a backplane is driven. A 10ns pulse used to be a glitch in the early days of VME and the receivers would filter it out. Today a 10ns pulse is the period of the processor bus clock and the devices will pass it just fine. The edge rates are not defined and are probably getting fast enough to start causing excessive crosstalk (FCT has been there for a long time). See ETL below. There are a large number of device configurations and families to chose from. 5v and 3.3v parts are available.
>
> Specify ETL devices. There are a very limited number of devices available and these devices do not include registers which can be useful in reducing skew. The threshold is tightly defined (1.5v 100mv) and skew is specified. They are 5v devices which may be a problem when used with submicron ASIC devices which are not 5v tolerant. They have fast edge rates which can cause excessive crosstalk and may require (according to simulations) an RL filter to slow the edge rate and reduce crosstalk to an acceptable level.
>
> Specify the characteristics which would be ideal for driving a VMEbus backplane. This driver most likely would not exist. We can’t build boards with a driver that does not exist. This would be the best solution except for that small problem. The VITA 2.1 task group is working with IC vendors to define new drivers and this may be a solution in the future.
>
> Specify ETL or V320 drivers. The V320 drivers include registers and have better specs than standard TTL drivers but they are not as good as ETL. Packaging wise, the V320 drivers not very suitable for the control signals so a mix of ETL and V320 may work out OK. The V320 drivers are also 5v parts.

The document currently specifies ETL or V320 drives. This was the consensus (I believe) at the May VSO meeting. Here is an alternative for Rule 1.

Rule 3.1:

2eSST boards **SHALL** use bus transceivers as defined in the ANSI/VITA 1-1994 standard.

Observation 3.1:

The threshold specification and skew on standard TTL transceivers is not tightly specified. The wide threshold region may introduce additional skew and this skew must be accounted for in the system design. Additional skew may reduce the maximum transfer rate.

Recommendation 3.1:

Consider using the VITA 2-199x Enhanced Transceiver Logic (ETL) bus transceivers or the V320 registered bus transceivers. These devices have improved specifications which will reduce skew and they are designed and specified for driving backplane buses such as the VMEbus.

This adds one more variable to the “will it work” equation.

Chapter 1

Introduction to the 2eSST Standard

1.1 VME Protocol Background

There have been several types of cycles defined throughout the history of the VMEbus standards:

SCT: Single Cycle Transfer is the traditional VMEbus read or write cycle. During a single cycle transfer, only a single data transfer of up to 32 data bits (up to four bytes) occurs. Single cycle transfers have a theoretical maximum transfer rate of 40 MB/s.

BLT: During a Block Transfer, the starting address is transferred followed by multiple data transfers. The master and slave increment their address counters with each data transfer. This allows faster operation than multiple single cycles for certain kinds of slaves (such as DRAM memory boards) since only the initial address has to be transferred. Block transfers also allow more efficient use of write post and read ahead buffers because the data is sequential. BLT transfers have a theoretical maximum transfer rate of 40 MB/s.

MBLT: Multiplexed BLock Transfer is defined in the VME64 standard. MBLT utilizes the 32 data lines plus the 31 address lines and LWORD* as a single 64-bit bus. This bus is time multiplexed between address and data, and after the address transfer, the entire bus is used for multiple data transfers. MBLT allows twice the transfer rate of BLT since it has twice the number of data bits. MBLT transfers have a theoretical maximum transfer rate of 80 MB/s.

2eVME: Two edge VME is defined by the VME64x standard. The 2eVME protocol utilizes both edges of the control signals to transfer data. Since both edges of the control signals are used, only one round trip bus delay is encountered instead of the two round trip bus delays that are encountered when the SCT, BLT and MBLT protocols are used. 2eVME transfers have a theoretical maximum transfer rate of 160 MB/s.

2eSST: Two edge Source Synchronous Transfer is the subject of this standard. As a source synchronous protocol, the performance of 2eSST is determined not by the propagation delay from source to destination, but by skew --- the variation in propagation delay through the drivers, backplane and receivers. As the skew decreases, the bandwidth of the system can increase. In theory, as technology improves, a source synchronous protocol is virtually unlimited in its potential transfer rate. This standard provides for a 21 slot backplane transfer rate of 320MB/s.

1.2 Objectives of the 2eSST Protocol

In developing the 2eSST protocol, several important objectives were considered:

Maximize Performance: Performance was the driving impetus for this new protocol. To meet this objective, the protocol was designed to ensure that all devices involved in the transfer would operate as fast as possible. For example, the protocol does not permit throttling, implying that the receiver of a transaction must be able to operate at the requested speed.

Minimize Complexity: The 2eSST protocol was designed to minimize the amount of logic that would be required to implement the protocol. This was made in light of the fact that most 2eSST interfaces would also support traditional VME protocols and the 2eVME protocol.

Minimize Application Limits: The 2eSST protocol can be used in 6U and 9U environments, with the traditional 96-pin connector, or the newer 160-pin connector defined in the VME64x standard. 3U boards, however, must use the 160-pin connector.

Maintain Compatibility: The 2eSST protocol and the previous 2eVME protocol and MBLT protocol were designed to be compatible with legacy VMEbus products. However, at the highest transfer rate defined by the 2eSST protocol, the DS1* and DTACK* signals will have pulse widths narrower than defined by the VME64 standard or VME64x standard. This is not a problem since legacy boards will not be participating in 2eSST transactions.

1.3 References

The following publications are used in conjunction with this standard. When they are superseded by an approved revision, that revision must apply.

Table 1.1: References

Document	Description
ANSI/VITA 1-1994	VME64 Standard, Approved April 10, 1995
ANSI/VITA 1.1 -1998	VME64x Standard
VITA 2-199x	Enhanced Transceiver Logic
IEC 61076-4-113	2.54 mm 160 pin connectors complementary to IEC 603-2 Style C connectors

1.4 Standard Terminology

To avoid confusion and to make very clear what the requirements for compliance are, many of the paragraphs in this standard are labeled with keywords that indicate the type of information they contain. The keywords are listed below:

Rule
Recommendation
Suggestion
Permission
Observation

These key words are used as follows:

Rule <chapter>.<number>:

Rules form the basic framework of this draft standard. They are sometimes expressed in text form and sometimes in the form of figures, tables or drawings. All rules shall be followed to ensure compatibility between board and backplane designs. All rules use the “**SHALL**” or “**SHALL NOT**” words to emphasize the importance of the rule. The upper-case “**SHALL**” or “**SHALL NOT**” words are reserved exclusively for stating rules in this standard and are not used for any other purpose.

Recommendation <chapter>.<number>:

Wherever a recommendation appears, designers would be wise to take the advice given. Doing otherwise might result in some awkward problems or poor performance. While the 2eSST architecture has been designed to support high-performance systems, it is possible to design a system that complies with all the rules but has poor performance. In many cases a designer needs a certain level of experience in order to design boards that deliver top performance. Recommendations found in this standard are based on this kind of experience and are provided to designers to speed their traversal of the learning curve. All recommendations use the “**SHOULD**” or “**SHOULD NOT**” words to emphasize the importance of the recommendation. The upper-case “**SHOULD**” or “**SHOULD NOT**” words are reserved exclusively for stating recommendations in this draft standard and are not used for any other purpose.

Suggestion <chapter>.<number>:

A suggestion contains advice that is helpful but not vital. The reader is encouraged to consider the advice before discarding it. Some design decisions that need to be made in designing boards are difficult until experience has been gained. Suggestions are included to help a designer who has not yet gained this experience. Some suggestions have to do with designing boards that can be easily reconfigured for compatibility with other boards, or with designing the board to make the job of system debugging easier.

Permission <chapter>.<number>:

In some cases a rule does not specifically prohibit a certain design approach, but the reader might be left wondering whether that approach might violate the spirit of the rule or whether it might lead to some subtle problem. Permissions reassure the reader that a certain approach is acceptable and will cause no problems. All permissions use the "**MAY**" words to emphasize the importance of the permission. The upper-case word "**MAY**" words are reserved exclusively for stating permissions in this draft standard and are not used for any other purpose.

Observation <chapter>.<number>:

Observations do not offer any specific advice. They usually follow naturally from what has just been discussed. They spell out the implications of certain rules and bring attention to things that might otherwise be overlooked. They also give the rationale behind certain rules so that the reader understands why the rule shall be followed.

1.5 Terminology

In this standard, hexadecimal numbers are preceded by a 0x prefix. For example, the hexadecimal number 20 would be represented as 0x20 in this standard.

In this standard, buses are represented by a bus name followed by a bit field. For example, a bus could be represented as D[31:0] where "D" is the bus name and "[31:0]" represents the bits 31 to 0 inclusive where bit 31 is the most significant bit and 0 is the least significant bit.

In this standard, active low signals are indicated by an "*" following the signal name.

Chapter 2

2eSST Protocol

2.1 Introduction

The 2eSST protocol was designed to enhance the performance of the VMEbus. The 2eSST protocol defines three new transfers rates for the VMEbus that are based on source synchronous transfer technology. The new transfer rates are 160 MB/s, 267 MB/s and 320 MB/s. The 2eSST protocol also provides for broadcast data transfers. A broadcast data transfer allows the master to transfer the same data to multiple slaves with a single transfer.

2.2 3U and 6U Implementations

The VMEbus standards define several different VME board sizes: 3U, 6U and 9U. The 2eSST protocol, like the previous VMEbus protocols, was designed to support the various board sizes. 3U boards only connect to the VMEbus through the P1 connector and the signals on the P2 connector are not available. 3U boards have a 32-bit data bus, a 32-bit or 40-bit address bus and they use the RESP* signal. Since the RESP* signal is required and it is located on the Z row of the P1 connector, 3U boards are required to use the 160-pin VME64x connectors. 3U boards use the 3U implementation of the 2eSST protocol.

6U and 9U VME boards generally connect to the VMEbus using the P1 and P2 connectors. Boards that connect to the VMEbus using the P1 and P2 connectors are considered 6U boards. Boards that connect to the VMEbus through only the P1 connector are considered 3U boards. 6U boards have a 64-bit data bus, a 32-bit or 64-bit address bus and use the RETRY* signal. Since the RETRY* signal is located on the B row of the P2 connector, 6U boards are not required to use the 160-pin VME64x connectors. 6U boards use the 6U implementation of the 2eSST protocol.

In this standard, the signal RETRY*/RESP* is used to indicate that boards using the 6U implementation will use the RETRY* signal and boards using the 3U implementing will use the RESP* signal. A board using the 6U implementation can communicate with a board using the 3U implementing and when it does, it must use the RESP* signal. If a board using the 6U implementing is to be compatible with both boards using the 3U and 6U implementations, it must be able to monitor and drive both RETRY* and RESP* and therefore, it must use the 160-pin VME64x connector. However, a board using the 6U implementation must not connect RESP* and RETRY* together and it must not drive both the RESP* and RETRY* signals at the same time.

2.3 Broadcast Transfers

The current VMEbus protocols allow a VMEbus master to write data to a specific VMEbus slave. If a master needs to transfer the same data to multiple slaves, the master would send the data to each slave individually. For example, to transfer a block of data to 20 slaves, the master would transfer the same block 20 times. The 2eSST protocol supports broadcast transfers which allow a master to write the same data to multiple slaves with a single transfer. Using the broadcast transfer capability of the 2eSTT protocol, the master can transfer a block of data to 20 slaves using a single transfer. Broadcast transfers support one slave per VMEbus slot which allows a maximum of 21 slaves in a 21 slot VMEbus backplane. During a broadcast transfer, all slaves are considered participating slaves rather that responding slaves. A participating slave receives the data but has limited interaction with the master.

2.4 Remapping the LWORD* Line

In the VME64 standard, the 32-bit address bus and the 32-bit data bus are combined to form a 64-bit data bus to provide a path for the 64-bit data transfers. Since the VMEbus lacks an address bit 0, the LWORD* line is redefined as address bit 0, A[0]. This effectively provides a full 32-bit address bus, labeled A[31:0], and reflects the usage of the line. Therefore, during the data phase, the LWORD* signal is used as a data transfer signal line by the MBLT, 2eVME and 2eSST protocols.

2.5 Source Synchronous Transfer

The main difference between the 2eSST protocol and the 2eVME protocol is that during the 2eSST data phases, the protocol is source synchronous. No acknowledgment is expected from the receiver of the data. Hence, the theoretical performance of 2eSST is limited only by the skew between transmitter and receiver of data. Like the 2eVME protocol, the 2eSST protocol uses both edges of the strobe to transfer data. The result is a protocol that defines a transfer rate of 320 MB/s, which is double the theoretical transfer rate of 2eVME.

The protocol can be broken into three main phases: address phase, data phase and termination phase.

2.6 Address Phase

The address phase of the 2eSST protocol is similar to the 2eVME protocol. The 2eSST address phase is split into three phases, address phase one, address phase two and address phase three. Three times as much information can be transferred during a 2eSST address transfer as compared to a VME64 address transfer. Table 2.1 defines the signal fields during the address phase for 6U implementations. Table 2.2 defines the signal fields during the address phase for 3U implementations.

Table 2.1: 6U Address Phase Signal Field Definitions

<table>
<tr><th>Signal Line</th><th>Address Phase 1</th><th>Address Phase 2</th><th>Address Phase 3</th><th>Data Phase</th></tr>
<tr><td>AM[5:0]</td><td>0x20</td><td>0x20</td><td>0x20</td><td>0x20</td></tr>
<tr><td>A[7:0]</td><td>XAM Code
(Table 2.4)</td><td>A[3:0] = 0
Device Address
A[7:4]</td><td rowspan="3">For XAM = 11,12
A[23:0] = Reserved

For XAM = 21,22
A[0] = Reserved
A[21:1] = Broadcast
Slave Select
A[23:22] = Reserved</td><td>D[39:32]</td></tr>
<tr><td>A[15:8]</td><td>Device Address
A[15:8]</td><td>Cycle Count</td><td>D[47:40]</td></tr>
<tr><td>A[23:16]</td><td>Device Address
A[23:16]</td><td>A[23:21] = 0
A[20:16] = GA of Master</td><td>D[55:48]</td></tr>
<tr><td>A[31:24]</td><td>Device Address
A[31:24]</td><td>Subunit Number
in Master</td><td>Reserved</td><td>D[63:56]</td></tr>
<tr><td>D[31:0]</td><td>Device Address
A[63:32] (= 0 for A32)</td><td>D[3:0] = Transfer Rate
D[31:4] = Reserved</td><td>Reserved</td><td>D[31:0]</td></tr>
</table>

Table 2.2: 3U Address Phase Signal Field Definitions

<table>
<tr><th>Signal Line</th><th>Address Phase 1</th><th>Address Phase 2</th><th>Address Phase 3</th><th>Data Phase</th></tr>
<tr><td>AM[5:0]</td><td>0x21</td><td>0x21</td><td>0x21</td><td>0x21</td></tr>
<tr><td>A[7:0]</td><td>XAM Code
(Table 2.4)</td><td>A[3:0] = 0
Device Address A[7:4]</td><td rowspan="3">For XAM = 11,12
A[23:0] = Reserved

For XAM = 21,22
A[0] = Reserved
A[21:1] = Broadcast Slave Select
A[23:22] = Reserved</td><td>D[23:16]</td></tr>
<tr><td>A[15:8]</td><td>Device Address
A[15:8]</td><td>Cycle Count</td><td>D[31:24]</td></tr>
<tr><td>A[23:16]</td><td>Device Address
A[23:16]</td><td>A[23:21] = 0
A[20:16] = GA of Master</td><td>Reserved</td></tr>
<tr><td>D[7:0]</td><td>Device Address
A[31:24]</td><td>Subunit Number
in Master</td><td>Reserved</td><td>D[7:0]</td></tr>
<tr><td>D[15:8]</td><td>Device Address
A[39:32] (= 0 for A32)</td><td>D[11:8] = Transfer Rate
D[15:12] = Reserved</td><td>Reserved</td><td>D[15:8]</td></tr>
</table>

Normally the address phase is followed by a data phase. However, the slave can terminate the transfer during address phase one or three. There are two ways the slave can terminate the transfer.

During address phase one or three, the slave can respond with a slave error response by asserting BERR*. During address phase three, the slave can respond with a slave suspend response by asserting RETRY*/RESP* and DTACK*. The master can retry a transfer that is terminated with a slave suspend response.

2.6.1 Extended AM codes

Since there are only a few unassigned address modifier codes left, an extended address modifier (XAM) coding scheme is used. AM code 0x20 is assigned for enhanced data transfer protocols on 6U implementations and AM code 0x21 is assigned for enhanced data transfer protocols on 3U implementations. The eight LSB (Least Significant Bits) of the address field A[7:0] are used to carry the XAM code information during the first address phase. See Table 2.3, which maps the address lines to the XAM code field. With 8-bits, 256 additional address modifier codes are available for 6U implementations and 256 additional address modifier codes are available for 3U implementations.

Table 2.3: Extended Address Modifier Line Definitions

A7	A6	A5	A4	A3	A2	A1	A0
XAM7	XAM6	XAM5	XAM4	XAM3	XAM2	XAM1	XAM0

The 2eSST definition defines four XAM codes. The XAM code definitions are shown in Table 2.4. With 256 possible XAM codes for each board size, the extended address addition leaves plenty of room for future expansion.

Table 2.4: Extended Address Modifier Code

Address Modifier Code Implementation	Extended Address Modifier Code	Address/Data Mode
0x20 6U	0x11	A32/D64 2eSST
	0x12	A64/D64 2eSST
	0x21	A32/D64, Broadcast 2eSST
	0X22	A64/D64, Broadcast 2eSST
0x21 3U	0x11	A32/D32 2eSST
	0x12	A40/D32 2eSST
	0x21	A32/D32, Broadcast 2eSST
	0x22	A40/D32, Broadcast 2eSST

2.6.2 Address Modes

In the 2eSST standard, the XAM codes are used to define the address mode. For 6U implementations, one address range is defined for A32 type transactions and another for A64 type transactions. During an A32 transaction, the address is provided on signal lines A[31:8] during address phase one and A[7:4] during address phase two. During an A64 transaction, the address is provided on signal lines D[31:0] and A[31:8] during address phase one and A[7:4] during address phase two.

For 3U implementations, two address ranges are defined: A32 and A40. During an A32 transaction, the address is provided on signal lines D[7:0] and A[23:8] during address phase one and A[7:4] during address phase two. During an A40 transaction, the address is provided on signal lines D[15:0] and A[23:8] during address phase one and A[7:4] during address phase two.

Usage of the supervisory/non-privileged and the program/data sub-modes are no longer necessary, with the very large A40 and A64 address ranges. These large address ranges will most likely provide enough space to map the memory into special functional groups, if that is required by the specific application.

2.6.3 Transfer Rates

In the 2eSST standard, the maximum transfer rate is specified on data lines D[3:0] for a 6U implementation and data lines D[11:8] for a 3U implementation. The transfer rate is specified during address phase two (refer to Table 2.1 and Table 2.2). Boards using the 6U implementation transfer 64-bits (eight bytes) per transfer and transfer rates of 160, 267 and 320 MB/s are defined. Boards using the 3U implementation transfer 32-bits (four bytes) per transfer and transfer rates of 80, 133 and 160 MB/s are defined. The minimum cycle time is the same for 3U and 6U implementations but the maximum data transfer rate for a 3U implementation is half the maximum data transfer rate for a 6U implementation. This is because the data bus width in a 3U implementation is half width of a data bus in a 6U implementation. The 2eSST transfer rates for a 6U implementation are defined in Table 2.5. The 2eSST transfer rates for a 3U implementation are defined in Table 2.6

.

Table 2.5: 6U 2eSST Transfer Rates

Transfer Rate Code	6U Transfer Rate (MB/s)	Nominal Strobe Width (ns)	Maximum Strobe Frequency (MHz)	Minimum Strobe Period (ns)	Nominal Strobe Duty Cycle (%)	Mnemonic
0x0	160	50	10	100	50	SST160
0x1	267	30	16.67	60	50	SST267
0x2	320	25	20	50	50	SST320
0x3-0xF	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

Table 2.6: 3U 2eSST Transfer Rates

Transfer Rate Code	3U Transfer Rate (MB/s)	Nominal Strobe Width (ns)	Maximum Strobe Frequency (MHz)	Minimum Strobe Period (ns)	Nominal Strobe Duty Cycle (%)	Mnemonic
0x0	80	50	10	100	50	3USST80
0x1	133	30	16.67	60	50	3USST133
0x2	160	25	20	50	50	3USST160
0x3-0xF	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

During address phase three, the slave can terminate the transfer with a slave error if it is unable to perform the requested transfer at the specified transfer rate.

The transfer rate specified is defined as a maximum transfer rate. The transmitter can transmit at a lower rate than specified. The transmitter can also throttle its transfer by momentarily reducing the transfer rate or stopping the transfer. However, transmitting below the specified transfer rate defeats the purpose of a high transfer rate protocol and should be avoided.

2.6.4 Known Length 2eSST Transfers

With many of the new microprocessors that employ caching architectures, as well as DMA controllers, the size of the data being written or requested is known in advance. The cycle count is presented on address lines A[15:8] during address phase two (refer to Table 2.1 and Table 2.2). This effectively informs the slave in advance of the amount of data that it is requested to receive in a write transaction or the amount of data it is to supply in a read request. The cycle count value sent is the beat count divided by two. There are two data beats in each cycle. For example, a cycle count of 0x00 = 0 data beats, a cycle count of 0x01 = 2 data beats, and a cycle count of 0x80 = 256 data beats.

Boards using the 6U implementation perform data transfers with 64-bit data words (eight bytes per transfer). The beat count represents the number of eight byte transfers. The byte count is the equal to the beat count times eight. The maximum cycle count is 128. For 6U implementations, this limits the transfer size to 2048 bytes (128 cycles * 2 data beats per cycle * 8 bytes per data beat).

Boards using the 3U implementation perform data transfers with 32-bit data words (four bytes per transfer).

The beat count represents the number of four byte transfers. The byte count is the equal to the beat count times four. The maximum cycle count is 128. For 3U implementations, this limits the transfer size to 1024 bytes (128 * 2 data beats per cycle * 4 bytes per data beat).

During write transfers, the master can transfer any even number of beats up to but not exceeding the beat count. During read transfers, the slave can transfer any even number of beats up to but not exceeding the beat count. The transmitter must not cross a 2K address boundary.

During address phase three, the slave can suspend the transfer if it is unable to perform the requested transfer at the present time. For example, if the slave's buffer is still full from a previous transfer, it can suspend the transfer and the master will retry the transfer at a later time. 6U slaves are required to support transfers up to 2048 bytes and 3U slaves are required to support transfers up to 1024 bytes.

2.6.5 Geographic Address

During address phase two, address lines A[20:16] are used to carry the Geographic Address (GA) of the board (refer to Table 2.1 and Table 2.2). If this feature is not implemented, the value must be 0. The Geographic Address is defined in Chapter 3 of the VME64x standard.

2.6.6 Subunit Number

During address phase two, address lines A[31:24] for 6U implementations and data lines D[7:0] for 3U implementations are used to carry the subunit number of the master (refer to Table 2.1 and Table 2.2). If this feature is not implemented, the value must be 0. If a VME board has multiple masters which perform 2eSST transfers, each master can be assigned a different subunit number.

2.6.7 Broadcast Transfer Address Phase

Broadcast transfers are indicated by the XAM codes as defined in Table 2.4. There are several differences between a regular address phase and a broadcast transfer address phase. During a regular address phase, the selected slave (responding slave) responds by asserting the DTACK* signal. However, during a broadcast transfer there may be multiple selected slaves and all of them cannot respond to the transfer. The selected slaves, during a broadcast transfer, do not respond to the master and they are considered participating slaves. The master generates the DTACK* signal during the address phase of a broadcast transfer. The address phase of a broadcast transfer has the same timing diagram as a regular address phase. However, since the participating slaves do not respond to the master, they are not able to control the timing of the address phase. Therefore, the minimum time from DS0* to DTACK* has been increased to allow the participating slaves sufficient time to track the address phase. The participating slaves must be designed to work within this time.

The participating slaves can request a retry of the transfer by asserting the RETRY*/RESP* signal during address phase three. During the address phase three, the participating slaves are allowed to assert the BERR* signal to indicate an error condition. Since the RETRY*/RESP* and BERR* signals from the participating slaves are not interlocked with the master, the participating slaves must drive the RETRY*/RESP* or BERR* signals within the specified time or the master may not receive the signal.

There are two levels of decoding associated with broadcast transfers. The broadcast slaves must decode the VMEbus address and address modifier codes like regular slaves. However, in the case of broadcast slaves, there may be more than one addressed slave. Address bits A[21:1] during address phase three carry the slave select signals that define which slaves are selected and will receive the broadcast data. Each of the 21 possible slaves is assigned a unique select bit. In systems which support geographic addressing, the geographic address is used to define the slaves unique select bit. When the geographic address is used, GA=1 would correspond with address bit A[1], GA=2 would correspond with address bit A[2] and so on. Therefore, the slave in slot 1 (GA=1) would respond to a broadcast transfer if address bit A[1] is asserted during address phase 3. This draft standard does not specify how the slave address is defined in systems which do not support geographic addressing,

2.7 Data Phase

The protocols defined in the VME, VME64 and VME64x standards utilize a handshake protocol whereby data strobes (DS1* and DS0*) are acknowledged by DTACK*. The receipt of DTACK* by the master causes the data strobes to be removed which in turn causes DTACK* to be removed. Once DTACK* is deasserted, a new cycle can begin. The protocols defined in the VME and VME64 standards require four delays through the drivers, backplane and receivers plus the settling time of the backplane. The 2eVME protocol defined in the VME64x standard improves upon this by using both edges of DS1* and DTACK* to qualify the data. This doubles the throughput, but performance is limited by the requirement for an acknowledgment from the receiver of the data.

The 2eSST protocol has an address phase which is similar to the 2eVME address phase. After the address phase, however, the 2eSST transmitter sends the data along with a delayed strobe (either DS1* or DTACK*). The transmitter does not wait for any acknowledgments from the receiver and both edges of the strobe are used. Therefore, data can be streamed at much higher rates. For write transfers (where the master is the transmitter of data), DS1* is used to qualify the data. For read transfers (where the slave is the transmitter of data), DTACK* is used to qualify the data. DS0* is not used as a strobe during the 2eVME or 2eSST data phase.

A consequence of the source synchronous nature of the 2eSST protocol is that the receiver of the data no longer has the ability to throttle the data transfer. In traditional VME block transfer reads, the master (the receiver of data) could throttle the transfer by controlling its assertion/deassertion of DS0*/DS1*. The slave (the receiver of data) could throttle writes by controlling DTACK*. In the 2eSST protocol, only the transmitter of data can throttle the transfer. This implies that the 2eSST receiver (master during reads, slave during writes) must be able to accept the data at the rate it is sent.

2.7.1 Broadcast Transfer Data Phase

During the data phase of a broadcast transfer the master transmits the data and the participating slaves receive the data. During the data phase, participating slaves are allowed to assert BERR* signal to indicate an error condition.

2.8 Transfer Termination

2eSST transfers are terminated by the transmitter. The receiver does not have the ability to terminate the transfer. Termination during error conditions is defined in the next section.

Transfers always terminate on an even beat. However, during a read operation the transmitter can indicate that the data in the last even data beat of the transfer is invalid. The invalid data indication accommodates slaves which are transferring data generated by external sources and do not have an even number of data words to transmit.

Terminating on an even beat helps achieve two of the objectives of the 2eSST protocol; it minimizes design complexity while maximizing throughput. By not allowing termination at odd beats, 2eSST state machines need not be designed for the special case where the transfer stops with the strobes in the asserted position necessitating a “clean-up” dummy cycle to return the strobes to the negated state. In addition, the number of test conditions a design must be verified under is reduced. By removing the potential for a dummy cycle, and by removing the requirement to monitor for terminations on every cycle, the state machines will also be able to run faster.

During a write transfer, the master controls the 2eSST address phase and the data phase. During the write data phase, the master sends data to the slave using DS1* as a strobe to qualify the data. The master terminates a write transfer by stopping the data transmission and then negating DS0*. The slave will respond by negating DTACK*. The master will always terminate on an even beat count. The 2eSST protocol allows the master to terminate the write transfer at any time, even before the beat count is reached. This can be used for a variety of special type applications and situations.

During a read transfer, the master controls the 2eSST address phase and the slave controls the data phase. During the data phase, the slave is in control and sends data to the master using DTACK* as a strobe to qualify the data. The slave terminates a read transfer by stopping the data transmission and then asserting RETRY*/RESP* and BERR*. The master will respond by negating DS0*. The slave will always terminate on an even beat count. The 2eSST protocol allows the slave to terminate the read transfer at any time, even before the beat count is reached. This can be used for a variety of special type applications and situations.

During a read transfer, the slave can indicate that the last data beat is not valid. The slave indicates the last data beat is not valid by asserting RETRY*/RESP* during the last data beat. The 2eSST standard does not define how the master handles the invalid data. The master can pass the invalid data to memory and set a status bit that indicates the last data beat is not valid.

2.9 Slave Error Termination

As with other protocols, a slave error termination is also included with 2eSST. This permits the slave to relay some error condition back to the master. Identification of the particular error condition, and actions taken is not covered in this standard. During a read transfer a slave indicates an error termination by stopping the transfer, driving RETRY*/RESP* high and asserting BERR*. During a write transfer, a slave indicates an error termination by asserting BERR*.

During broadcast transfers, the BERR* signal from the participating slaves is not interlocked with the master. Therefore, the participating slaves must drive the BERR* signal within the specified time or the master may not receive the signal. A minimum time is specified between when the master negates DS0* and when it negates DTACK*. This provides a defined time period, after the last data beat is sent, in which the participating slave can assert a BERR* a be sure the master will receive it.

2.10 Bus Time Out Timer

The 2eBTO timer defined for the 2eVME protocol also supports the 2eSST protocol and is the preferred timer. If the VMEbus BTO timer is used, the following items must be considered. During a 2eSST transfer, DS0* is held low during the entire data transfer cycle. The transfer size of the 2eSST transfer must be selected to allow the transfer to complete before the timer expires.

Chapter 3

Requirements

This chapter defines the requirements for the 2eSST protocol in terms of rules, recommendations, suggestions, observations, and permissions.

3.1 Transceivers, Connectors and Backplanes

Rule 3.1:

2eSST boards **SHALL** use VITA 2-199x Enhanced Transceiver Logic (ETL) bus transceivers with the resistor inductor network defined in VITA 2-199x or V320 transceivers or electrically equivalent transceivers on all address, data and control bus signal lines.

Rule 3.2:

System integrators using the 2eSST protocol **SHALL** ensure that the proper VME64 or VME64x backplane and boards loaded into the system will allow for monotonic rising and falling bus signals through the threshold region of the receiver.

Observation 3.1

The bus signals can have non-monotonic rising and falling edges at the driver. A board that monitors the signals that it drives must be prepared for non-monotonic edges.

Rule 3.3:

Boards using the 2eSST protocol **SHALL** support the RETRY*/RESP* line as defined in this draft standard.

Rule 3.4:

The RESP* signal is in the Z row of the 160-pin connector, therefore the 160-pin connector **SHALL** be used on boards that use the 3U implementation of the 2eSST protocol. Boards that use 6U implementation of the 2eSST protocol and communicate with boards using the 3U implementation **SHALL** use the 160-pin connector. Boards using the 6U implementation are not required to support the 3U implementation.

Observation 3.2:

Boards using the 6U implementation of the 2eSST protocol can use either the 96-pin or the 160-pin connectors. The 160-pin connector provides additional ground pins that will improve the signal quality which is important to boards running high speed protocols such as 2eSST.

3.2 Address Phase Protocol

The 2eSST protocol uses the three phase address transfer developed for 2eVME. This three phase address transfer is indicated by AM code 0x20 (for 6U implementations) and 0x21 (for 3U implementations). Within the first phase of this address transfer is a field, the XAM (Extended AM) code that indicates what protocol is to be used once the address transfer completes. The VME64x standard defines several of the XAM codes for 2eVME. This standard defines four additional XAM codes for the 2eSST protocol which are defined in Table 2.4.

Rule 3.5:

A master using the 6U implementation **SHALL** transfer the information defined in Table 2.1 on the address and data lines during the address transfer phase. A master using the 3U implementation

SHALL transfer the information defined in Table 2.2 on the address and data lines during the address transfer phase.

Rule 3.6:

All address and data lines **SHALL** be at logical zero for features which are not implemented or are part of reserved fields.

Rule 3.7:

Starting addresses **SHALL** be aligned on 16-byte boundaries (i.e. 128-bit aligned).

Rule 3.8:

2eSST address timing **SHALL** be as defined in Table 3.1 and Table 3.2, as measured at the connector on the backplane.

Rule 3.9:

2eSST address transfer phase **SHALL** be as defined in Figure 3.1 through Figure 3.4.

Observation 3.3

The address phase transfer and timing for the 2eSST protocol is the same as in the 2eVME protocol. The address phase of a 2eSST broadcast is the same as a non-broadcast 2eSST address phase except the slaves do not respond. The interpretation of some of the data fields presented during the 2eSST address phase is different than those presented during the 2eVME address phase.

Rule 3.10:

BERR* **SHALL** only be asserted on the first or third address phase instead of toggling DTACK*. See Figure 3.3 and Figure 3.4.

Rule 3.11:

DTACK* **SHALL** be the only valid response during address phase two.

Rule 3.12:

Masters **SHALL** ignore RETRY*/RESP* during address phase one and two, and BERR* during address phase two.

Observation 3.4:

Slave error and slave suspend operations are not supported during address phase two.

Rule 3.13:

RETRY*/RESP* **SHALL** only be asserted during address phase three.

Observation 3.5:

RETRY*/RESP* is not asserted in the first and second address phases.

Observation 3.6:

If RETRY*/RESP* is asserted before DTACK* is toggled on the third address phase, the master interprets that response as indicating that the slave is suspending the operation but the master can expect data when it tries again. This can also be interpreted as a slave suspend response. See Figure 3.2.

Observation 3.7:

Since the target address is in two parts, the target device generates DTACK* during the first phase of the address transfer if that portion of the address is recognized. The lowest byte of the address comes in the second address phase and therefore cannot be used as part of the address decode. The low byte only specifies an internal address.

Observation 3.8:

The Geographical Address (GA) in the second address phase provides a mechanism to indicate which board is the bus master for a particular transaction.

Observation 3.9:

The subunit number, transferred in the second address phase, provides a mechanism for identifying which part of a master initiated a transaction. In the case of a VME module with several processors, it could be the processor number.

Rule 3.14:

2eSST masters and slaves **SHALL** use the AM and XAM codes as defined in Table 2.4

Observation 3.10:

Address modifier codes 0x20 and 0x21 have been defined in the VME64x standard to specify a three-phase address transfer. These AM codes are shared with the 2eVME protocol, and are expected to be shared with other protocols as they are developed over time.

Rule 3.15:

If a targeted 2eSST slave is unable to operate at the transfer rate defined by the transfer rate code, it **SHALL** terminate the transfer with a slave error during phase three of the address transfer.

Recommendation 3.1:

2eSST masters and slaves **SHOULD** also support 2eVME transfers.

Observation 3.11:

Recommendation 3.1 ensures that a 2eSST capable board will be able to drop back to the slower 2eVME transfer rates if the board is placed in a system that cannot support 2eSST transfers.

Rule 3.16:

6U 2eSST masters supporting SST320 **SHALL** also support SST267 and SST160. 3U 2eSST masters supporting 3USST160 **SHALL** also support 3USST133 and 3USST80.

Rule 3.17:

6U 2eSST masters supporting SST267 **SHALL** also support SST160. 3U 2eSST masters supporting 3USST133 **SHALL** also support 3USST80.

Observation 3.12:

The 2eSST standard does not define a minimum transfer rate. Therefore, a master may claim to support a transfer rate but actually transmit at a lower rate. For example, a master may claim support for the SST267 rate but actually transmit at SST160.

Permission 3.1:

6U 2eSST slaves **MAY** support any combination of SST320, SST267 and SST160. 3U 2eSST slaves **MAY** support any combination of 3USST160, 3USST133 and 3USST80.

Recommendation 3.2:

The mnemonics defined in Table 2.5 and Table 2.6 **SHOULD** be used to indicate a product's support for that level of 2eSST performance.

Permission 3.2:

Boards using the 6U implementation **MAY** also participate in 3U 2eSST transfers using the applicable AM and XAM Codes defined for 3U 2eSST transfers.

Rule 3.18:

The cycle count in the second address phase specifies the maximum number of cycles that **SHALL** occur during the transaction. The cycle count is the number of data beats divided by two. For example, a cycle count of 0x00 = 0 data beats, a cycle count of 0x01 = 2 data beats, and a cycle count of 0x80 = 256 data beats. The maximum cycle count is 128 cycles. Slaves are **SHALL** support a cycle count of 128 cycles.

Permission 3.3:

The cycle count **MAY** be zero which indicates that no data will be transferred.

Observation 3.13:

Since the transmitter can only transfer an even number of beats, the modulo 2 number for the beat count is compatible with the specified operation.

Observation 3.14:

The cycle count represents a maximum value only. During 2eSST transfers, the receiver must be prepared for an early termination.

Observation 3.15:

The cycle count is provided to allow slaves to optimize their performance. During reads, the cycle count tells the slave how much data to transmit. During writes, the slave can use this field to determine if sufficient room is available in its write posting FIFOs, and suspend the transaction if insufficient room is available. During a write transfer, it is not advisable for the master to indicate a cycle count that is larger than the intended transfer. It is possible the slave will needlessly suspend the transfer because it believes it does not have room in its buffer.

3.3 Data Phases

The data phase of 2eSST transfer presents the data on D[31:0] and A[31:0], (D[15:0] and A[15:0] for 3U implementations) in the same manner as MBLT and 2eVME transfers. LWORD* has been defined as A[0] which is the same as in MBLT and 2eVME. During write transactions, the data is transmitted from master to slave. During read transactions, the data is transmitted from slave to master. Regardless of which device is transmitting the data, the data is sent in the same manner: data qualified with a strobe, followed by the next data qualified with the opposite edge of the same strobe. When the master is transmitting the data, DS1* is used as the qualifying strobe. When the slave is transmitting the data, it uses DTACK* to qualify the data.

The transmitter of data does not await acknowledgment of the data beat from the receiver. Instead it continuously transmits the data at a rate indicated by the transfer rate code.

Rule 3.19:

2eSST data transfer phase **SHALL** be as defined in Figure 3.5 through Figure 3.9.

Rule 3.20:

The transmitter **SHALL** transmit the data with setup and hold times to the strobe as defined in Table 3.1 and with the strobe timing defined in Table 3.3.

Observation 3.16:

The 2eSST protocols operate with strobes that switch in the center of the data window halfway between the permitted occurrence of data transitions. This has several benefits:

1. The strobe (which is the single most important timing signal qualifying the data) switches when the data lines are quiet and there is minimal cross talk and ground bounce to affect the strobe timing.

2. The source skew of the strobe is minimized since the delay between data and strobe can be done very precisely utilizing the opposite phase of the transmit clock or a 2x clock. If the opposite phase of the transmit clock is used, the duty cycle must be controlled.

3. Because the skew in a system is symmetric (data can skew either side of the strobe), the data will be guaranteed to be valid when the strobe reaches the receiver. The strobe then can be used to qualify the data with minimal logic.

4. Since the strobe switches in the center of the data window (at the transmitter), the slower data rates have improved setup and hold times.

Both edges of the strobe are used to qualify the data. To ensure that the strobes end up in the inactive state and to simplify state machine design, an even number of data beats will always be transmitted.

Observation 3.17:

The 2eSST design specifies that the slowest data has been presented to the receiving registers by one set-up time before the fastest strobe can arrive to clock the registers. It also specifies that the slowest strobe has already clocked the receiving registers by one hold time prior to the fastest data changing at the registers.

Rule 3.21:

The transmitter **SHALL NOT** attempt to transfer more data during the data phase than is specified by the cycle count.

Permission 3.4:

The transmitter **MAY** transfer less data during the data phase than the cycle count indicates.

Rule 3.22:

During 2eSST transfers the transmitter **SHALL NOT** cross any 2048 byte boundary.

Observation 3.18:

During the address phase, the master may transmit a starting address and cycle count that indicates a transfer could cross a 2048 byte boundary. During a write transfer, the transmitter (master) must terminate the transfer and not cross a 2048 byte boundary. During a read transfer, the transmitter (slave) must terminate the transfer and not cross a 2048 byte boundary.

Rule 3.23:

The transmitter of data, during the data phase, **SHALL NOT** transmit data faster than as specified by the transfer rate code at the start of the transfer.

Permission 3.5:

The transmitter is not required to transmit at the specified rate. The transmitter **MAY** transmit at a transfer rate less than the specified transfer rate. The transmitter **MAY** also throttle its transfer by momentarily reducing the transfer rate or stopping the transfer. When throttling, the transmitter is required to meet the specified setup and hold times, however, the strobe period and duty cycle **MAY** vary.

Rule 3.24:

The receiver of data, during the data phase, **SHALL** be prepared to receive the entire data transfer at the transfer rate as specified by the transfer rate code at the start of the transfer.

Observation 3.19:

Because the transmitter has control of the strobe, it has the capability to effectively throttle the transfer as its needs dictate. The receiver, however, has no such throttling control and must be prepared to accept the entire data transfer at the rate specified during address phase two.

Rule 3.25:

The transmitter, during the data phase, **SHALL** transfer an even number of data beats.

Observation 3.20:

While a 2eSST transfer always consists of an even number of data beats, the even data word of the last data beat of a read transfer **MAY** be marked invalid by the slave.

3.4 Transfer Termination

In the 2eSST protocol, the transmitter of the data normally terminates the transfer. The slave can terminate the transfer with an error condition.

Rule 3.26:

During a write cycle, the master **SHALL** terminate the transfer by ending the data transfer on an even data beat and then negating DS0*.

Rule 3.27:

During a read transfer, the slave **SHALL** terminate the transfer by ending the data transfer on an even data beat and then asserting RETRY*/RESP* and BERR*. During a read transfer, the slave can indicate that the last even data word is not valid by asserting RETRY*/RESP* during the last data phase.

3.5 Slave Error Termination

During a 2eSST read or write transfer, a slave can indicate an error termination.

Rule 3.28:

During a write transfer, the slave **SHALL** assert the BERR* signal to indicate a slave error termination.

Permission 3.6:

The slave **MAY** assert BERR* at any time during the write data transfer.

Permission 3.7:

If a master receives a BERR* during a write cycle, the master **MAY** terminate the transfer at an even beat boundary or the master **MAY** continue to transmit data until the cycle count has expired.

Rule 3.29:

During a read transfer, the slave **SHALL** indicate an error termination by negating RETRY*/RESP* prior to asserting BERR*.

3.6 Support for Broadcast Transfers

The 2eSST protocol provides support for broadcast transfers. Broadcast transfers allow the master to transmit data to multiple slaves. During non-broadcast transfers, the slave drives DTACK* during the address and data phases and is considered a responding slave. During a broadcast transfer, the slaves do not respond to the master by asserting DTACK* and are considered participating slaves. The master generates DTACK* during the address and data phases of a broadcast transfer.

During the data phase of a broadcast transfer the master transmits the data and the participating slaves receive the data. A participating slave receives the data but has limited interaction with the master. A participating slave can retry the transfer during phase three of the address phase by asserting RETRY*/RESP*. A participating slave can assert BERR* during address phase three to indicate an error condition. A participating slave can also assert BERR* during the data phase to indicate an error condition.

Rule 3.30:

The XAM code during address phase one **SHALL** be used to indicate a broadcast transfer. Refer to Table 2.1, Table 2.2 and Table 2.4

Rule 3.31:

Address lines A[21:1] during address phase three **SHALL** be used to select the participating slaves.

Rule 3.32

If the geographic address is used to define the slave selects, then GA[1] **SHALL** correspond to A[1], GA[2] **SHALL** correspond to A[2] and so on.

Rule 3.33:

The master **SHALL** assert DTACK* during the address phase and data phase.

Rule 3.34:

The participating slaves **SHALL NOT** assert DTACK* during the address phase or data phase.

Rule 3.35:

Participating slaves **SHALL** be designed to operate with the defined timing since they are not able to delay the acknowledgment.

Observation 3.21:

Since the participating slaves are not able to delay the acknowledgment, the timing parameter T2 has been increased to allow the participating slaves time to receive the information provided during the address phase.

Rule 3.36:

During phase three of the address transfer, participating slaves that are not able to receive data at the current time **SHALL** suspend the transfer by issuing a slave suspend response. The master will then retry the transfer at a later time. The participating slave issues a slave suspend response by asserting only the RETRY*/RESP* signal as shown in Figure 3.2. The master **SHALL** assert DTACK*. Slaves that cannot receive data at the specified rate **SHALL** terminate the transfer with an error termination by asserting BERR*.

The RETRY*/RESP* drivers are specified as three-state in the VME64 standard. Participating slaves that drive the RETRY*/RESP* signal to indicate a slave suspend response **SHALL** only drive the RETRY*/RESP* signal low. Participating slaves **SHALL NOT** drive the RETRY*/RESP* signal high.

Observation 3.22:

Like responding slaves, participating slaves cannot terminate or suspend the data phase of the transfer. They are required to receive the entire data transfer at the specified transfer rate. The cycle count and transfer rate are indicated during address phase 2.

Permission 3.8:

Like responding slaves, participating slaves **MAY** assert BERR* during the data phase to indicate an error condition.

3.7 Bus Time Out Timer

Observation 3.23:

Some bus timers are designed to assert BERR* when DS0* or DS1* has been asserted for greater than a set period, without monitoring the state of DTACK*. In such a case, the bus timer could time-out a 2eSST transfer since DS0* is held low for the duration of the transfer.

Recommendation 3.3:

It is recommended that the 2eBTO(x) bus time out timer **SHOULD** be used in conjunction with the 2eSST protocol. If a VMEbus BTO timer is used, the bus timer value **SHOULD** be set to a value greater than the longest expected 2eSST transfer.

3.8 Timing Diagrams

This section includes the 2eSST timing diagrams and timing parameters. Source timing parameters are measured on the backplane at the source connector. Destination timing parameters are measured on the backplane at the destination connector.

Table 3.1: Unique 2eSST Timing Parameters

Timing Parameter	Description	@ Source (ns)	@ Destination (ns)
S1	Address setup time	-5 min	-11 min
S2	RETRY*/RESP* to DTACK*/BERR*setup time	35 min	10 min
S3	Data Setup time, SST160 (3USST80) mode	18 min	12 min
S3	Data Setup time, SST267 (3USST133) mode	10.8 min	7.2 min
S3	Data Setup time, SST320 (3USST160) mode	9 min	6 min
H1	Address hold time	5 min	5 min
H3	Data hold time, SST160 (3USST80) mode	18 min	12 min
H3	Data hold time, SST267 (3USST133) mode	10.8 min	7.2 min
H3	Data hold time, SST320 (3USST160) mode	9 min	6 min
T1	Handshake delay time	0 min	0 min
T2	Slave delay time (Note 1)	0 min	0 min
T2	Broadcast Master delay time (Note 2)	100 min	100 min
T3	Delay time (Note 3)	30 min	5 min
T4	Broadcast - DS0* to RETRY*/RESP* (Note 4)	65 max	90 max

Note 1: This time is used for non-broadcast transfers.

Note 2: This time is used for broadcast transfers when the master asserts the acknowledge signal.

Note 3: If the slave asserts BERR* on a write, BERR* must be asserted for time T3 prior to the negation of DTACK*. On a read, RETRY*/RESP* must be valid for time T3 prior to the assertion of BERR*.

Note 4: During a broadcast transfer, the participating slave must drive RETRY*/RESP within time T4 to ensure that setup time S2 is met.

Table 3.2: Standard VME64 Timing Parameters

Timing Parameter	Description	@ Master Min (ns)	@ Slave Min (ns)
4	IACK*, AM[5:0], and Address valid to AS* assertion setup time	35	10
6	Time from DTACK* high to when Data Lines can be driven by MASTER	0	0
7	Delay time from when data lines released to when MASTER can assert DS1* to begin the data transfer phases	0	0
8	Data lines to DS0*/DS1* setup time	35	10
10	Delay time from AS* to the first assertion of DS0*/DS1*	0	-10
12	WRITE* valid to DS0*/DS1* setup time	35	10
16	IACK* and AM[5:0] hold time from the last assertion of DTACK* or BERR*	0	0
18A	AS* negation time from last DTACK* or BERR*	0	0
23	WRITE* hold time from the last negation of DS0*/DS1*	10	0
26	Delay time from the first assertion of DS1* to start the data transfer phases until SLAVE can drive the data lines	0	0
30	Delay time from when DS0*/DS1* are negated until the SLAVE can negate DTACK* and BERR*.	0	0
31	Delay time from when data lines are released before SLAVE can drive/release DTACK*/BERR* high at the end of a read data phase.	0	0

.

Table 3.3: 2eSST Strobe Timing

Timing Parameter	Description	SST160 3USST80 (ns)	SST267 3USST133 (ns)	SST320 3USST160 (ns)
T5	Strobe Period	100 min	60 min	50 min
T6	Strobe High	50 nominal	30 nominal	25 nominal
T7	Strobe Low	50 nominal	30 nominal	25 nominal

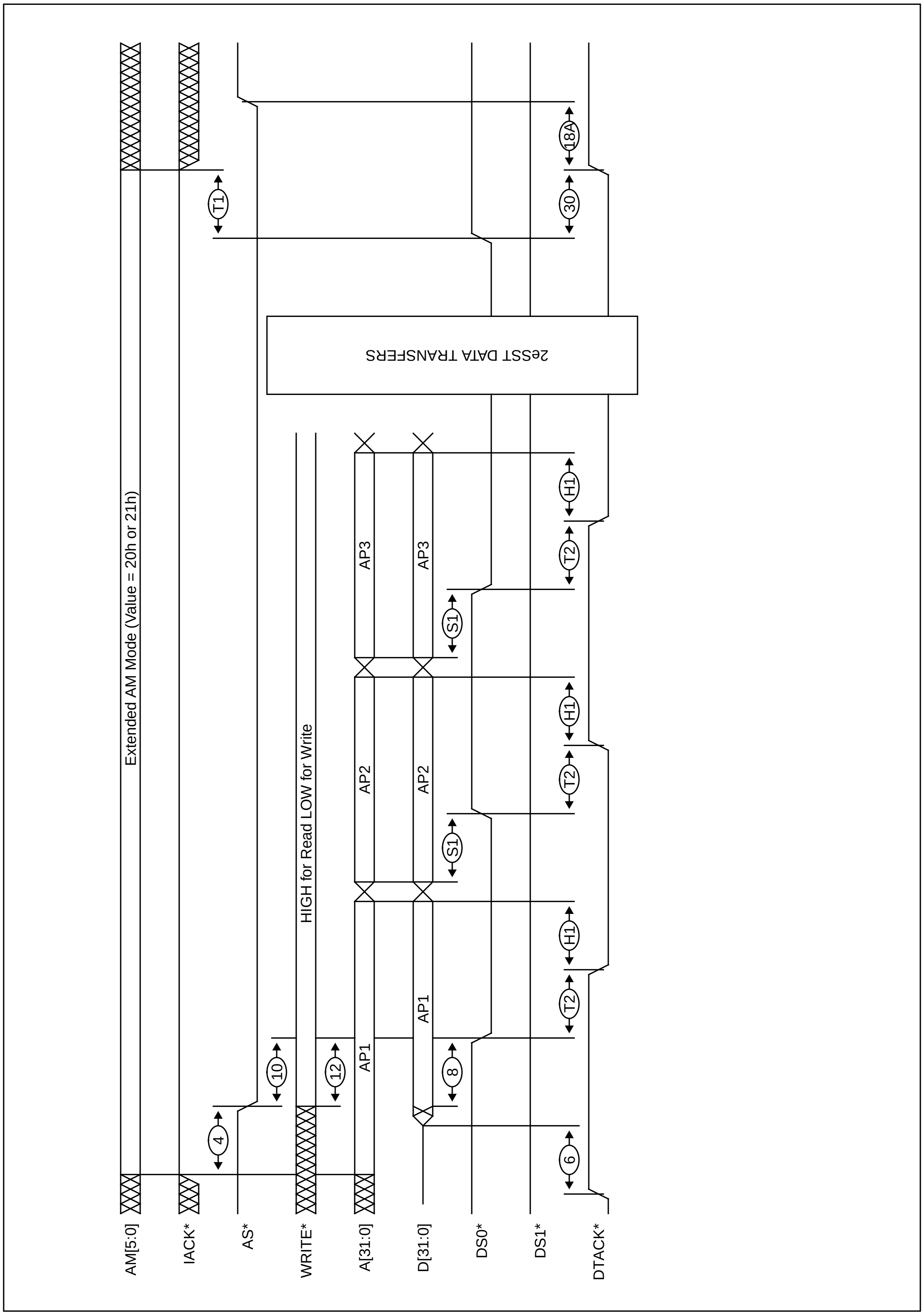

Figure 3.1: 2eSST Address Transfer

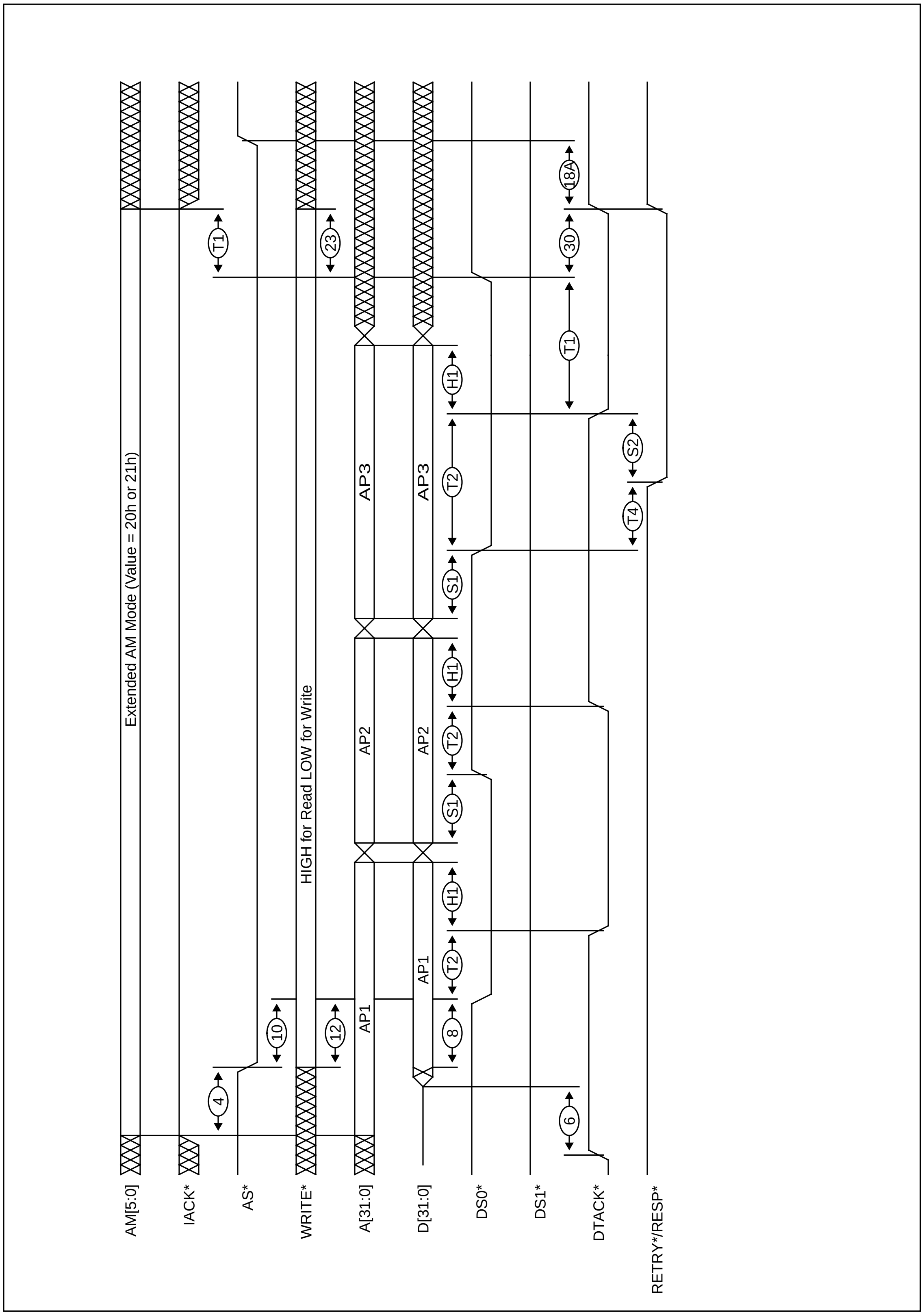

Figure 3.2: 2eSST Address Transfer - Slave Suspend Response

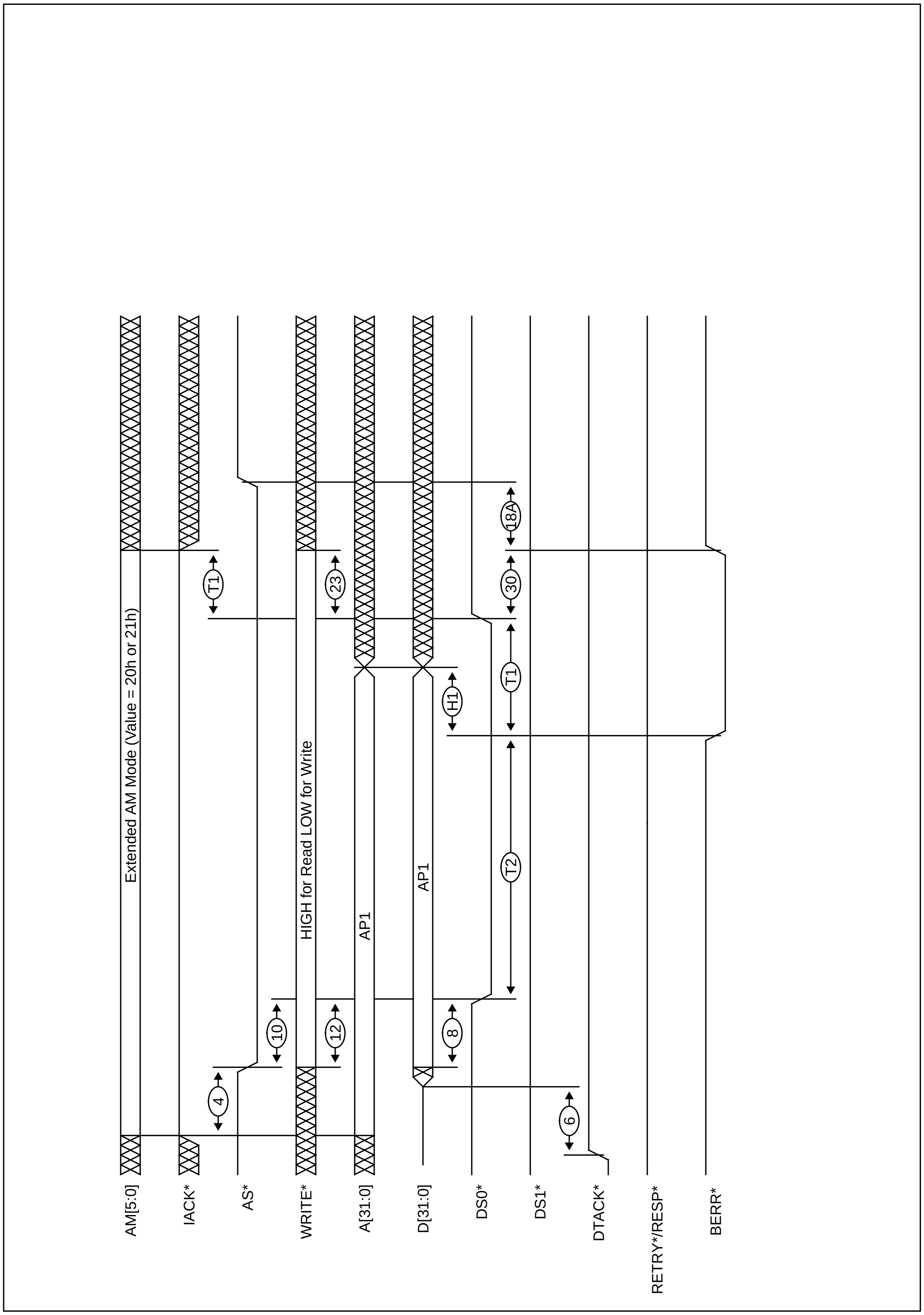

Figure 3.3: 2eSST Address Transfer - Slave Error Response

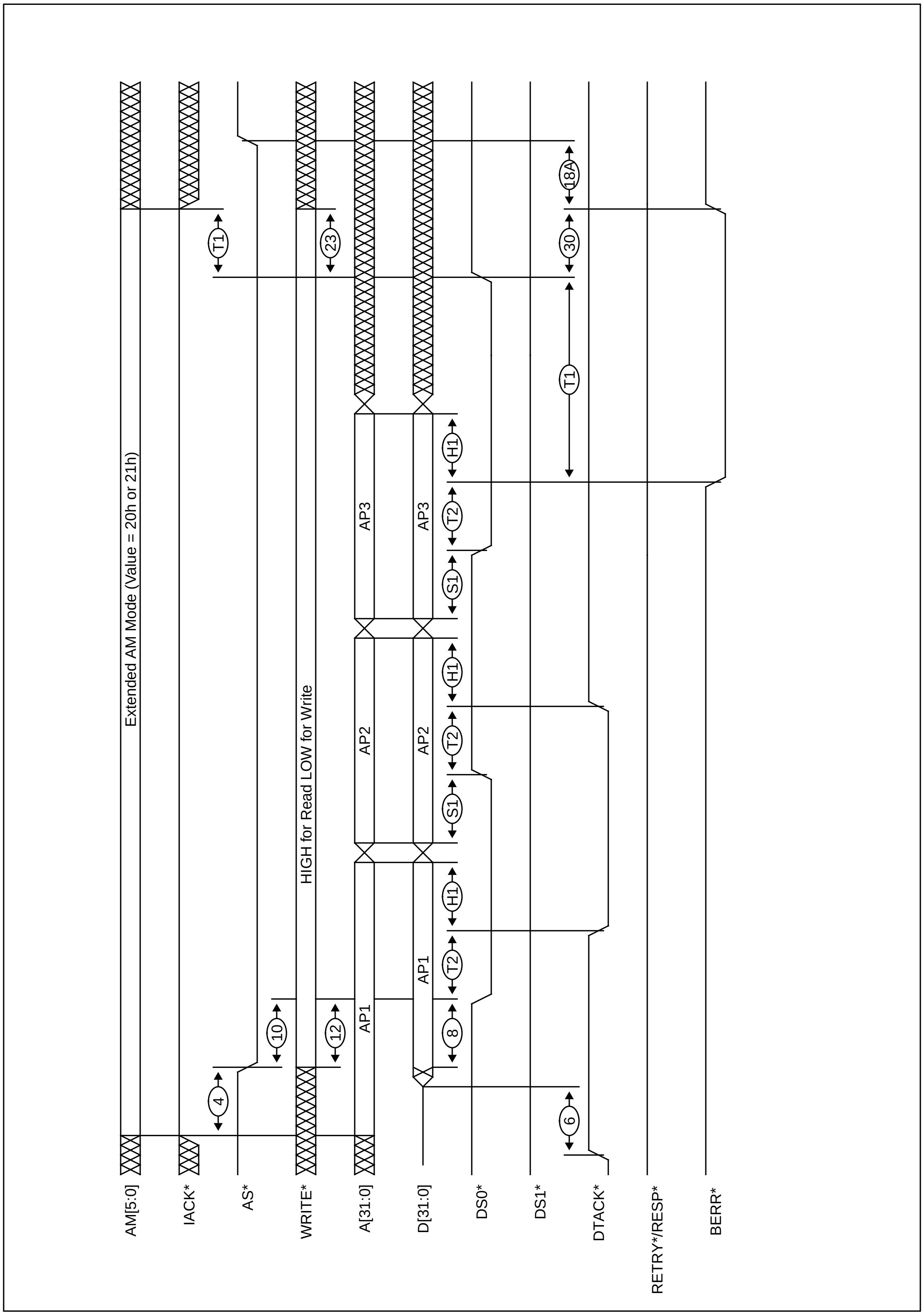

Figure 3.4: 2eSST Address Transfer - Slave Error Response

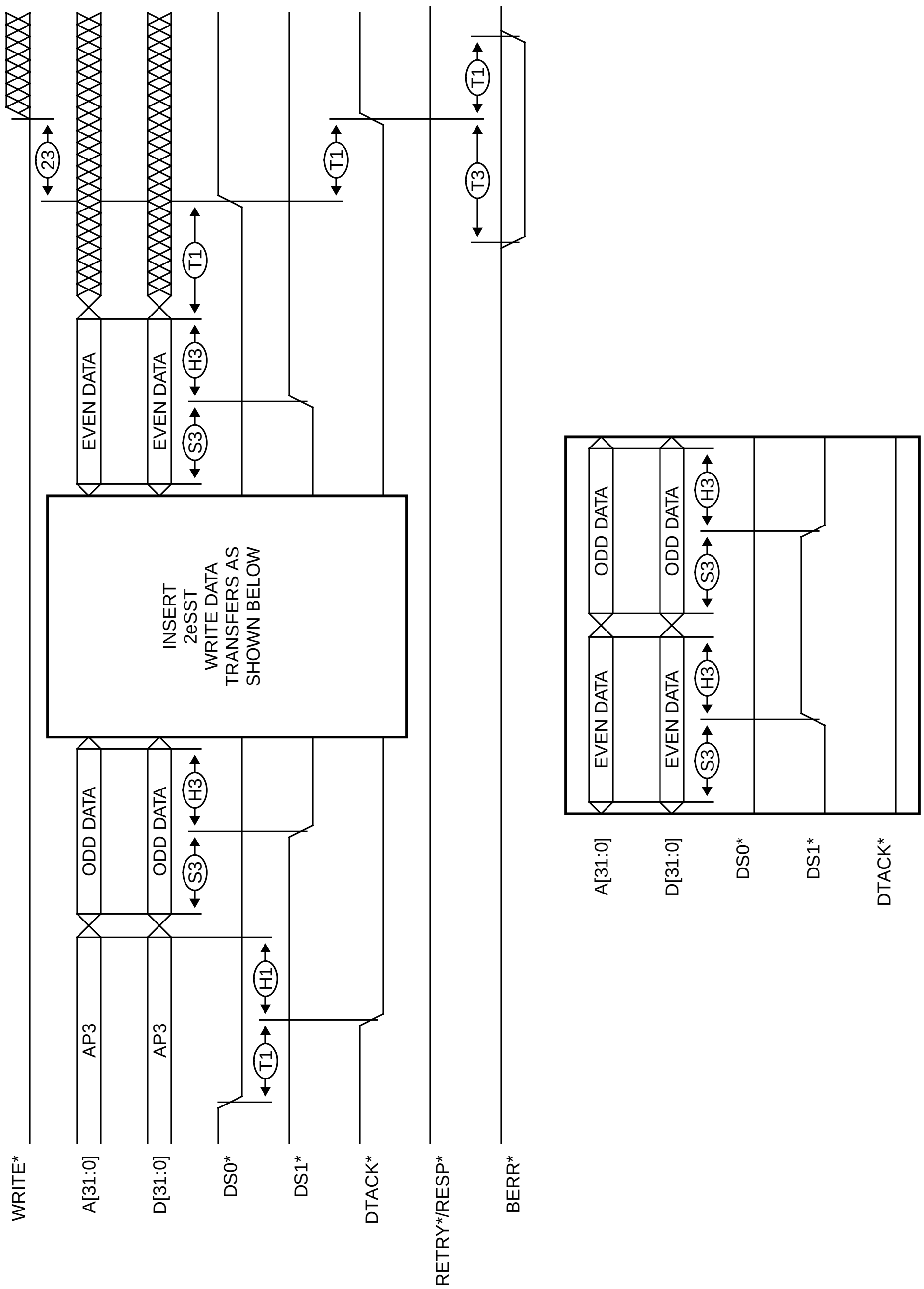

Figure 3.5: 2eSST Write

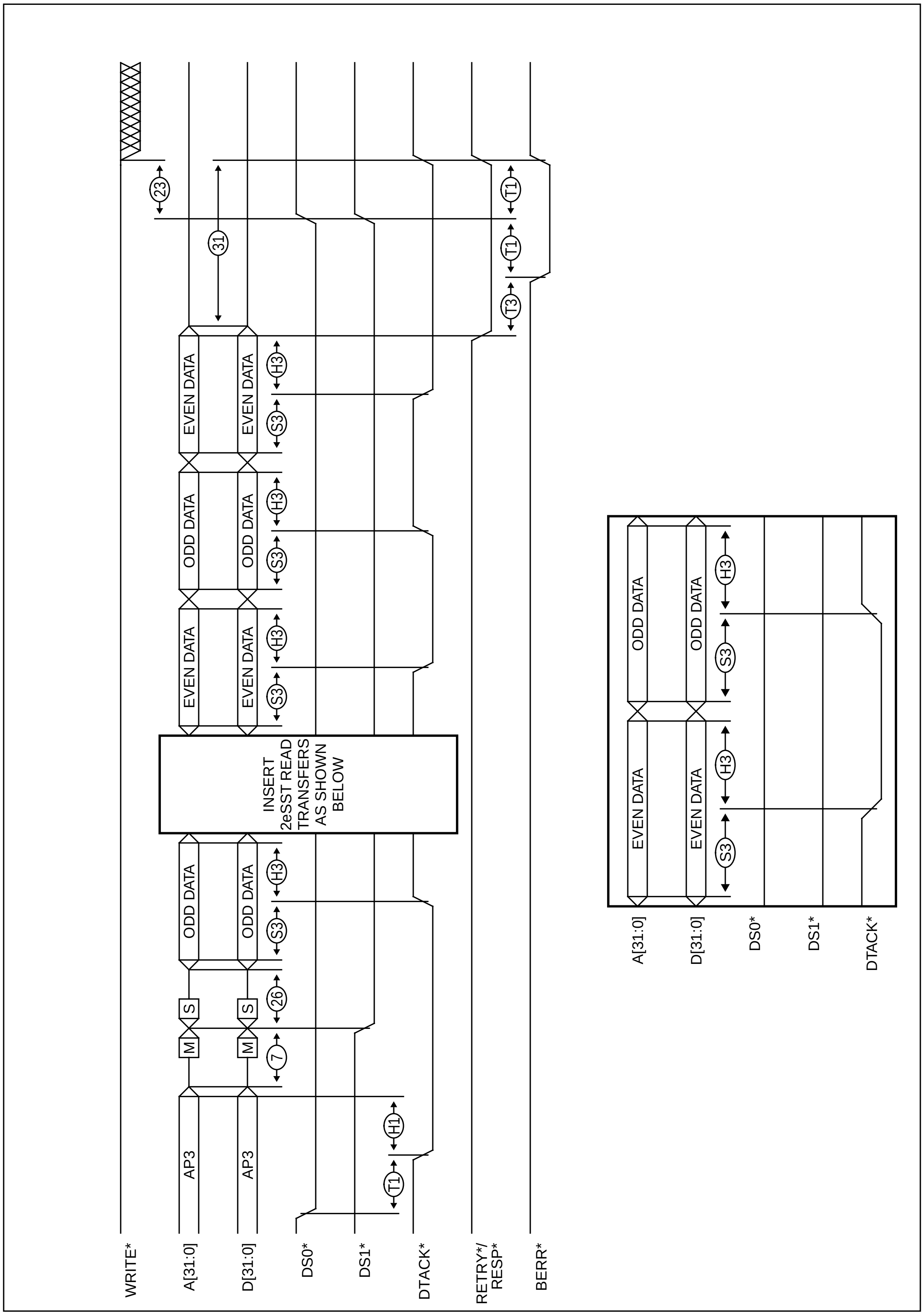

Figure 3.6: 2eSST Read

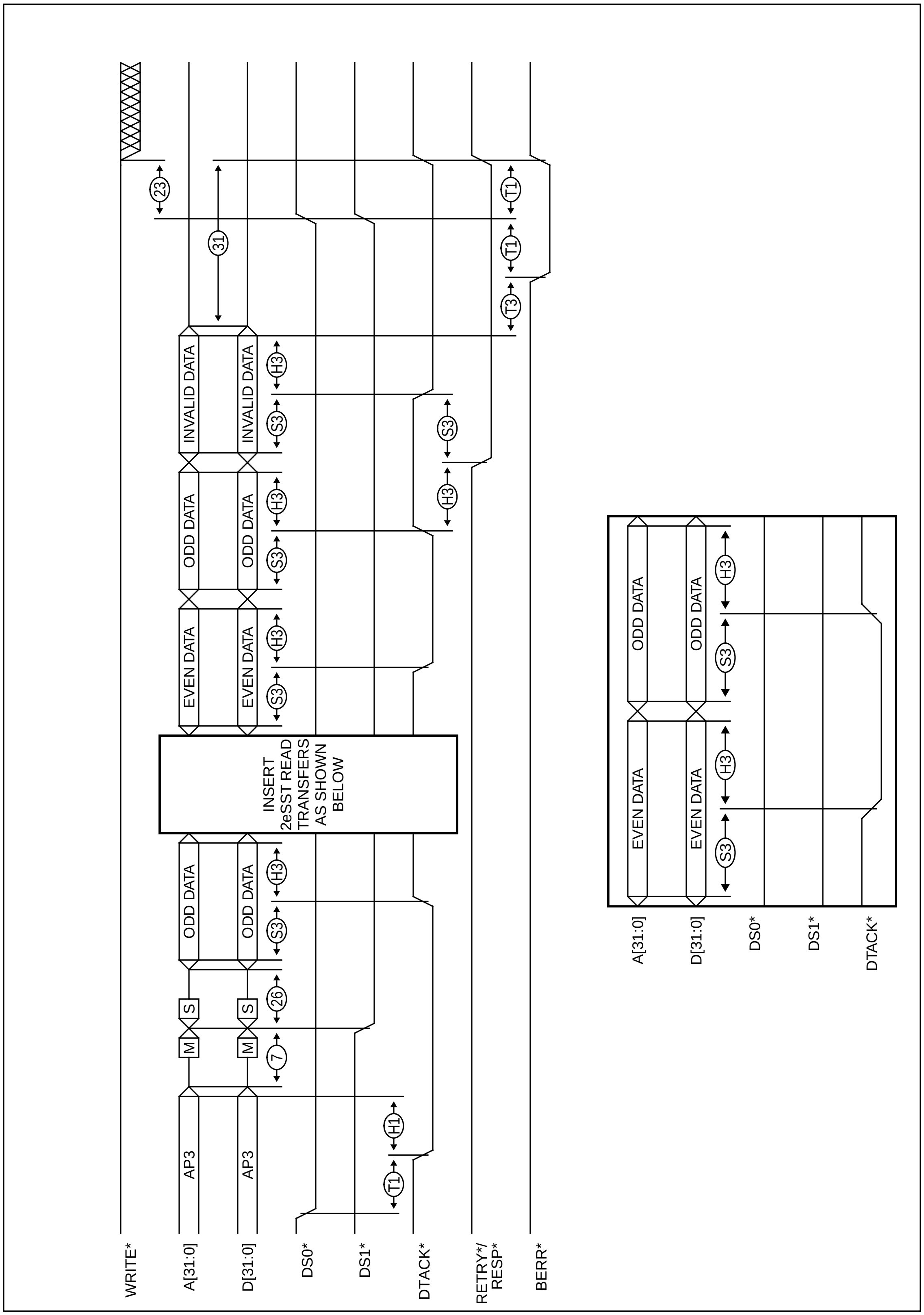

Figure 3.7: 2eSST Read, Last Word Invalid

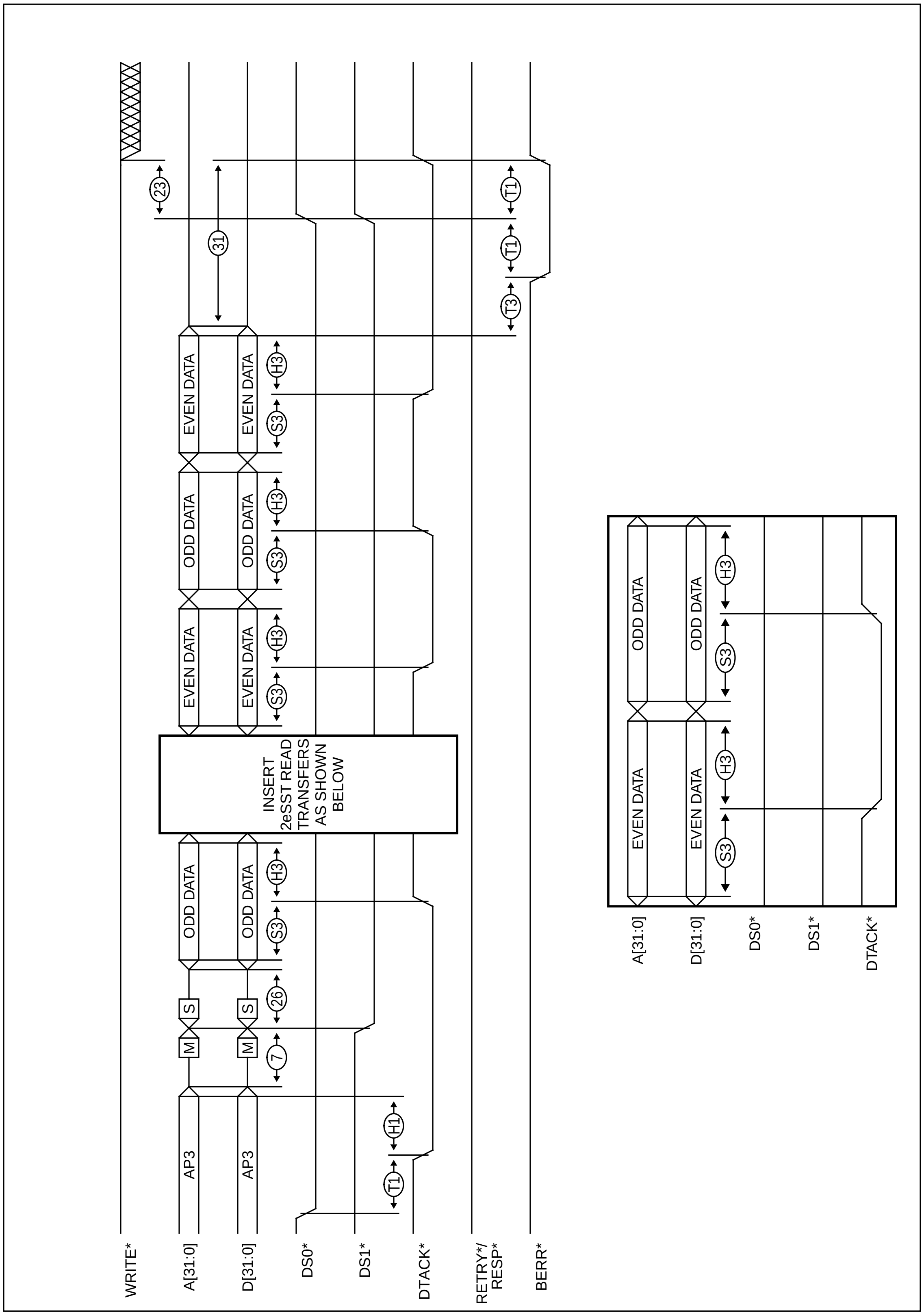

Figure 3.8: 2eSST Read, Error Termination

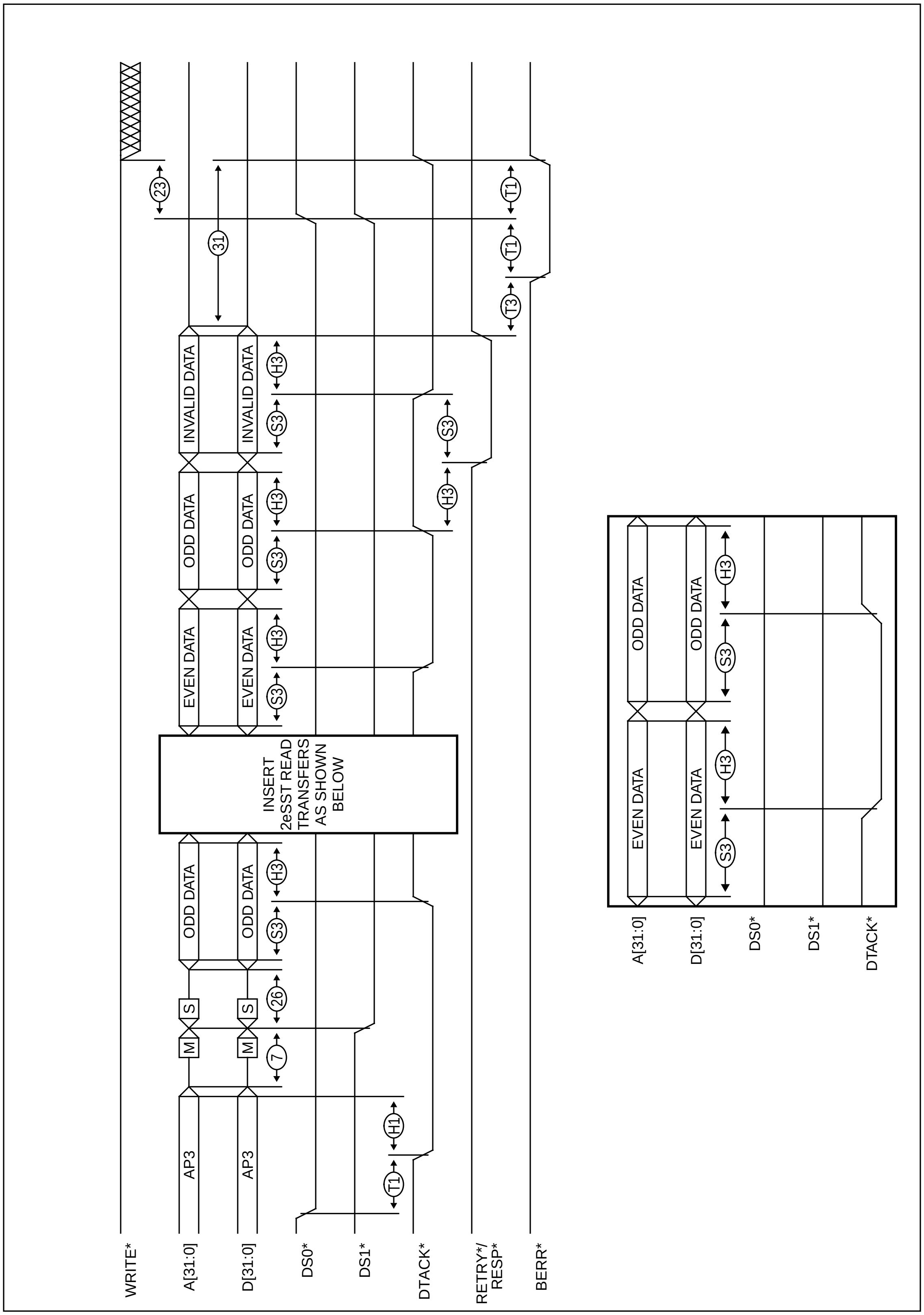

Figure 3.9: 2eSST Read, Last Word Invalid, Error Termination

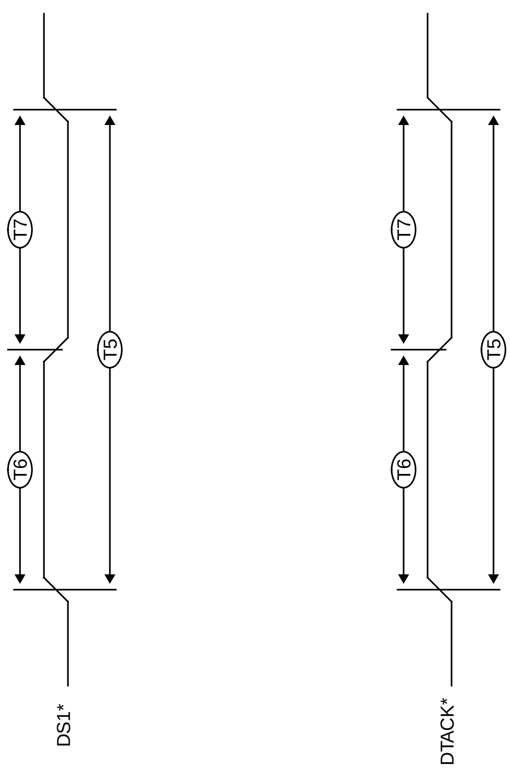

Figure 3.10: 2eSST Strobe Timing

Chapter 4

Theory of Operation

4.1 Introduction

This chapter describes the theory of operation behind the 2eSST protocol and source synchronous transfer.

4.2 Source Synchronous Transfers

Traditional VMEbus transfers utilize a handshake protocol whereby data strobes (DS1* and DS0*) are acknowledged by DTACK* which then allows the data strobes to be removed which in turn allows the DTACK* to be removed. Once DTACK* is deasserted, a new cycle can begin. The traditional VME protocol requires four delays through the drivers, backplane and receivers plus the settling time of the backplane. In contrast, after the address phase, the 2eSST protocol sends the data and strobe and does not wait for any acknowledgments. Therefore, data can be streamed at much higher rates.

The transfer rate of the system is not determined by the propagation delay from source to destination, but by skew: the variation in propagation delay through the drivers, backplane and receivers. For example, a system with a 20 nanoseconds of propagation delay and ±2 nanoseconds of skew can operate at a higher transfer rate than a system with 10 nanoseconds of propagation delay and ±3 nanoseconds of skew even though the second example has shorter propagation delay. The transfer rate of the source synchronous protocol is determined by how precisely skew can be controlled in the system.

4.3 Data Centered Strobes

The 2eSST protocol operates with strobes that switch in the center of the data window. In other words, the strobe switches in the data window halfway between when data transitions are allowed to occur. This has several very important benefits in terms of minimizing the skew.

1. The strobe, which is the single most important timing signal qualifying the data, switches when the data lines are quiet and there is minimal crosstalk and ground bounce to affect the strobe timing.

2. The source skew of the strobe is minimized since the delay between data and strobe can be done very precisely utilizing the opposite phase of the transmit clock or a 2x clock. If the opposite phase of the transmit clock is used, the duty cycle must be controlled.

3. The destination skew is also minimized since the receiver can use the strobe to clock the data. This minimizes the logic on the receive side.

4. Since the strobe switches in the center of the data window (at the transmitter), the slower data rates have improved setup and hold times.

4.4 Skew

In an ideal world, the transmitted data would change precisely at the beginning of the data window and the strobe would change precisely in the center of the data window as defined by the 2eSST protocol. This relationship between data and strobe would remain constant as the signals travel from the transmitter to the receiver. However, in the real world, the receiver can see the relationship of the data and strobe change. These variations in timing are called skew.

A 2eSST transfer involves a transmitting VME board (the source), a receiving VME board (the destination) and a backplane. Each of these three elements will contribute to the overall skew. Skew must be minimized in each element to reduce the total skew. Controlling the skew is the key to a successful 2eSST implementation.

4.4.1 Source skew

Source skew occurs at the transmitter (the source) before the data and strobe signals reach the backplane. On a 6U VME board there are 64 data lines and one strobe line involved in a 2eSST transfer. Therefore, there are 65 different signal paths. Each signal path includes logic gates, printed circuit board (PCB) traces and VMEbus drivers. All of these elements will introduce different delays into the signal path, which will ultimately result in skew. Since each element of signal path is a potential source of skew, the best way to reduce skew is to reduce the number of elements and maintain an equal number of elements in each path.

Observation 4.1:

The various VMEbus drivers and logic devices are all on the same card so they will have similar (but not identical) supply voltages and temperatures. Therefore, do not expect to see worst case differences in propagation delay between the drivers.

Observation 4.2:

Variations in wiring length and loading between the signal paths will result in different delays, which will increase skew. Reducing trace length differences and loading variations will reduce skew.

Observation 4.3:

If buffers (such as 74XX245) are used as the VMEbus drivers, then the skew of the drivers will contribute to the total skew. In some applications, registered transceivers (such as 74XX646, which include a register and driver) can be used as pipeline or deskewing devices and VMEbus drivers. These devices include a register and driver in a single package and will reduce the number of discrete logic elements in the critical signal paths and isolate the output from any skew on the input. This can reduce the skew in the signal path.

4.4.2 Backplane Skew

Typically the backplanes are designed with controlled impedances and equal length traces between slots to minimize skew. In a typical system, asymmetrical loading of the signal lines causes the majority of the skew introduced into the system by the backplane. This skew can be substantial.

Observation 4.4:

The various data bits can have substantially different amounts of loading. This variation must be accounted for as part of the backplane skew. For example, if other cards plugged into the backplane support say D16 (but not D32), then bits D[15:0] can be heavily loaded but bits D[31:16] can be lightly loaded. Furthermore, the address bits (which are also used for data) can be heavily loaded for say A[23:1] but lightly loaded for A[31:24].

Observation 4.5:

A lightly loaded 21-slot backplane utilizing the VME320 technology can have about 200 pF equivalent capacitance. A fully loaded backplane can have around 450 pF capacitance.

Observation 4.6:

While all boards would monitor DTACK*, not all boards are required to monitor DS1*. Therefore, the strobe loading can be different for transmit and receive.

4.4.3 Destination Skew

Destination skew occurs at the receiver (destination). Destination skew is caused by differences in delay between the data and strobe signals, which are caused by the unequal propagation delays in the logic and traces. These are the same effects that caused source skew. Destination skew includes an additional source of skew. Variations in the receiver threshold combined with variations in signal rise and fall times can introduce quite significant skew.

Observation 4.7:

Normal TTL circuits are specified for worst case thresholds between 0.8 and 2.0 volts over the full range of voltage and temperature. Since the strobe receiver is on the same card as the data receivers, they will have similar (but not identical) supply voltages and temperatures. Therefore, do not expect to see worst case differences in thresholds and/or propagation delays between the strobe receiver and the data receivers.

Observation 4.8:

Tighter control and/or specification of the receiver thresholds can substantially reduce skew. The ETL and the ABT parts have tighter specifications than normal TTL parts and therefore will have lower skew.

The 2eSST design guarantees that the slowest data has been presented to the destination registers at least one set-up time before the fastest strobe can arrive to clock the registers. The set-up time must be included as part of the destination timing budget.

The 2eSST design guarantees that the slowest strobe has already clocked the receiving registers at least one hold time before the fastest data can change at the registers. The hold time must be included as part of the destination timing budget.

4.4.4 Timing Budget

Since the source backplane and destination can all introduce skew, a timing budget must be established and limits defined for each element. The skew budget, receiver setup time and receiver hold times for the 2eSST protocol are defined in Table 4.1.

Table 4.1: 2eSST Timing Budget

Transfer Rate	Parameter	Time (ns)
SST160 3USST80	Transmitter skew	±7.0
	Backplane skew	±6.0
	Receiver skew	±7.0
	Receiver setup time	5.0
	Receiver hold time	5.0
SST267 3USST133	Transmitter skew	±4.2
	Backplane skew	±3.6
	Receiver skew	±4.2
	Receiver setup time	3.0
	Receiver hold time	3.0
SST320 3USST160	Transmitter skew	±3.5
	Backplane skew	±3.0
	Receiver skew	±3.5
	Receiver setup time	2.5
	Receiver hold time	2.5

Appendix A

Glossary of Additional 2eSST Terms

Introduction

This is an extension of the terms defined in the VME64 and VME64x Standards, Appendix A. Terminology described in these standards are not repeated in this standard. Refer to the VME64 and VME64x Standards for a listing of the VME64 Terminology.

3U

A term generally used to describe single high VME boards that are 100 mm in height. In this standard, 3U is also used to define boards that only use the P1 connector for communication.

6U

A term generally used to describe double high VME boards that are 233.35 mm in height. In this standard, 6U is also used to define boards that use both the P1 and P2 connectors for communication.

Broadcast Transfer

A transfer in which the master transmits data to multiple slaves. During a broadcast transfer, the slaves are participating slaves rather than responding slaves.

Participating Slave

A participating slave receives data from the master during a broadcast transfer. A participating slave participates in the transfer but does not drive DTACK* during the address phase or data phases (does not respond to the master).

Responding Slave

A responding slave responds to the master during normal (non-broadcast) 2eSST transfers. A responding slave drives DTACK* (responds to the master) during the address and data phases.

Source Synchronous Protocol

A protocol where the source (the transmitter) transmits data synchronized to a strobe.

Subunit Number in Master

If a 2eSST master has several devices, such as multiple DMA controllers or an interface to multiple processors, a subunit number can be assigned to each device. A device can identify itself by sending its subunit number during address phase two.

Two Edge Protocol

A protocol that transfers data using both the rising and falling edges of the strobe signal.

VME320 Backplane

A backplane using a unique, patented, wiring topology.

Appendix B

Backplane Technology

The 2eSST protocol requires monotonic rising and falling edges of the bus signals at the receiver and low skew between the bus signals. Any backplane topology that satisfies this requirement and is compatible with the specified bus drivers may be used. For example, standard backplane topologies which are limited in length will behave as a lumped loads rather than a transmission lines. Backplanes that are long enough to appear as transmission lines will require special topologies to mitigate the transmission line effect.

Backplanes utilizing the VME320 technology are designed to meet the requirements of the 2eSST draft standard. Backplanes utilizing the VME320 technology have a different wiring pattern (US Patent Number 5,696,667, issued December 12, 1997) to achieve their performance. For example, in a 21-slot backplane, corresponding pins on every slot are wired directly to slot 11 rather than connecting first to adjacent slots.

The effect is that the backplanes utilizing the VME320 technology do not act as distributed transmission lines with resulting reflections and ringing. Instead, they act as lumped capacitances. Backplanes utilizing the VME320 technology act like a lumped capacitance concentrated at the center slot (slot 11 in a 21-slot backplane). The wiring between say slot 1 (or slot 21) and slot 11 acts as a lumped inductance of about 80 nH. To this must be added the inductance of the card stub. Therefore, the lumped inductance and the lumped capacitance act like a single-section LC delay element that has a delay equal to the square-root of L times C. For maximum loading (450 pF) this works out to 6.3 ns. For minimum loading (200 pF) this works out to 3.8 ns. The difference is a skew of ±2.5 ns (5 ns total).

The lumped capacitance equivalent circuit of backplanes utilizing the VME320 technology tends to slow the signal rise-times and fall-times. While this is generally advantageous, it does introduce skew at the receivers due to variations in risetime caused by variations in loading and variations in driver capability.