System Engineering Notes
No. 825
Using the SYM53C94, SYM53C95 and SYM53C96
Rev 3.0, May 1997

SYMBIOS
L O G I C

Symbios Logic has discovered three software issues dealing with the SYM53C94, SYM53C95, and SYM53C96, here after referred to as the SYM53C94. The three issues described in this engineering note are:

- Back-to-Back Transfers
- Handling the Odd-byte
- Extra ACK/

The SYM53C94 parts affected by these issues can be identified by their part number. Device part numbers identify revisions and they are printed on the package below the name. The issues described in this engineering note apply only to devices with the following part numbers.

Name	Part Number
SYM53C94	609-3400508
SYM53C95	609-3400512
SYM53C96	609-3400514

1. BACK-TO-BACK TRANSFERS

Back-to-back transfers mean that consecutive bytes arrive from the Target, but the SYM53C94 transfer counter decrements to zero in mid-date stream with no corresponding phase change. This could occur in scatter-gather operations or could occur if the total number of bytes to be transferred is greater than the maximum count of a 16-bit counter.

If the following three modes are enabled simultaneously, they can cause the ACK/ signal to hang, thus producing undesirable results.

- Back-to-back transfers
- Threshold-8-mode or burst mode
- Synchronous DATA IN phase

Following is an explanation of the operational condition and the recommended device setup.

The operational condition relates to a narrow window that is a function of the speed of both the host DMA and Target. The possible cases relating to operation are described in the following paragraphs.

For illustration purposes, byte *n* is defined as either the last byte or the last word from the first transfer. The byte, or word, will be removed from the FIFO with the DACK/signal. The leading edge of the DACK/ signal will decrement the transfer counter (TC) to zero. Byte *n* + 1 is the next byte sent by the Target.

Improper Operation:

> If byte $n + 1$ arrives (SCSI REQ active) before the DACK/ for byte n has been deasserted, the SYM53C94 asserts ACK/ on the SCSI bus and leaves it asserted.

> Figure 1 shows one SCSI REQ/ that causes the SCSI ACK/ signal to remain asserted during the last DMA burst. (Time t_1 is greater than zero.)
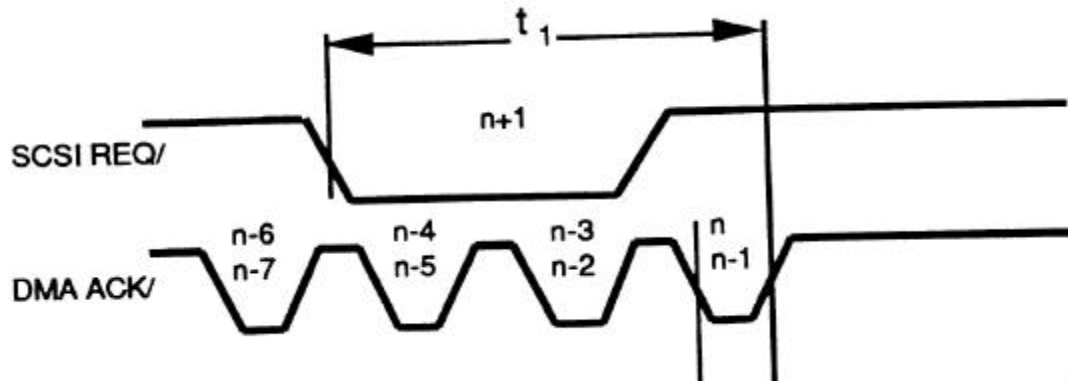

Figure 1. Case One (Failure)

Proper Operation:

The device functions normally if $n + 1$ does not arrive until after DACK/ goes high, as shown in Figure 2; or if $n + 1$ and $n + 2$ arrive before DACK/ goes active, as shown in Figure 3.

Figure 2 shows one SCSI REQ/ that will function normally during the last DMA burst. (Time t_2 is greater than zero.)
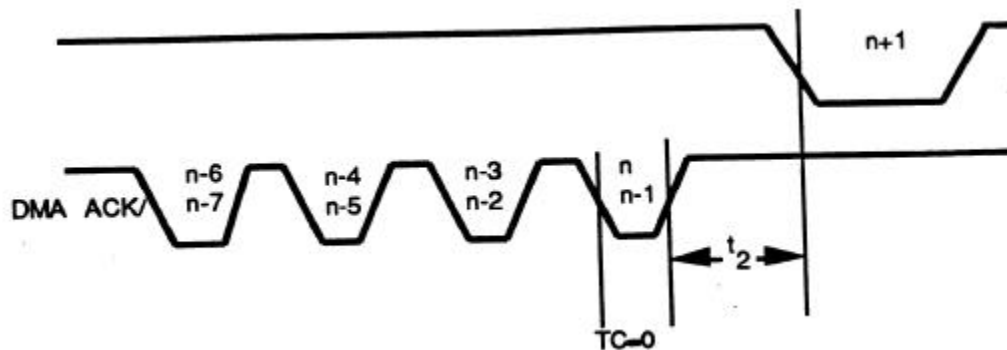

Figure 2. Case Two (Pass)

Figure 3 shows two SCSI REQ/s that will function normally during the last DMA burst. (Time t3 is greater than zero).
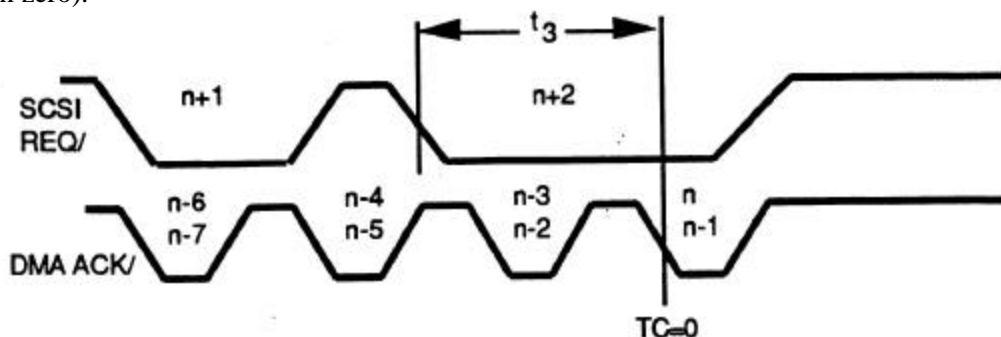

Figure 3. Case Three (Pass)

Note that this condition (ACK/ hanging) will not occur if

- the Target changes phase when the transfer counter decrements to zero, or
- if the Target pauses until DACK/ deasserts after the transfer counter decrements to zero.

Since the relationship of timings between the host DMA and Target may not be predictable, it is safer to avoid using the three modes simultaneously.

2. HANDLING THE ODD-BYTE

In modes 1 and 3 the SYM53C94 can only transfer 16-bit words over the DMA interface. Occasionally the SCSI transfer will end with one byte, rather than one word. This last byte is called the 'odd-byte'. Save Residual Byte (SRB) mode may be used to prevent a DMA transfer when there is a single byte left at the end—only when data is flowing into the FIFO from the SCSI bus. If SRB is not enabled, then the last byte will be transferred over the DMA interface as the lower byte of a word with the upper 8 bits being set to 0. The Transfer Counter is used to determine the end of the transfer. SRB prevents the assertion of DMA Request (DREQ) only if the Transfer Counter decrements to one and the phase is Data In for Initiator mode or Data Out for Target mode. Here are ways to successfully handle the odd-byte.

Initiator Asynchronous Data Out

SRB has no effect during Initiator Data Out. Let 'n' be the number of bytes to send to the Target. Set the Transfer Count to n-1, then issue a DMA Transfer Info command. An interrupt will be generated when the Transfer Counter decrements to zero; the Target will still be waiting for the last byte. Have the processor write the odd-byte to the FIFO at address 02, then issue a non-DMA Transfer Info command. It does not matter whether the Target continues to request Data Out after receiving the odd-byte.

Initiator Asynchronous Data In

When SRB is enabled, DREQ will not be asserted for the last byte. An interrupt will be generated, from the previous DMA Transfer Info command, with the Transfer Counter equal to one. The processor may remove the odd-byte from the FIFO by reading address 02. It does not matter whether the Target continues to request Data In after sending this odd-byte. If the Target continues to send data, then a complete 16-bit word may optionally be transferred by DMA with the odd-byte becoming the lower half and byte n+1 becoming the upper half of the word. This is accomplished by issuing a new DMA Transfer Info command.

Another way of handling the odd byte in this case is to use a non-DMA Transfer Info command, as described in Initiator Asynchronous Data Out, above.

Initiator Synchronous Data Out

SRB has no effect during Initiator Data Out. Use a non-DMA Transfer Info to handle the odd byte as described in asynchronous Data Out above. The current printing of the Data Manual incorrectly states that DMA must be used for all synchronous data transfers. It should say that DMA must be used for Initiator Synchronous Data In.

Initiator Synchronous Data In

When SRB is enabled, DREQ will not be asserted for the odd-byte. But DMA is required for synchronous Data In transfers. The odd-byte may be read out of the FIFO provided the Target stops sending data after sending the odd-byte. The chip's internal offset counters are not updated when the processor reads from the FIFO. Because of this, subsequent requests for Data In would eventually cause the SCSI bus to hang after the processor reads the odd-byte.

If the Target continues to send data, the odd-byte then becomes the lower half of a 16-bit word. In this case, SRB works like Reserve FIFO Byte (RFB) – except that the processor must not access the FIFO bottom. A subsequent DMA Transfer Info command will cause the odd-byte from the last sub-block to become the "reserved" byte for the subsequent block. If the memory location of the second sub-block is contiguous with the first, then no intervention is required from the processor. The most difficult case is scatter-gather in protected mode systems. 'Scatter' means that data coming off the disk is to be scattered in non-contiguous locations through out memory. These are called sub-blocks. If the second sub-block is misaligned, then the 16-bit DMA will correctly place the high byte in correct starting location. All subsequent word transfers will be correctly placed as well. The low byte from the first transfer is really the odd-byte of the previous sub-block. The processor must move it to the right location. In protected memory systems, it may be illegal to write the low byte of the first transfer. In that case, the word must be DMA'ed to someplace safe, then the processor must move both bytes to the proper location.

If the first sub-block has an odd-byte at the end, and the second sub-block is to begin on an even boundary, then the Initiator must insure that the SCSI phase will change after the odd-byte is transferred into the SYM53C94. The Initiator may do this by breaking up the SCSI transfer into multiple transfers.

3. EXTRA ACK/

When operating under the following conditions:

- Initiator Synchronous Data In
- Threshold-8 enabled
- Save Residual Byte (SRB) enabled

the SYM53C94 may send an extra ACK to the SCSI Target. This may occur depending upon the relative timing between the DMA bus and SCSI bus, as described above. Fortunately, many Target controllers, including SYM53C90 family devices, will ignore this extra ACK and proceed normally.

The only way to avoid this problem is to turn off either Threshold-8 or SRB modes during Initiator Synchronous Data In.