Timing Analysis of Synchronous and Asynchronous Buses

ABSTRACT

This paper presents detailed examples of bus timing calculations for both synchronous and asynchronous busses, showing that bus throughput can be maximized by taking into account the characteristics and limitations of the transceiver technology being used. Based on these examples, a performance analysis of the currently available high speed bus interface technologies is made in terms of their maximum attainable transfer rate on both types of backplane busses. The results show that the use of a faster transceiver, as judged by its data sheet, doesn't necessarily result in a faster bus.

INTRODUCTION

In order to derive the highest possible throughput from a backplane bus, a careful analysis and optimization of timing parameters is essential. The maximum speed attainable at the physical level of the bus is a function of the transceiver technology, the electrical length of the bus, and the type of protocol, synchronous or asynchronous, being used. A clear understanding of the bus timing constraints lets the designer take best advantage of a given technology, such as TTL, ECL, or BTL (Backplane Transceiver Logic). Contrary to intuitive thinking, a faster transceiver will not always result in a faster bus. It can be shown through examples that greater bus transfer rates can be obtained by using specially designed bus transceivers, such as the BTL Trapezoidal, that at first glance may appear to be slower than the equivalent AS or FAST devices. These devices, in addition to improving bus bandwidth, also reduce crosstalk, ground noise, and system power requirements.

BUS PROPAGATION DELAY AND SETTLING TIME

Traditionally, system designers have used standard TTL devices to drive the backplane bus. Unfortunately, although TTL appears to provide fast rise and fall times, it cannot cleanly drive the capacitance of a loaded backplane or the resistance required for proper termination. BTL technology is a result of work that was done within the IEEE 896.1 Futurebus committee specifically to solve the problems of driving a backplane with transmission-line characteristics. By using a smaller voltage swing, lower capacitance drivers, and receivers with precision thresholds, BTL transceivers overcome the "bus driving problem."

Simply stated, the problem is one of driving a low impedance transmission line (*Figure 1* and *Figure 2*). The capacitive loading of a bus due to TTL transceivers reduces its impedance from an unloaded value of 60–100Ω to a fully loaded impedance of less than 20Ω. A properly matched termination resistance would therefore require a current of over 300 mA in order to cleanly drive a 3V nominal TTL swing! Since most TTL drivers cannot supply this current, they must depend on reflections to build up the bus voltage to a DC level. This results in a settling-time penalty of one or more bus round-trip propagation delays on every signal transition, or 35 ns on a typical 20" TTL bus.

The low output capacitance of BTL transceivers allows the total capacitive loading of a card in a backplane to be kept under 10 pF. This doubles the impedance of a loaded bus to almost 30Ω. BTL also specifies a reduced signal swing of 1V, which allows a properly terminated bus to be driven cleanly at under 75 mA. Consequently, there are no reflections, and the settling time is zero. A BTL driver can be guaranteed to cross the threshold of every receiver on the backplane with the incident edge of a signal wavefront.

The propagation delay of a bus is also a strong function of the capacitive loading. In the TTL case, the capacitive loading increases the signal propagation delay by a factor of 3 to 5 over an unloaded bus. In a 20" bus, BTL can reduce this delay from a value of 13 ns in the TTL case to less than 9 ns, increasing the potential bus bandwidth significantly.

SYNCHRONOUS BUS TIMING

For our first example, let's consider burst data transfers on a synchronous bus. In many backplane systems, burst transfers provide the highest performance, because the overhead associated with the address cycle can be spread out over a number of data cycles. Although other types of transactions may be more complex and require more time (clock cycles), it is likely that many systems will be optimized for burst transfers.

In this example, we are making some simplifying assumptions which ignore some of the penalties associated with a general-purpose synchronous bus. One of these is that the entire interface is synchronized to the bus clock. In general, each card in a backplane will be running off of its own internal high-speed clock. This results in resynchronization metastability problems at both the master and slave interfaces, as well as a clock latency penalty of typically 50% of the clock period. We are also ignoring the return of status from the slave on each data transfer, by assuming all status can be generated before the data is clocked. This would not be true, for example, if parity had to be verified before the next data transfer could take place.

Clock Skew

In this example, the system clock is being distributed to each board through a clock line on the backplane. Since the clock line is being driven from a single point, the loaded capacitance on it is considerably less than on most other lines, and the settling time is typically zero, even in a TTL-based backplane. Due to the finite propagation delay across the bus, however, the clock edge still arrives at each board at different times, creating a relative edge inaccuracy commonly referred to as clock skew.

The worst-case skew can be cut in half by locating the clock source centrally on the backplane, rather than at one end. Additional clock skew will be introduced by the propagation delay differences in the receiver and logic gates that process the clock signal between boards. For a typical 20" TTL bus, with the clock driver located at the midpoint, total skew can easily exceed 10 ns; in our case, 5 ns for the bus, plus 7 ns for the receiver and a transparent latch used to implement bus wait states.

TTL

data sheet

~3V

real life

2.0 – 2.4

1.5V

0.6 – 0.8

0V

Driver
Prop. Delay
5 ns

Settling Time
35 ns

AN009633-1

BTL (Trapezoidal)

2.1V

1.6
1.55V
1.5

1V

Driver
Prop. Delay
15 ns

Zero
Settling Time

AN009633-2
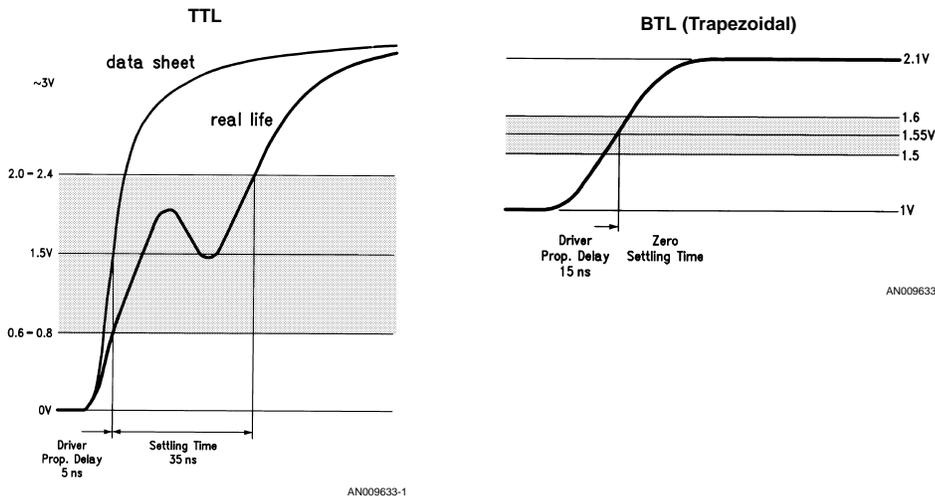
FIGURE 1. Settling Times

Synchronous Data Transfer Timing

In this example (*Figure 3*), the worst case data propagation delay from the master to the slave is simply the sum of the delays of the individual components of the data path. This path travels through the master's edge-triggered flip-flop and bus driver, across the length of the bus, and then through the slave's bus receiver and flip-flop, where the incoming data is latched. However, because this is a synchronous system, the data can be "pipelined" to some extent within the intervening logic. This means that the minimum clock cycle possible under this configuration is the sum of the logic skews, plus the maximum bus propagation delay, the set-up and hold times of the receiver, and the clock skew (*Figure 4*).

The advantage of a synchronous system is that the absolute timing requirements are set by the clock; the entire system can be optimized with this constraint in mind. This can become a disadvantage as technology advances beyond the point at which the synchronous bus was designed. A synchronous system must be continually redesigned for higher clock rates in order to take advantage of improvements in technology. Synchronous busses are therefore more suited to specific applications than to general-purpose, extended lifespan products.

Synchronous Timing Calculations

The first set of calculations assumes a TTL bus with AS transceivers and logic. As can be seen, the bus settling time overwhelms all the other skews and delays in the system. The upper limit of a discrete TTL synchronous bus implementation is roughly 15 MT (megatransfers/second). No particular advantage is gained by using FAST devices because, while the maximum propagation delays specified for that family are shorter than for AS, the maximum skews are generally greater. The effect of skew specifications is another subtlety of system performance analysis.

Two types of BTL transceivers are currently available, the BTL Trapezoidal and the BTL Turbo. The Trapezoidal transceivers have controlled rise and fall times on their drivers of 6 ns (nominal) to reduce crosstalk interference and switching noise within the backplane. In addition, the receivers incorporate crosstalk filters that practically eliminate far-end crosstalk problems on the bus. The Turbo transceivers eliminate these Trapezoidal features, but are much faster as a result. Switching noise problems are overcome by the use of individual ground return lines for each driver. Stripline backplane construction and careful layout techniques are required to minimize crosstalk.

Although the BTL Trapezoidal transceiver delays are much greater than those of the TTL devices, the absence of settling time results in a smaller overall clock cycle time. A maximum transfer rate of 18 MT becomes possible. When the Turbo devices are used, system throughput increases to 24 MT in this discrete implementation.
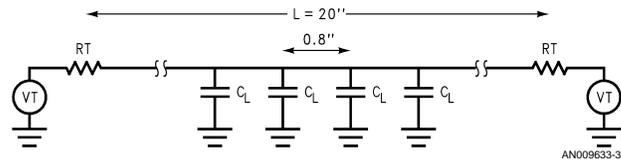
C_L (TTL) ≈ 25 pF/0.8" = 375 pF/ft.

C_L (BTL) ≈ 10 pF/0.8" = 150 pF/ft.

Z_O ≈ 75Ω Unloaded Bus Impedance

C_O ≈ 20 pF/ft. Distributed Capacitance of Unloaded Bus

T_O ≈ 1.8 ns/ft. Unloaded Bus Propagation Delay

$Z_L = Z_O / \sqrt{1 + (C_L/C_O)}$

Loaded Bus Impedance

$T_L = L \times T_O \times \sqrt{1 + (C_L/C_O)}$

Loaded Propagation Delay

T_L (TTL) ≈ 13.3 ns	TL (BTL) ≈ 8.75 ns

FIGURE 2. Effects of Capacitive Loading
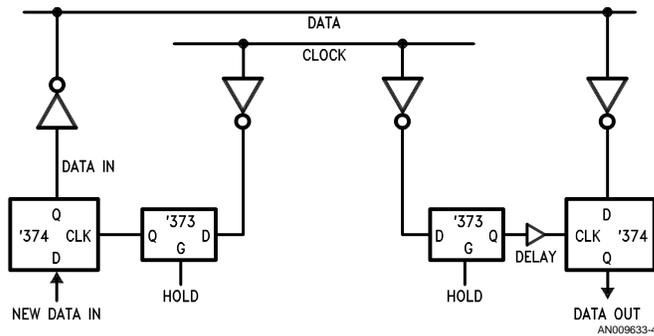
FIGURE 3. Synchronous Bus Logic for Burst Data Transfers

	TTL AS	BTL Trap	BTL Turbo
1) Max '374 Skew	5.0	5.0	5.0
2) Max Bus Driver Skew	4.5	10.0	5.0
3) Max Bus Delay	35.0	9.0	9.0
4) Max Bus Receiver Skew	4.5	13.0	6.0
5) Max '374 Setup and Hold	5.0	5.0	5.0
6) Max Clock Skew	12.0	12.0	12.0
TOTAL (ns)	66.0	54.0	42.0
MTransfers/second	18.5	18.5	23.8

FIGURE 4. Synchronous Burst Data Transfer Timing
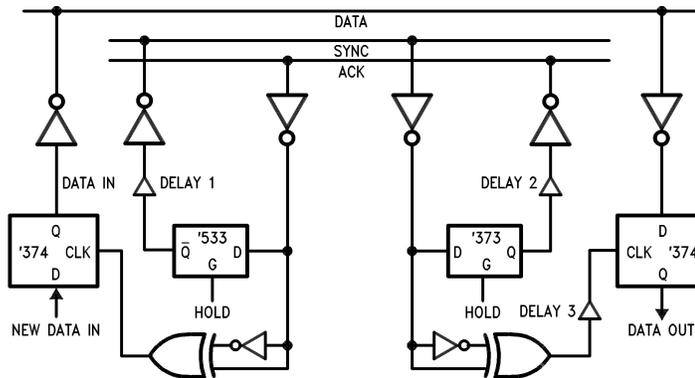
FIGURE 5. Asynchronous Bus Logic for Burst Data Transfers

DELAY 1	TTL AS	BTL Trap	BTL Turbo	
1) Max XOR Delay	6.5	6.5	6.5	
2) Max '374 Delay	9.0	9.0	9.0	
3) Max Data Driver Delay	6.5	15.0	7.0	
4) <Min '533 Delay>	−4.0	−4.0	−4.0	
5) <Min Sync Driver Delay>	−2.0	−5.0	−2.0	
TOTAL (ns)	16.0	21.5	16.5	

DELAY 2	TTL AS	BTL Trap	BTL Turbo	
1) Max XOR Delay	6.5	6.5	6.5	
2) Max '374 Hold Time	3.0	3.0	3.0	
3) Delay 3	5.5	14.0	7.0	
4) <Min '373 Delay>	−3.5	−3.5	−3.5	
5) <Min Ack Driver Delay>	−2.0	−5.0	−2.0	
5) <Min Data Receiver Delay>	−2.0	−5.0	−2.0	
TOTAL (ns)	7.5	10.0	9.0	

DELAY 3	TTL AS	BTL Trap	BTL Turbo	
1) Max Data Receiver Delay	6.5	18.0	8.0	
2) Max '374 Setup Time	2.0	2.0	2.0	
4) <Min Sync Receiver Delay>	−2.0	−5.0	−2.0	
5) <Min XOR Delay>	−1.0	−1.0	−1.0	
TOTAL (ns)	5.5	14.0	7.0	

FIGURE 6. Asynchronous Bus Logic Delay Calculations

TTL-AS	1	2	3	4	5	6	7	8	9	10	133 ns

BTL - Trap.	1	2	3	4	5	6	7	8	9	10	130 ns

BTL - Turbo	1	2	3	4	5	6	7	8	9	10	88 ns

	TTL AS	BTL Trap	BTL Turbo
1) Max Ack Receiver Delay	6.5	18.0	8.0
2) Max '533 Delay	7.5	7.5	7.5
3) Delay 1	16.0	21.5	16.5
4) Max Sync Driver Delay	6.5	15.0	7.0
5) Max Bus Delay + Skew	35.0	10.0	10.0
6) Max Sync Receiver Delay	6.5	18.0	8.0
7) Max '373 Delay	6.0	6.0	6.0
8) Delay 2	7.5	10.0	9.0
9) Max Ack Driver Delay	6.5	15.0	7.0
10) Max Bus Delay	35.0	9.0	9.0
TOTAL (ns)	133.0	130.0	88.0
MTransfers/second	7.5	7.7	11.4

FIGURE 7. Asynchronous Burst Data Transfer Timing
(Worst Case)

The largest cycle time delay in the final BTL Turbo example is clock skew. Bus skews can be reduced by distributing the clock to each board independently, using a dedicated trace on the backplane such that all lines are of equal length. This makes the clock propagation delay from the driver to each board the same, and thus practically eliminates the bus skew. In addition, better tolerances on driver, receiver, and logic propagation delays (smaller skews) will improve both the clock skew and the effect of transceiver delays on the cycle time.

ASYNCHRONOUS BUS TIMING

Our second example is also of a burst transfer, but this time using asynchronous bus timing. In this system, the master issues a strobe along with the data, and waits for an acknowledgement from the slave before removing the current data from the bus lines. All timing is controlled by the two participants in the data transfer. (Once again, we are assuming that new status does not have to be generated on each data transfer.)

The greatest advantage of an asynchronous bus protocol is its ability to adapt the speed of the bus to the speed of any two communicating boards. The most flexibility is achieved when no technology dependencies are introduced into the protocol. Unlike a synchronous system, where every board is designed with the same timing constraints in mind, a technology-independent module is designed with no assumptions about the timing of the rest of the system. Instead, each transmitting board simply guarantees that its data is valid on the bus at least zero nanoseconds before it issues its synchronization signal, and each receiving board is responsible for ensuring that its data has been successfully latched before issuing an acknowledge. The protocol itself imposes no artificial set-up or hold time limitations.

The result of this lack of timing constraints is that a board built today, using today's technology, is guaranteed to work in a system designed perhaps twenty years from now. That system will be forced to slow down whenever necessary to accommodate the greater internal delays and skews of the older module. However, if two future modules are communicating, they will transfer data at the maximum rate allowed by the future technology. The new IEEE Futurebus standard implements this type of protocol.

ASYNCHRONOUS DATA TRANSFER TIMING

The requirement that boards generate their own data synchronization and acknowledge signals, and the likelihood of zero set-up and hold times on the bus, make the timing of the asynchronous system more complicated than the previous example (*Figure 5*). Also, we are maximizing the performance of the sync/ack handshake by transferring data on each signal transition. This is known as a two-edge handshake.

On the master side, the board must guarantee that its data is valid on the bus before issuing the synchronization signal. This means that a delay must be inserted in the sync signal path (Delay 1) which includes the maximum propagation delays through the XOR clock generation circuit, edge-triggered flip-flop, and data bus driver. This is excessive, however, because the minimum delays through the sync latch and bus driver can be subtracted (*Figure 6*).

On the slave side, delays are required to guarantee that both the set-up and hold time specifications of the data latch are met. The set-up time delay (Delay 3) ensures that the sync signal, which may have minimum propagation delays through the sync bus receiver and XOR clock generator, arrives at the edge-triggered data flip-flop a set-up time after the data, which may have a maximum delay through the data bus receiver. The hold time delay (Delay 2) ensures that the data remains at the data flip-flop a hold time after the sync signal, which this time may have a maximum propagation delay through the XOR and the set-up time delay element

just introduced. Since the removal of data is controlled by the ack signal, the hold time delay can be reduced by the minimum delays through the ack latch and bus driver, and the minimum propagation delay of the data bus receiver.

| TTL-AS | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 109 ns |

| BTL - Trap. | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 60 ns |

| BTL-Turbo | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 42 ns |

	TTL AS	BTL Trap	BTL Turbo
1) Min Ack Receiver Delay	2.0	5.0	2.0
2) Min '533 Delay	4.0	4.0	4.0
3) Delay 1	16.0	21.5	16.5
4) Min Sync Driver Delay	2.0	5.0	2.0
5) Min Bus Delay + Skew	35.0	1.0	1.0
6) Min Sync Receiver Delay	2.0	5.0	2.0
7) Min '373 Delay	3.5	3.5	3.5
8) Delay 2	7.5	10.0	9.0
9) Min Ack Driver Delay	2.0	5.0	2.0
10) Min Bus Delay	35.0	0.0	0.0
TOTAL (ns)	109.0	60.0	42.0
MTransfers/second	9.2	16.7	23.8

FIGURE 8. Asynchronous Burst Data Transfer Timing
(Best Case)

This is all very confusing at first, but these delay elements now in place in our circuit guarantee the receiver set-up and hold time requirements while maintaining the technology independence of the bus protocol. Now we can calculate the burst data transfer rate on this asynchronous bus.

The critical path is now the sync/ack handshake. The circuit delays are in place to make sure that data is transferred successfully. To calculate the transfer rate, simply add up all the propagation delays through the sync/ack loop (*Figure 7* and *Figure 8* on the master, the ack receiver, the sync latch, Delay 1, and the sync driver; a bus propagation delay; on the slave, the sync receiver, the ack latch, Delay 2, and the ack driver; and another bus propagation delay.

Should you use worst-case values throughout your evaluation? The beauty of a technology-independent asynchronous protocol is that is will adapt to the speed of the individual logic elements in the sync/ack handshake path. If all the devices happen to have worst-case characteristics, then yes; if they are all fast parts, however, then data transfer will take place under best-case conditions. Both calculations are included, providing the expected operating range of the circuit.

ASYNCHRONOUS TIMING CALCULATIONS

Once again, the TTL design is overwhelmed by the settling time of the bus. Since the sync/ack signal pair are acting as clocks in this system, glitches that may occur during the signal settling time are intolerable. This means that the 35 ns bus settling time must be hard-wired into the receiver logic, and cannot be reduced under best-case conditions. The per-

formance of an asynchronous TTL backplane, from 7.5 to 9.2 MT, cannot approach that of a similar synchronous backplane.

The BTL Trapezoidal system has very similar performance to a TTL backplane under worst-case conditions. However, because there is no settling time penalty associated with BTL signals, the effect of improvements in device operation have a far more pronounced effect. In the best case, the performance is close to that of the equivalent synchronous system. Also, since the bus signal propagation delay is a function only of the distance between the two boards, modules placed in adjacent slots will experience almost no backplane delays.

A BTL Turbo board benefits from the same clean electrical environment that a Trapezoidal one does, except with a 40–50% overall improvement in performance. In the best case, the performance is the same as that of the equivalent synchronous system. Of course, as device parameters improve, with lower propagation delays and skews, the performance of the asynchronous system will continue to improve. The largest reductions in the transfer cycle time will come as interfaces for asynchronous busses such as Futurebus are integrated onto a single piece of silicon, where skews and delays can be more tightly controlled.

CONCLUSION

The use of transceivers designed specifically for the transmission-line environment typical in today's high-speed backplanes provides advantages in both the performance

and electrical integrity of a system. The advantages of BTL only become obvious after a careful analysis of data transfer timing considerations. The Trapezoidal and Turbo options provide a designer with the opportunity to make the appropriate application-dependent cost/performance tradeoffs. A sometimes controversial issue is the appropriateness of a synchronous versus an asynchronous design. The former will usually provide an immediate performance advantage in a fully synchronized environment, but a carefully-designed general-purpose asynchronous protocol will often have a longer useful product life.

REFERENCES

1. R.V. Balakrishnan, "The Proposed IEEE 896 Futurebus — A Solution to the Bus Driving Problem," *IEEE Micro,* August 1984.

2. R.V. Balakrishnan, "Physical Layer Timing Design Considerations for High Speed Backplane Busses," Buscon/87 West, January 1987.

3. D.B. Gustavson and J. Theus, "Wire-OR Logic on Transmission Lines," IEEE Micro, June 1983, pp. 51–55.

4. *Interface Databook,* National Semiconductor Corp., Santa Clara, CA, 1986.

5. *ALS/AS Logic Databook,* National Semiconductor Corp., Santa Clara, CA, 1987.

6. *IEEE P896 D7.5a,* "Futurebus: A Bus Standard for Multiprocessing Architectures", IEEE, June 1987.

TABLE 1. Device Parameters

Device	Parameter (Transition)	Minimum Prop. Delay	Maximum Prop. Delay	Maximum Skew	Setup/Hold
DM74AS374	LH	3.0	8.0	5.0	**2.0/3.0**
Edge-Triggered Flip-Flop	HL	4.0	**9.0**	**5.0**	
DM74AS373	LH	**3.5**	**6.0**	**2.5**	2.0/3.0
Transparent Latch	HL	**3.5**	**6.0**	**2.5**	
DM74AS533	LH	**4.0**	**7.5**	**3.5**	2.0/3.0
Inverting Transparent Latch	HL	4.0	7.0	3.0	
DM74AS86	Other Input L	2.0	**6.5**	4.5	
2-Input XOR	Other Input H	**1.0**	6.0	5.0	
DM74AS240	LH	2.0	6.5	**4.5**	
Bus Driver/Receiver	HL	2.0	5.7	3.7	
DM74AS242	LH	**2.0**	**6.5**	**4.5**	
Bus Transceiver	HL	2.0	5.7	3.7	
DS3896	Rx	**5.0**	**18.0**	**13.0**	
BTL Trapezoidal Transceiver	Tx	**5.0**	**15.0**	**10.0**	
DS3893	Rx	**2.0**	**8.0**	**6.0**	
BTL Turbo Transceiver	Tx	**2.0**	**7.0**	**5.0**	

Note: Values in boldface are those used in the preceding calculations.
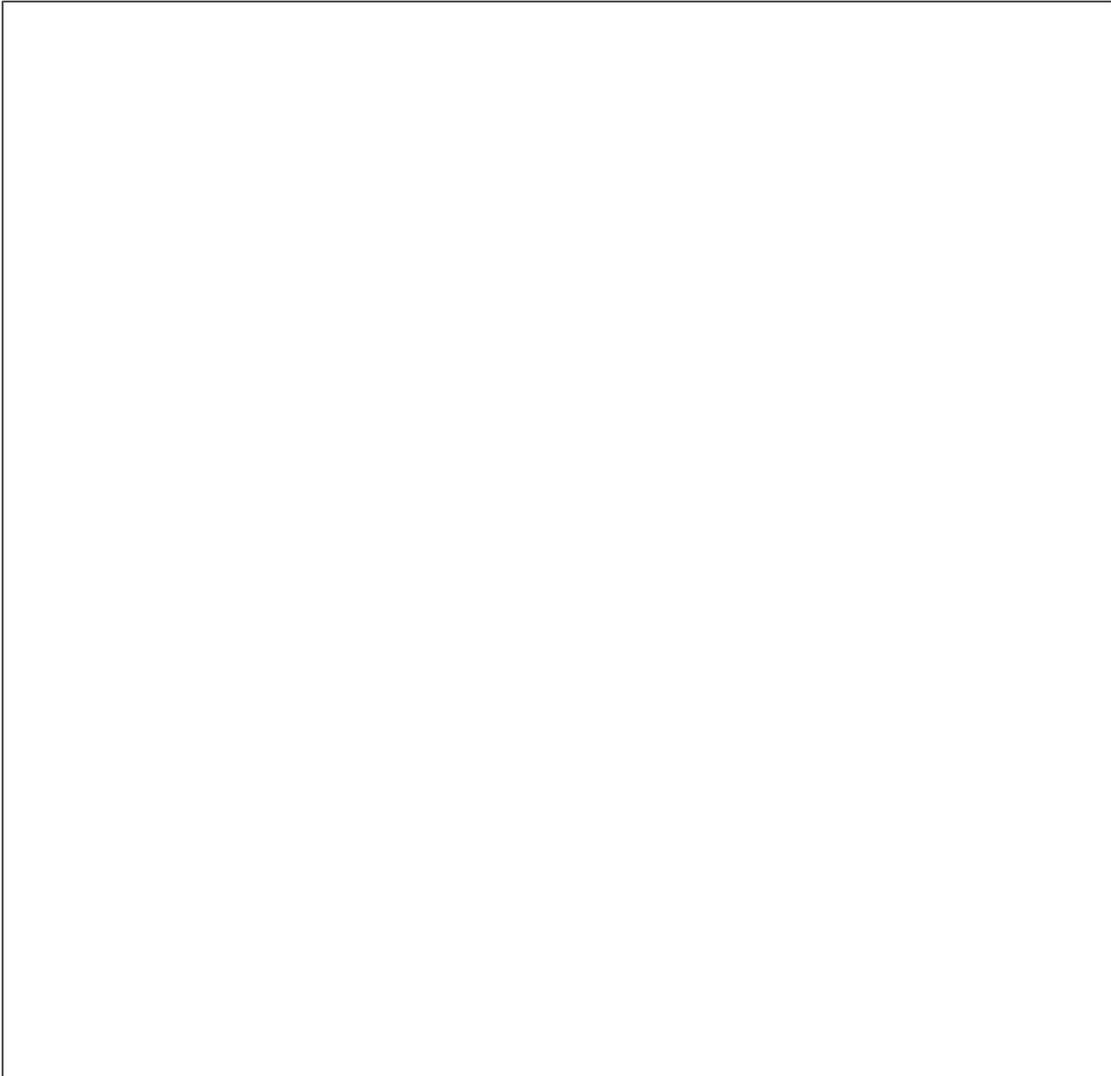
LIFE SUPPORT POLICY

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.