Determining Arbitration and Threshold Levels in a SONIC™ Based MicroChannel® Adapter

National Semiconductor
Application Note 747
Bill Carlson, FAE
June 1993

ABSTRACT

With the number of bus master adapter boards increasing in MicroChannel based systems, many issues arise. This is especially true regarding Bus Master Ethernet LAN controllers such as the DP839EB-MCS. As such, the entire MCA environment needs to be considered so that critical settings for arbitration levels, threshold levels, and fairness options can be chosen. This paper describes these issues as they relate to National Semiconductor's DP839EB-MCS 32-bit Ethernet LAN controller board, which utilizes the DP83932 (SONIC).

The major issues include bus latency, bus efficiency and the contributing factors affecting these critical system level parameters. Factors such as bus occupancy times, DRAM refresh rates, floppy controller accesses, CPU accesses, mass storage transfer rates, latency tolerances, and priority levels all contribute to latency and efficiency. Within this environment, the high performance levels of the SONIC are achieved, even in worst-case scenarios in heavily loaded file servers with multiple bus masters.

It is also important to note that many of the basic concepts and considerations required in this application will also apply to other buses, although the detailed analysis will differ.

OVERVIEW

The DP83932 (SONIC) is a high performance, 32-bit, bus mastering Ethernet controller designed for a wide variety of applications. These applications include motherboards, routers, bridges and gateways, buffered and intelligent adapter boards, and bus master adapter boards. In each of these applications, determining the optimum thresholds and arbitration levels are key parameters to choose to ensure optimum performance. In determining these parameters, the anticipated system configuration needs to be understood. Specifically, the number and type of bus mastering devices in a system needs to be determined. Once these bus masters have been identified, the device thresholds and board arbitration levels can be determined.

Determining the anticipated number and type of bus masters directly affects a bus specification known as *Bus Latency* . Bus latency is defined as the time between when a bus master requests the bus to when it actually gets it.

Bus latency is a critical systems level specification because if it is too long, a bus master who doesn't get the bus when it needs it could suffer performance degradations or even more severe conditions such as a lost Ethernet packet or missed "sector" in a streaming tape drive. As such the Ethernet controller subsystem needs to have enough tolerance to handle large latencies to guarantee it's access to the bus and avoid this missed packet condition. The SONIC was specifically designed to perform in these applications.

By having a high speed, 66 MB/s, DMA host interface the SONIC maximizes bus bandwidth and minimizes time on the bus. Coupled with two efficient, 32 byte receive and transmit FIFOs, the SONIC will tolerate most latencies found in many applications.

Determining bus latencies is easy in many applications. Bridges and gateways, motherboards, intelligent and/or buffered adapter boards are systems in which the anticipated bus masters are known. In these systems it would be common to have the host CPU, a DMA controller, and peripheral devices (SCSI, FDDI, . . .) all known by the system designer before the product is shipped out the door.

It is the designer who has to design a bus master adapter board or motherboard for a target bus (be in MicroChannel, EISA, VME, etc.) with expansion slots who has a tougher problem. He doesn't know what the end system configuration will be so he has to design to what is anticipated to be a worst case system configuration. The adapter board designer's customers would be the systems integrators who need to make sure that his board is designed properly so it will operate in fully loaded systems and still attain the high performance that he expects from this type of bus-mastering device.

Towards this end, this paper is written to assist the SONIC adapter board designer in choosing the correct arbitration and threshold levels for an IBM PS/2 Model 80 application, most probably operating as a file server having multiple LAN and mass storage devices on the MCA bus. For designers of other systems, this paper should help in understanding many of the issues that arise in a bus master LAN environment.

Before discussing this, a few MCA specifics need to be addressed. First off is the arbitration scheme. There can be up to 8 bus master expansion boards on the Model 80 MCA bus, including 8 DMA channels, the system CPU, refresh, and NMI which are on the system motherboard. Most have their own arbitration level as programmed via a POS register. When a device wants ownership of the bus, it asserts the PREEMPT* signal and will then monitor the ARB/GNT* signal, and when high (as controlled by the central arbitration logic on the system board) will place it's arbitration vector on the bus. If it's vector has the highest value, it wins the bus, ARB/GNT* goes low, PREEMPT* is de-asserted, and it can now do data transfers. If other devices want the bus they can asynchronously assert PREEMPT*. The first device has 7.8 μs to get off the bus and then all requesting devices, including the first if it wants to, compete for the bus and the arbitration process starts over again. When determining system characteristics, this 7.8 μs is often used as it dictates the maximum amount of time that a device can own the bus if others are requesting it.

SONIC™ is a trademark of National Semiconductor Corporation.
MicroChannel® is a registered trademark of International Business Machines Corp.

©1995 National Semiconductor Corporation

Another aspect of the MCA architecture is a feature called Fairness. Fairness allows all devices access to the bus in a round-robin fashion as determined by pre-assigned priority levels. Carefully choosing which devices are fair or not allows proper performance levels for the various devices on the bus. If fairness is enabled for a device and it currently owns the bus and another device(s) wants it, it will wait to re-arbitrate until all other requesting devices have had a chance on the bus themselves (this is noticed by the absence of an active PREEMPT* signal). In this way no device will hog the bus and prevent others from accessing it. If fairness is disabled for a device, it will arbitrate for the bus any chance a valid arbitration cycle is available, regardless whether other devices are waiting to arbitrate also. Even with fairness enabled, the winner of the bus still needs the highest arbitration level, however, properly setting the fairness option will determine who will do the arbitrating.

In determining the arbitration levels and thresholds the designer of the SONIC bus master adapter board needs to account for a worst case bus situations. This would most likely be a high performance file server with multiple adapter boards. These could include an ESDI disk controller, an SCSI controller for additional disk and tape backup facilities and from 1 to 4 LAN boards to handle a heavily loaded network. Other anticipated bus master boards could also be included in this scenario (e.g., FDDI) but our discussion will be limited to the aforementioned configuration. (This is indeed a worst case scenario. A more typical case for a file server would have 1 or 2 LAN boards and both a SCSI and ESDI controller).

To summarize our worst case scenario for this analysis, we will assume the MicroChannel PS/2 has these adapter boards installed:

- 4 SONIC Bus Master Adapter Boards
- 1 Bus Master SCSI Controller
- 1 Bus Master ESDI Controller

DETERMINING ARBITRATION LEVELS AND THE FAIRNESS OPTION

When determining these it must be understood that the mass storage devices and the LAN controllers have different goals when it comes to bus utilization. The mass storage devices will have large blocks of data to transfer that are typically already stored in a local buffer on the adapter board or on the drive itself. All ESDI disk controllers have a local buffer, some with megabytes of storage. Most SCSI host adapters have buffering as well, although a trend is to use a bus-mastering SCSI controller IC that can gain the bus similar to the way the SONIC does. These don't have local buffering outside of their internal FIFO, but have the data storage on the disk drive itself. The main priority for the storage devices is to transfer as much data as possible for as long as it has the bus. Of second priority is latency toleration. These devices can wait a reasonable amount of time before they get the bus. Because they already have a large amount of data buffered, no data should be lost if it isn't granted the bus immediately. However, when it does get the bus, it needs to transfer as much as possible.

The Bus Master LAN controllers, on the other hand, need to have quicker access than the mass storage devices and within their latency period. This is especially true when receiving a packet, for to get a FIFO overrun error would cause upper protocol layers to initiate long and time consuming recovery procedures. Once they are on the bus, however, they are on for a relatively short period of time. This is due to the fast 20 MB/s MCA transfer rate and the smaller amount of data that is to be transferred at one time. (A disk or tape cache can have many Kbytes available for transfer, the 32 byte FIFO will transfer at the most that amount.)

With this in mind, the LAN controllers should be configured to have near immediate access to the bus. As such, each should be set to have a priority level higher than the storage devices. Thus whenever an arbitration takes place, a LAN controller should always participate and win so it can attain bus ownership as soon as possible. The setting of the fairness option should also be chosen to allow the LAN boards immediate bus access. If all devices had enabled the fairness option it is possible for the LAN board to be off the bus for a longer period of time than it's latency tolerance allows, for example as shown in Table I.

TABLE I. Possible (but Not Optimum) Priority Settings for Adapters, but Not the Optimum Solution

Device	Priority	Fairness
LAN0	0	Yes
LAN1	1	Yes
LAN2	3	Yes
LAN3	4	Yes
SCSI	6	Yes
ESDI	7	Yes

In this scenario all devices have fairness enabled and the LAN boards have the higher priority. If a LAN board is awaiting arbitration it will win vs. the ESDI and SCSI boards. However, since fairness is enabled for the LAN boards it means that they must defer arbitrating until all other devices have been on the bus. These boards should participate in every arbitration cycle and by enabling fairness for them, this is prevented. Specifically in this example, the SCSI and ESDI boards will be on the bus consecutively for 7.8 μs each (for 16.2 μs total, including arbitration time) and the LAN boards would miss the intermediary arbitration cycle; this might exceed the boards latency toleration. By disabling fairness on the LAN boards, each is guaranteed to participate in every arbitration cycle and not have to wait for other device's arbitrations and bus occupancy times. Because of this and their higher priority levels, a LAN board will **always** arbitrate and win when an arbitration cycle occurs. We now have this:

TABLE II. Priority Settings for Adapters with Correct Fairness Setting

Device	Priority	Fairness
LAN0	0	No
LAN1	1	No
LAN2	3	No
LAN3	4	No
SCSI	6	Yes
ESDI	7	Yes

What about the storage devices? Fairness should be enabled for them. Due to the large amounts of data available for them to transfer in their respective caches, they will always have a need to own the bus and so they will always be requesting it. If fairness were disabled, the higher priority device (the SCSI controller in this case) would hog the bus and prevent the ESDI controller from accessing it. Thus fairness should be enabled for them.

To summarize, the above configuration will give each LAN board immediate access to the bus. The SCSI and ESDI boards would each have accessibility to the bus and although delayed due to the higher priority LAN boards, their latency tolerances are much higher and would incur only a minor, yet expected loss in bus acquisition time. The settings for the DMA slave ESDI controller that is configured with the Model 80, does indeed default to these settings. Fairness is enabled for it and it occupies DMA channel 7, the lowest priority DMA Channel.

The following *Figure 1* illustrates the sequence of events in a fully loaded, extreme worst case situation by properly setting the arbitration levels and fairness. Other devices such as refresh and the floppy controller will be included later when FIFO thresholds are discussed.

It should be remembered that the system CPU, the floppy controller, refresh, and other devices will be on the bus as well. These, along with the adapter boards all contribute to bus latency. Because of this latency the SONIC's FIFO threshold must be set properly to tolerate the expected latencies and avoid overrun/underrun errors. When set properly the SONIC will achieve the high performance the designer wants and the system's integrator expects.

DETERMINING THRESHOLD LEVELS

The FIFO threshold is an option that is programmed in the SONIC's Data Configuration Register and both the receive and transmit FIFOs can be programmed for different values. What is the FIFO threshold? The threshold is simply the point in time that the DMA engine requests the bus after a certain amount of data has filled the FIFOs. For example, a threshold of 1 long word for the receive FIFO would mean that after 4 received bytes from the network have filled the receive FIFO the DMA engine will request the bus. For the transmit FIFO, a threshold of 4 long words would cause the DMA engine to request the bus when the number of bytes in the FIFO falls below 16.

When determining the threshold levels, we need to first explore the specific latencies expected in our worst case scenario. The latency calculation is done by adding together the bus occupancy times of the various bus masters, their priority levels, and the fairness option. We will assume the following:

- All adapter boards have 32-bit MCA bus master interfaces
- The SONIC board transfer rate will be at 250 ns (although MCA will operate @ 200 ns and the SONIC can do synchronous transfers on other buses @ 100 ns)
- Arbitration time will be 300 ns (0.3 μs)
- EMPTY/FILL Mode is enabled for FIFO buffering
- The Floppy controller will request service from DMA Channel 2 every 12 μs and will remain on the bus for 500 ns.
- Refresh occurs every 15.1 μs and inserts itself in the middle of an arbitration cycle, extending it 200 ns for a total arbitration time of 500 ns.

In this example we will assume that the SCSI controller just got on the bus and then immediately afterwards all four LAN boards and the ESDI controller request the bus by asserting PREEMPT*. This example takes a worst case latency and will show how the chosen threshold and arbitration levels and fairness options will guarantee proper system performance by showing how all four LAN boards will be able to access the MCA bus. When these devices request the bus it is to be understood that their FIFO thresholds have been reached. The LAN controllers will be buffering a received packet, a very critical bus access.

What should the threshold levels be for the 4 LAN controllers? Choosing the proper threshold involves trade-offs between a number of systems level specifications. By having a low threshold, maximum latency is assured. However, fewer bytes will transfer so the arbitration percentage will be higher, reducing efficiency. Also, the controller will request the bus more often causing bursty traffic across the bus. A larger threshold on the other hand, solves these problems at the expense of lower bus latency tolerance. In light of this, the thresholds of LAN0:1 should be higher than LAN2:3. LAN0:1 won't see larger latencies due to their higher priorities. However, they shouldn't request the bus again before LAN2:3 get a chance, increasing the latency they already incur. LAN2:3, however, need to tolerate longer latencies than LAN0:1 because, due to their priorities, they will be off the bus for longer periods of time. They will request the bus sooner and more often, however, this shouldn't impact system performance due to the short bus duration. By choosing a threshold of 16 bytes for LAN0:1 and 8 bytes for LAN2:3, as summarized in Table III, a good balance between these issues is achieved.

SCSI	LAN0	LAN1	LAN2	LAN3	ESDI	LAN0	LAN1	LAN2	LAN3	SCSI

FIGURE 1. Bus Ownership in Example PS/2 Under Worst Case Bus Request

Table III shows the arbitration bus priority assignments that show proper settings for the IBM PS/2 Model 80 devices. It should be remembered that these device assignments are determined by the MCA specification. Some of the assignments are pre-set, while others can be occupied by installable adapter boards. For example, refresh and NMI are pre-set to arbitration levels −2 and −1. The Floppy controller occupies DMA channel 2. The other DMA channels are available for adapter boards.

TABLE III. Arbitration, Fairness, and FIFO Threshold Settings

Device	Priority	Fairness	Threshold	Latency	Latency µs
Refresh	−2				
NMI	−1				
LAN0	0	No	16 Bytes	16 Bytes	12.8
LAN1	1	No	16	16	12.8
Floppy	2				
LAN2	3	No	8	24	19.2
LAN3	4	No	8	24	19.2
Available (Note 1)	5				
SCSI	6	Yes			
ESDI (Note 2)	7	Yes			
Available	8–E				
CPU	F				

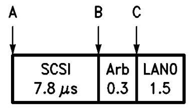
Note 1: An IBM ST-506 disk controller will default to an arbitration level of 5 with fairness enabled.

Note 2: An IBM ESDI controller will default to arbitration level of 7 with fairness enabled.

Devices 8–E are available for bus masters. In our example, DMA channels 0, 1, 3, and 4 are masked out and are used to hold the bus mastering LAN controllers. The bus master SCSI host adapter is put at ARB 6 with DMA channel 6 masked out. A standard PS/2 Model 80 comes with an ESDI disk controller operating as a DMA slave at ARB 7. This is the default setting for this controller. Because of this, the LAN designer doesn't have to worry about the arbitration level and fairness options for this controller. It can be assumed that the SCSI host adapter will be configured in the same way: with a low priority and with fairness enabled. In our example we have assumed a bus mastering ESDI controller; however, the standard one is a DMA slave device. For our discussion, though, we will assume it is a bus master for clarity's sake.

Once the arbitration levels and thresholds are determined for the LAN boards, they must be set when installed. IBM automatically sets the default values for the ESDI controller, but what about the LAN boards. How should they be set? Does the end user have to be aware of all these issues just to install a board? A simple solution would be for the driver to call a BIOS routine that would poll all the MCA slots to determine how many LAN boards are installed. The driver would then set the threshold and arbitration levels appropriately for each board. Using this method the user would be far removed from the details of these specifics and a smooth installation would be insured.

At point "A" in *Figure 2* below, LAN0:3 and the ESDI controller request the bus. At point "B", 7.8 µs later the SCSI controller removes itself and an arbitration cycle begins with the other devices participating. It should be noted that if the bus-mastering SCSI controller IC is in the middle of a block transfer when it gets off, it will need to tell the target so it won't request more data transfers of it and the system any more. It does this by simply refusing to issue more acknowledges to the target after the REQ/ACK offset has been met (in synchronous mode). In this way the target won't be requesting the initiator until it has access to the system bus again. The effect is that the SCSI controller can be off the bus even during the middle of a block transfer. After the arbitration following this SCSI transfer, LAN0 will win due to it's higher priority. To determine system latency we will need to calculate the sum total of the occupancy times of all devices. If this latency is less than the maximum latency tolerance of all the LAN devices, proper bus access and performance levels can be expected. If not, FIFO overruns would occur, the situation we are trying to prevent and will show won't happen.



TL/F/11141–1

FIGURE 2. Initial DMA Sequence

With that, how long will LAN0 be on the bus? Since LAN0 didn't get the bus until point "C", 8.1 µs later, and the controller has been programmed for EMPTY/FILL mode, it will transfer the sum of the number of bytes determined by the FIFO threshold and the number of bytes accumulated from the network since the request was made. Let's call the "threshold" transfer time T_T and the transfer time for the accumulated bytes T_A. We will call the number of accumulated bytes simply "#". Since our threshold for LAN0 is 16 bytes, T_T will be the time it takes to transfer 16 bytes. T_A will be the time it takes to transfer the number of bytes accumulated since the request was made (8.1 µs), as well as T_T. So we have:

$$T_{TOT} = T_T + T_A$$

$$T_T = 16 \text{ Bytes} \left(\frac{1 \text{ Transfer}}{4 \text{ Bytes}}\right) 0.25\ \mu\text{s/Transfer} = 1.0\ \mu\text{s}.$$

$$\# = (8.1\ \mu\text{s} + 1.0\ \mu\text{s})/(0.8\ \mu\text{s/Byte})$$
$$= 11.375 \text{ Bytes Accumulated.}$$

8 bytes (two long words) will transfer with 3 bytes left in FIFO and 3 bits in serial/parallel converter. (The SONIC will transfer only long-word values to/from the FIFO).

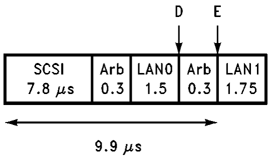
$$T_A = 8 \text{ Bytes} \left(\frac{1 \text{ Transfer}}{4 \text{ Bytes}}\right) 0.25\ \mu\text{s/Transfer} = 0.5\ \mu\text{s}.$$

$$T_{TOT} = 1.0\ \mu\text{s} + 0.5\ \mu\text{s} = 1.5\ \mu\text{s}.$$

Therefore the total transfer time for LAN0 is 1.5 µs. LAN0 will then request the bus again when it's FIFO threshold has been reached. Since there are 3 bytes left in FIFO and 3 bits in the serial/parallel converter,

$$T_{REQ} = (16 - 3 - \tfrac{3}{8} \text{ Bytes})\ (0.8\ \mu\text{s/Byte}) = 10.1\ \mu\text{s}.$$

So LAN0 will request the bus 10.1 µs later. It should be noticed that LAN0 (and LAN1 also) have a latency tolerance of 12.8 µs. This latency is more than adequate for the current latency of 8.1 µs.



TL/F/11141–2

FIGURE 3. Initial Latency for LAN1 Card

At point "D" LAN0 finished it's transfer and LAN1:3 and the ESDI controller arbitrate with LAN1 winning due to it's higher priority. Total bus occupancy for LAN1 will again be $T_{TOT} = T_T + T_A$.

$T_T = 1.0\ \mu s$ (because of the 16 byte transfer as calculated above).

$$\# = \frac{9.9\ \mu s + 1.0\ \mu s}{0.8\ \mu s/\text{Byte}} = 13.625 \text{ Bytes Accumulated.}$$

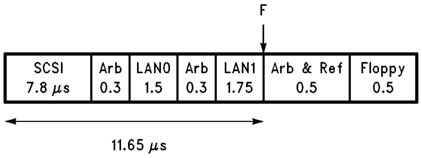
12 additional bytes (3 long words) will transfer with 1 byte remaining in the FIFO and 5 bits in serial/parallel converter.

$$T_A = 12 \text{ Bytes} \frac{0.25\ \mu s}{4 \text{ Bytes}} = 0.75\ \mu s$$

$$T_{TOT} = 1.0\ \mu s + 0.75\ \mu s = 1.75\ \mu s.$$

Therefore LAN1 will own the bus for 1.75 μs. Since LAN1's latency tolerance of 12.8 μs is greater than the current latency of 9.9 μs, it will be guaranteed access and no FIFO overruns will occur. LAN1 will then request the bus when it's FIFO threshold has again been reached. Since there is 1 byte left in the FIFO and 5 bits in the serial/parallel converter, the request time will be:

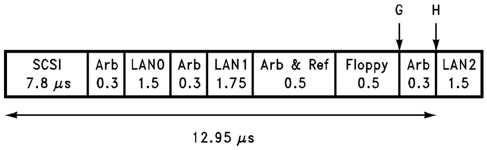
$$T_{REQ} = (16 - 1 - 5/8 \text{ Bytes})\ (0.8\ \mu s/\text{Byte}) = 11.5\ \mu s$$



TL/F/11141–3

FIGURE 4. Latency Till End of LAN1 Card Bus Occupancy Followed by Arbitration and Floppy Disk Access

At point "F" the SCSI controller, LAN0 and LAN1 have had their turn on the bus. At this point another arbitration will take place. Since the system needs to refresh memory, we will put in a refresh cycle now. This refresh will extend the arbitration by 200 ns, to a total of 500 ns. We also need to account for a floppy controller access. It is important for the floppy controller to gain access to the bus because if one of it's drives is a "floppy tape" and a byte was lost, the tape would have to stop, rewind, and re-read/write to that logical sector, taking a very bad performance hit. This situation needs to be prevented. We will assume that DMA channel 2 will win this arbitration and the floppy controller will transfer one byte, staying on the bus for approximately 500 ns. We now have:



TL/F/11141–4

FIGURE 5. Bus Latency Time for LAN2 Card

After the floppy access, LAN2:3 and the ESDI controller will arbitrate at point "G", with LAN2 winning and beginning to transfer at point "H". Since LAN2's latency tolerance is 19.2 μs and 12.95 μs is the current latency, there is 6.25 μs of margin left to guarantee proper access. How long will LAN2 stay on the bus?

$$T_{TOT} = T_T + T_A.$$

$$T_T = 0.5\ \mu s \text{ (for any 8 Byte Transfer)}$$

$$\# = (12.95\ \mu s + 0.5\ \mu s)\ (1 \text{ Byte}/0.8\ \mu s)$$

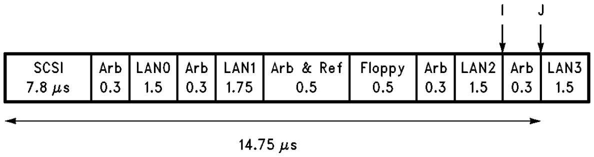
$$= 16.8125 \text{ Accumulated Bytes.}$$

The SONIC will then transfer the additional 16 bytes (4 long words) that were accumulated in the FIFO and keep the remaining 6.5 bits in the serial/parallel converter.

$T_A = 1.0\ \mu s$ (from a previous calculation for a 16 byte transfer)

$$T_{TOT} = 0.5\ \mu s + 1.0\ \mu s = 1.5\ \mu s.$$

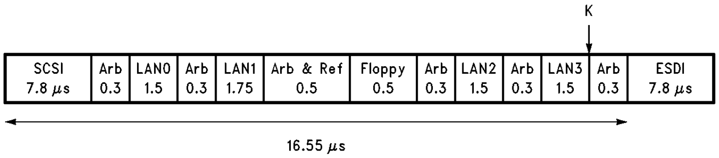
LAN2 will then re-arbitrate when it's FIFO has reached 8 bytes. This will be as shown in *Figure 6*.



TL/F/11141–5

FIGURE 6. Bus Latency Time for LAN3 Card

T_{REQ} = (8 − 6.5/8 Bytes) (0.8 μs/Bytes) = 5.75 μs later.



TL/F/11141–6

FIGURE 7. Total Bus Latency Time until Beginning of ESDI Drive Bus Access

At point "I" LAN2 is finished and LAN3 and the ESDI board will arbitrate with LAN3 winning. Since LAN3 has a latency tolerance of 19.2 μs and only 14.75 μs have occurred since LAN3 could have owned the bus, the latency margin of 4.45 μs is left over and a proper bus access has been guaranteed. LAN3 will then occupy the bus for:

$$T_{TOT} = T_T + T_A$$

T_T = 0.5 μs (from before for an 8 byte threshold)

# = (14.75 μs + 0.5 μs) (1 Byte/0.8 μs)

= 19.0625 Accumulated Bytes.

The SONIC will transfer 16 bytes (4 long words) with 3 bytes remaining in the FIFO and 0.5 bits in the serial to parallel converter.

T_A = 1.0 μs for a 16 byte transfer so we have

T_{TOT} = 0.5 μs + 1.0 μs = 1.5 μs.

LAN3 will then arbitrate again when its FIFO threshold of 8 bytes has been reached. This will be:

$$T_{REQ} = \left(\frac{8 - 3 - 0.5}{8}\right) (0.8\ \mu s/\text{Byte}) = 3.95\ \mu s$$

So LAN3 will request the bus again in 3.95 μs. At this point we have the following sequence of events:

At point "K", the ESDI controller will arbitrate and win and will stay on the bus for 7.8 μs. After winning the bus, the ESDI controller will deassert PREEMPT*. The SCSI controller can now assert PREEMPT* (because fairness has been enabled for it) to request the bus again since it has still more data to transfer.

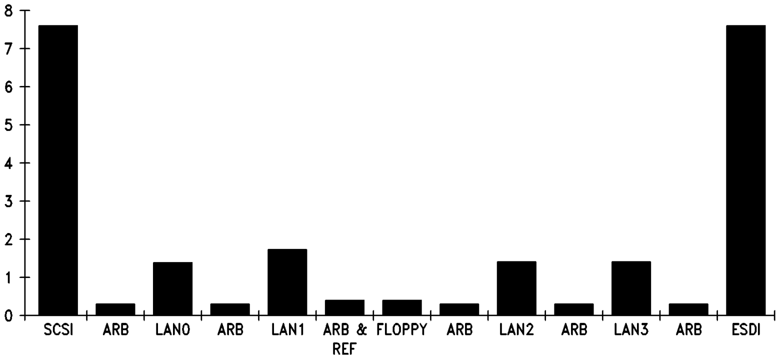
In all of the previous illustrations we showed all devices and their respective occupancy times and their relative sequence. The following graph visually shows how long all devices will own the bus relative to each other. It is quite apparent that due to the SONIC's and MCA's high speed DMA, the LAN controllers are on for a minimal amount of time. Streaming Mode MCA adapters would be on for half the time.

In this example we have taken a worst case scenario by assuming all the LAN boards and the ESDI board will request the bus simultaneously at the very beginning of the SCSI transfer period. We have shown that even in this situation all devices have accessed the MCA bus without error and with plenty of latency margin left over. Table IV summarizes these results.

TABLE IV. Accrued Latency

Device	Accrued System Latency (μs)	Device Latency Tolerance (μs)	Latency Margin (μs)
SCSI	0	(Note)	
LAN0	8.1	12.8	4.7
LAN1	9.9	12.8	2.9
REFRESH	11.65		
FLOPPY	12.15	(Note)	
LAN2	12.95	19.2	6.25
LAN3	14.75	19.2	4.45
ESDI	16.55	(Note)	

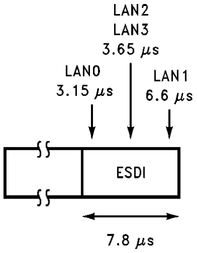
Note: These latencies are particular to the device in question.



TL/F/11141–8

FIGURE 8. Individual Bus Usage Times for All Bus Masters, and Arbitration Cycles

Since we are basing our calculation on this simultaneous request, what will happen when these LAN boards arbitrate again? Will this worst case scenario happen again? Based on our previous calculations the LAN boards will request again at different times. The following diagram shows when the LAN boards will arbitrate once more:



TL/F/11141–7

FIGURE 9

It can be seen now that starting with a worst case scenario as described above, the next set of LAN requests will be staggered apart throughout the ESDI transfer and our worst case scenario has all but disappeared, even after starting with it in the beginning. The LAN boards will still request and occupy the bus consecutively, however, they will now be on the bus for a shorter period of time. This is because the controller will get the bus sooner than in our worst case scenario; fewer bytes would have been accumulated in the FIFO since its threshold was reached hence a shorter transfer period. This means that other devices such as the CPU and mass storage controllers can have the bus sooner and occupy it longer than before. This equates to overall faster data throughput and more processing time for the CPU. It is up to the designer to determine when this worst case scenario will occur again, but it can be seen that the probabilities are exceptionally low that it will ever be repeated; however, if it did by properly setting arbitration and threshold levels and fairness options, the high performance of the SONIC can be readily achieved.

Since all devices have had a chance on the bus, what happens to the CPU during this worst case scenario? It has duties of its own such as protocol processing, updating descriptor lists, managing packets, etc. In the rare instance of this worst case scenario it wouldn't have immediate access to the bus. However, in nearly all the following accesses where the LAN accesses are staggered apart, there would be plenty of time for the CPU to access system memory.

One of the assumptions of this example is that no two consecutive transfers of 7.8 µs will occur in a row on the MCA bus when the LAN controllers are requesting it. The only way for this to happen was if there was a board which needed the bus immediately, and had a higher priority than the LAN boards and also would own the bus for a long period of time. However, a long bus occupancy time suggests a large buffer to hold all that data that is being transferred. A large buffer means it can tolerate longer latencies which means it can be set to a lower priority level, which effectively means this situation is avoided. Thus the LAN boards can effectively remain at the highest priority level and not be potentially locked out due to multiple, consecutive, 7.8 µs transfers, which won't happen.

A concern throughout this analysis may be bus efficiency. Since the SONIC transfers just a few bytes at a time, it will request the bus often causing the arbitration time to be a significant portion of the transfer cycle. However, because of the Ethernet transfer rate of 1.25 MB/s these requests won't be often. When compared to the transfer times of the SCSI and ESDI boards, these arbitration times are not too significant (see *Figure 8*) and won't occupy much bus bandwidth. With these lower thresholds and bursty transfers, these inefficiencies become apparent. However, the SONIC more than compensates in other areas.

The 20 MB/s transfer rate of the DMA allows for minimal time on the bus. With Streaming Mode MicroChannel, the bus occupancy can be further lowered by having a 40 MB/s data rate. By keeping the FIFO down to 32 bytes, the buffering of runt packets is eliminated. A larger FIFO may buffer many of these unwanted packets in a heavily loaded network and wastes valuable bandwidth. Also, the SONIC's buffer management structure has been designed for simplicity and performance.

With much of the performance bottleneck happening in the upper protocol layers, a very fast and efficient driver becomes a necessity. The SONIC's register oriented buffer management scheme makes this possible. Upating descriptor lists is simple and doesn't take much processor overhead. It is very efficient.

The on-board CAM can hold up to 16 different physical and multicast addresses. This allows supporting multiple protocols at the MAC level. By assigning a different physical address to each of the different protocols supported by the file server, protocol filtering can be done at a very low level, where it is much more efficient. To implement this with a controller that supports only one physical address would necessitate it to enter promiscuous mode, meaning that it would have to buffer every packet on the network. This would be a very great waste of system bandwidth.

Another way to improve efficiency would be to tie multiple SONICs together while maintaining a single MCA bus interface. The MREQ* and SMACK* pins on the SONIC allow it to be a slave to other devices, even other SONICs. By tying multiple SONICs together, they could be time multiplexed into one MCA time slot; this would have the advantage of requiring only one arbitration cycle for multiple controllers. Not only would the efficiency go up but costs would come down as multiple SONICs would share just one bus interface. In short, the SONIC provides an optimal balance to achieve exceptional performance at all levels where system performance is measured.

Determining Arbitration and Threshold Levels in a SONIC Based MicroChannel Adapter

AN-747

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor
Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: 1(800) 272-9959
TWX: (910) 339-9240

National Semiconductor
GmbH
Livry-Gargan-Str. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (81-41) 35-0
Telex: 527649
Fax: (81-41) 35-1

National Semiconductor
Japan Ltd.
Sumitomo Chemical
Engineering Center
Bldg. 7F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Ciba Prefecture 261
Tel: (043) 299-2300
Fax: (043) 299-2500

National Semiconductor
Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductores
Do Brazil Ltda.
Rue Deputado Lacorda Franco
120-3A
Sao Paulo-SP
Brazil 05418-000
Tel: (55-11) 212-5066
Telex: 391-1131931 NSBR BR
Fax: (55-11) 212-1181

National Semiconductor
(Australia) Pty, Ltd.
Building 16
Business Park Drive
Monash Business Park
Nottinghill, Melbourne
Victoria 3168 Australia
Tel: (3) 558-9999
Fax: (3) 558-9998

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and National reserves the right at any time without notice to change said circuitry and specifications.