To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 29 October 2002
Subject: 02-451r1 SAS bit and byte ordering overview

Revision History
Revision 0 (28 October 2002) First revision
Revision 1 (29 October 2002) Incorporated feedback from the SAS call. The group voted to remove the endianness example table, so it won't make it into SAS (but is left here for reference).

Related Documents
sas-r02b - Serial Attached SCSI revision 2b

Overview
An endianness overview is needed (especially since SATA uses little-endian, with different byte ordering but the same bit ordering).

Suggested Changes

3.6 Bit and byte ordering

In a field in a table consisting of more than one bit that contains a single value (e.g., a number), the least significant bit (LSB) is shown on the right and the most significant bit (MSB) is shown on the left (e.g. in a byte, bit 7 is the MSB and is shown on the left; bit 0 is the LSB and is shown on the right). The MSB and LSB are not labeled if the field consists of 8 or fewer bits.

In a field in a table consisting of more than one byte that contains a single value (e.g., a number), the byte containing the MSB is stored at the lowest address and the byte containing the LSB is stored at the highest address (i.e., big-endian byte ordering). The MSB and LSB are labeled.

NOTE 1 SATA numbers bits within fields the same as this standard, but uses little-endian byte ordering.

In a field in a table consisting of more than one byte that contains multiple fields each with their own values (e.g., a descriptor), there is no MSB and LSB of the field itself and thus there are no MSB and LSB labels. Each individual field has an MSB and LSB, but they are not labeled.

Multiple byte fields are represented with only two rows, with the non-monotonically increasing byte number indicating the presence of additional bytes.

A data dword consists of 32 bits. Table 1 shows a data dword containing a single value, where the MSB is on the left in bit 31 and the LSB is on the right in bit 0.

Table 1 — Data dword containing a value

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSB																Value															LSB

Table 2 shows a data dword containing four one-byte fields, where byte 0 (the first byte) is on the left and byte 3 (the fourth byte) is on the right. Each byte has an MSB on the left and an LSB on the right.

Table 2 — Data dword containing four one-byte fields

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSB			Byte 0 (First byte)				LSB	MSB		Byte 1 (Second byte)			LSB	MSB		Byte 2 (Third byte)			LSB	MSB		Byte 3 (Fourth byte)			LSB						

[The following table was voted out of this proposal. The rest of the material is just informative for this proposal only.]

Table 3 shows some example fields in big endian (e.g., SAS address frames, SSP frames, and SMP frames).

Table 3 — Big endian example

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB) [a]	3-bit field	(LSB) [a]	(MSB)				
1	12-bit field						(LSB)	1-bit field
2	(MSB)	16-bit (2 byte) field						
3								(LSB)
4	(MSB)	32-bit (4 byte) field						
7								(LSB)
8	(MSB) [a]	8-bit (1 byte) field						(LSB) [a]
9		56-bit (7 byte) field (e.g., containing a descriptor)						
15								

[a] These MSB and LSB labels are not normally included in this standard.

Table 4 shows the first 4 bytes of that example as they appear in a data dword. When this dword appears on the wire after 8b10b coding, a bit related to bit 24 (in the middle of the 12-bit field) will appear first and a bit related to bit 7(in the middle of the 16-bit field) will appear last. (The bits are swapped within each byte going into the CRC generator, so the CRC covers burst errors across dword boundaries.)

Table 4 — Data dword containing first four bytes of the big endian example

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSB	3-bit field	LSB	MSB	12 bit field										LSB	1-bit field	MSB	16-bit field														LSB

Table 5 shows similar example fields in little endian (e.g., STP frames). Notice how the unaligned fields in

bytes 0 and 1 must be rearranged to make the 12-bit field contiguous in little-endian sense.

Table 5 — Little endian example

Bit Byte	7	6	5	4	3	2	1	0
0					(LSB)	(MSB) [a]	3-bit field	(LSB) [a]
1	1-bit field	(MSB)			12-bit field			
2								(LSB)
3	(MSB)			16-bit (2 byte) field				
4								(LSB)
7	(MSB)			32-bit (4 byte) field				
8	(MSB) [a]			8-bit (1 byte) field				(LSB) [a]
9								
15				56-bit (7 byte) field (e.g., containing a descriptor)				

[a] These MSB and LSB labels are not normally included in this standard.

Table 6 shows the first 4 bytes of that example as they appear in a data dword. When this dword appears on a physical link after 8b10b coding, a bit related to bit 0 (the LSB of the 3-bit field) will appear first and a bit related to bit 31 (the MSB of the 16-bit field) will appear last. (Unfortunately, SATA did not swap bits 31 to 0, 30 to 1, etc. going into the CRC generator, so the CRC doesn't cover burst errors across dword boundaries as well as it could.)

Table 6 — Data dword containing first four bytes of the little endian example

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSB								16-bit field							LSB	1-bit field	MSB											LSB	MSB	3-bit field		LSB

An ASCII string is a collection of bytes, each which has its own MSB and LSB. The string itself does not have an MSB and LSB. An ASCII string appears the same in both little-endian and big-endian systems. Table 7

shows an example string.

Table 7 — String example containing “abcdefgh”

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB) [a]			First byte (‘a’)				(LSB) [a]
1	(MSB) [a]			Second byte (‘b’)				(LSB) [a]
2	(MSB) [a]			Third byte (‘c’)				(LSB) [a]
3	(MSB) [a]			Fourth byte (‘d’)				(LSB) [a]
4	(MSB) [a]			Fiifth byte (‘e’)				(LSB) [a]
5	(MSB) [a]			Sixth byte (‘f’)				(LSB) [a]
6	(MSB) [a]			Seventh byte (‘g’)				(LSB) [a]
7	(MSB) [a]			Eighth byte (‘h’)				(LSB) [a]

[a] These MSB and LSB labels are not normally included in this standard.

The ATA IDENTIFY DEVICE and IDENTIFY PACKET DEVICE commands return model number, serial number, and firmware revision fields that contain ASCII strings in an unnatural order (based on the historical 16-bit parallel ATA bus). Table 8 shows an example ATA IDENTIFY DEVICE string field returning the same string as above.

Table 8 — ATA IDENTIFY DEVICE string example containing “abcdefgh”

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB) [a]			First byte (‘b’)				(LSB) [a]
1	(MSB) [a]			Second byte (‘a’)				(LSB) [a]
2	(MSB) [a]			Third byte (‘d’)				(LSB) [a]
3	(MSB) [a]			Fourth byte (‘c’)				(LSB) [a]
4	(MSB) [a]			Fiifth byte (‘f’)				(LSB) [a]
5	(MSB) [a]			Sixth byte (‘e’)				(LSB) [a]
6	(MSB) [a]			Seventh byte (‘h’)				(LSB) [a]
7	(MSB) [a]			Eighth byte (‘g’)				(LSB) [a]

[a] These MSB and LSB labels are not normally included in this standard.