TMS320C671x/TMS320C621x EDMA Performance Data

Jamon Bowen
Jeffrey Ward

TMS320C6000 Architecture

ABSTRACT

The enhanced DMA (EDMA) controller of the TMS320C621x™/TMS320C671x™ devices is a highly efficient data transfer engine, capable of maintaining transfers at up to 1800 MB/sec at a 225 MHz CPU clock frequency. This document details actual bandwidth achieved under various operating conditions. Ideal transfer bandwidth is explored in *TMS320C6000 EDMA IO Scheduling and Performance* (SPRAA00).

Contents

List of Figures

List of Tables

1 Introduction

The enhanced DMA (EDMA) controller of the TMS320C671x/ TMS320C621x DSP is a highly efficient data transfer engine, capable of handling up to 8 bytes per EDMA cycle, resulting in 1800 Mbytes per second of total data throughput at a CPU rate of 225 MHz. EDMA performs all data movement between the on-chip level-two (L2) memory, external memory (connected to the device through an external memory interface, or EMIF), and the device peripherals. These data transfers include CPU-initiated and event-triggered transfers, master peripheral accesses, cache servicing, and non-cacheable memory accesses. The EDMA architecture has many features designed to facilitate multiple high-speed data transfers simultaneously. With a working knowledge of this architecture and the ways which data transfers interact, it is possible to create an efficient system and maximize the bandwidth utilization of the EDMA.

This document aims to give designers a basis for estimating system performance. Most of the tests operate under best case situations – no other DSP interference, to measure the maximum throughput that can be obtained. Two application notes, *TMS320C621x/TMS320C671x EDMA Architecture* (SPRA996) and *TMS320C6000 EDMA IO Scheduling and Performance* (SPRAA00), augment this document by describing ways that interference from other DSP activities can inhibit performance, and methods to mitigate interference. The transfers described in this document serve as a sample of interesting or typical performance conditions. The performance of other configurations is neglected in the interest of brevity and can generally be inferred from the examples given. When estimating other configurations performance, it is important to remember that the CPU to EMIF clock ratio is more important than the absolute level of these clocks. This document provides a good source of actual performance values to estimate the performance of a system.

2 EDMA Performance Data

Transfer performance is based on time. It is necessary to define quantities of time based on transfer events. Figure 1 is a simplified schematic of an example transfer in the EDMA showing the important time intervals. The time intervals are explained in detail in *TMS320C6000 EDMA IO Scheduling and Performance* (SPRAA00).

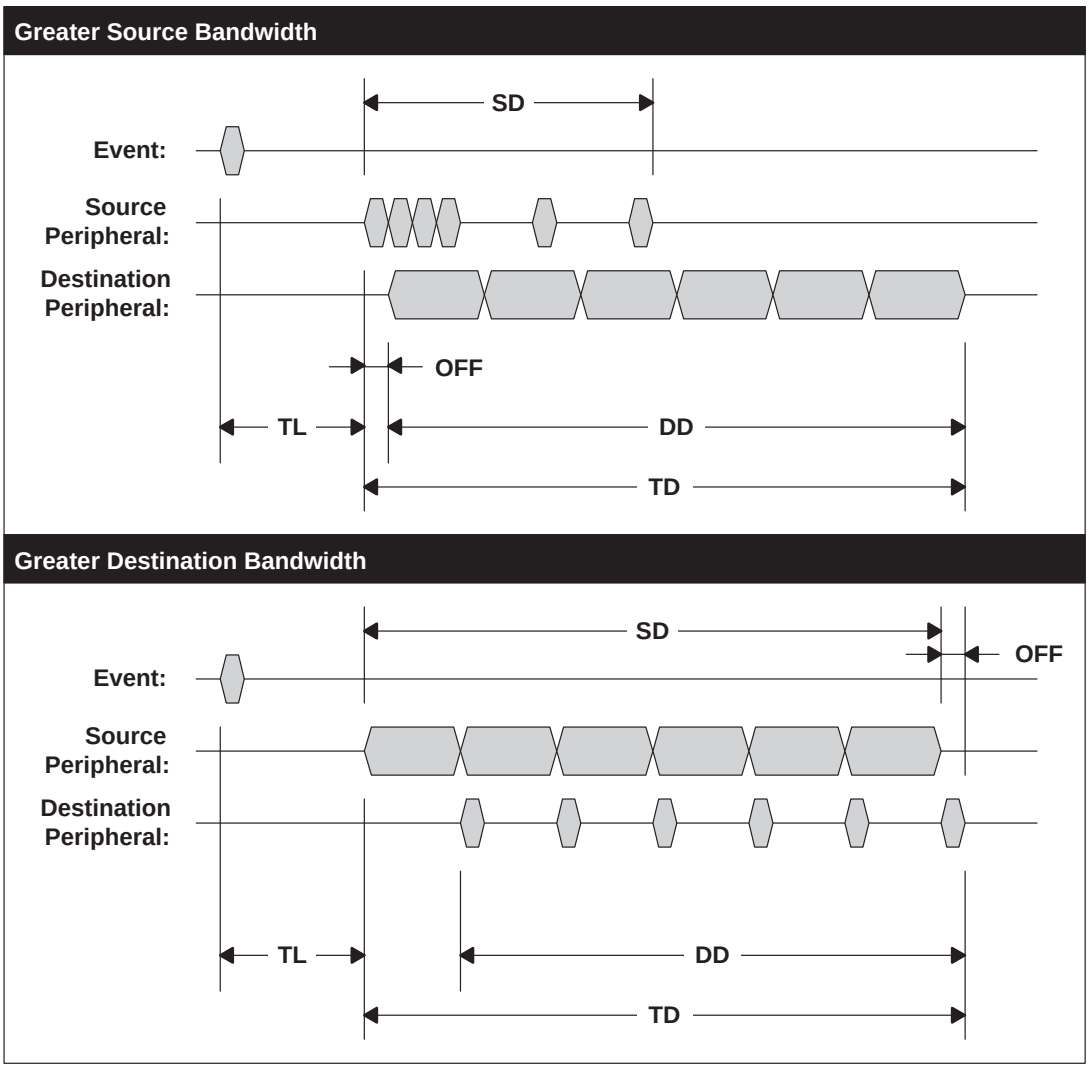

Figure 1. Transfer Time Intervals

- **TL** is transfer latency, defined as the time from event to the beginning of transfer at the source peripheral.
- **SD** is source duration, or the interval of time during which the source peripheral reads data.
- **DD** is destination duration, or the interval of time during which the destination peripheral writes data.
- **TD** is transfer duration, defined as the time from the first read at the source to the last write at the destination.
- **OFF** is the source/destination offset.

Transfer bandwidth is limited by the lesser of the source or destination port bandwidth. The maximum speed the transfer could achieve is equal to the bandwidth of the limiting port. The transfer bandwidth is found by taking total bytes transferred and dividing it by the greater of the source duration (SD) or destination duration (DD). The transfer will never complete within the time defined by the transfer bandwidth, however, due to the offset and latency in the transmission. The offset and initial latency are variable and depend on a number of factors, so they are not considered when calculating transfer bandwidth. It is important to remember that these two values are generally a bigger concern for short transfers and need to be included when scheduling EDMA traffic in a system. See *TMS320C6000 EDMA I/O Scheduling and Performance* (SPRAA00) for details.

2.1 Transfer Latency

The EDMA ideal transfer latency (no interference from other transfers) is based on the following:

- If the source port is EMIF, transfer latency is approximately 20 CPU cycles plus 3 EMIF cycles. For SDRAM, there may be an additional delay due to SDRAM paging mechanisms.
- If the source port is L2, transfer latency is approximately 21 CPU cycles.
- If the source port is an on-chip peripheral (such as the McBSP), the transfer latency is approximately 25 CPU cycles.

These values vary based on the type of source peripheral and readiness of the source and destination ports and peripherals. Also, SDRAM has page open/close overhead which can cause delays in transfers that tend to increase with element count. SBSRAM, which isn't paged, is used for most of the tests in this section to eliminate the uncertainty that page opening and closing brings.

Table 1 shows data taken on a TMS320C6713 CPU. The ideal and actual latencies are shown for comparison. No interfering transfer requests (TRs) are present. In this case, the SDRAM page opening delay appears to be about 2 EMIF cycles.

Table 1. Transfer Latency

CPU Frequency (MHz)	EMIF Frequency (MHz)	Source Peripheral	Expected Transfer Latency		Actual Transfer Latency	
			nS	CPU cycles†	nS	CPU cycles†
225	100	EMIF (SBSRAM)	118.9	26.8	117	26.3
225	66	EMIF (SBSRAM)	134.3	30.2	124	27.9
225	100	EMIF (SDRAM)	138.9	31.3	134	30.2
225	66	EMIF (SDRAM)	164.6	37.0	160	36.0
150	100	EMIF (SBSRAM)	163.3	24.5	158	23.7
150	66	EMIF (SBSRAM)	178.8	26.8	164	24.6
150	100	EMIF (SDRAM)	183.3	27.5	181	27.2
150	66	EMIF (SDRAM)	209.1	31.4	205	30.8
Don't care	Don't care	L2 Memory	–	21	–	21
Don't care	Don't care	McBSP	–	25	–	25

† Note that CPU cycle period is defined by the CPU frequency shown.

In addition to transfer latency there is another factor that influences overall transfer time: the source/destination offset. This value depends on the source and destination, the default burst size of the source/destination, and the transfer size. Calculating this value is discussed in *TMS320C6000 EDMA IO Scheduling* (SPRAA00).

2.2 Bandwidth vs. Transfer Flexibility

QDMA and EDMA channel parameters allow many different transfer configurations. Most typical transfers burst properly, and memory bandwidth is fully utilized. However, in some less common configurations, transfers are unable to burst, and performance is affected. To properly design a system, it is important to know which configurations offer the best performance for high speed operations, and which must trade off throughput for flexibility. These considerations are especially important for long memory transfers. Single element transfer performance is latency dominated and is unaffected by these conditions.

2.2.1 Element Size Considerations

To make full utilization of bandwidth in the transfer engine, it is important to fully utilize the bus width available and allow for data bursting. It is best to use 32-bit element size whenever possible. Setting element size in the options register (OPT) to 32 bits will cause a transfer to make full use of the available bus width (whether it is 32 or 64 bits wide) as well as allow the transfer to burst data in multi-element blocks. Setting the element size to less than 32-bits causes the EDMA to mask out the remaining data, wasting bandwidth.

Table 2 shows performance data from a TMS320C6713 DSP running at 225 MHz, transferring 4, 16, and 128 bytes from/to L2 to/from SBSRAM using an EDMA (or equivalently, a QDMA) channel. The SBSRAM is a 32-bit, 100 MHz memory with an ideal bandwidth of 400 Mb/sec. The transfer is performed with 8, 16, and 32-bit element sizes. Note that the 32-bit element size most efficiently uses available bandwidth.

Table 2. Bus Width Utilization

Element Size	Total Data (bytes)	Read Access		Write Access	
		Transfer Bandwidth (Mb/sec)	SBSRAM Bandwidth Utilization	Transfer Bandwidth (Mb/sec)	SBSRAM Bandwidth Utilization
32-bit	4†	400	100%	400.0	100.0%
32-bit	16	320	80%	307.7	76.9%
32-bit	128	376.5	94.1%	390.2	97.6%
16-bit	4	200	50.0%	50.0	12.5%
16-bit	16	64	16.0%	42.1	10.5%
16-bit	128	58	14.5%	40.3	10.1%
8-bit	4	40	10%	30.3	7.6%
8-bit	16	30.8	7.7%	22.0	5.5%
8-bit	128	28.8	7.2%	20.2	5.1%

† Note that a 4-byte (32 bit) transfer to a 64-bit SBSRAM cannot fully utilize bandwidth.

Non-programmable accesses (cache and master peripheral) are hardware programmed to fully utilize bus width.

2.2.2 Update Modes: Increment, Decrement, Index, and Fixed

The source update mode (SUM) and the destination update mode (DUM) help define how the address pointer is updated throughout a transfer. For information on update modes, see *TMS320C6000 DSP EDMA Controller Reference Guide* (SPRU234).

Table 3 shows performance data from a TMS320C6713 DSP running at 225 MHz, transferring 128 bytes from L2 to SBSRAM. The SBSRAM is a 32-bit, 100 MHz memory with an ideal bandwidth of 400 Mb/sec. The source and destination index modes are varied.

Table 3. Update Mode Effects (L2 to EMIF example)

SUM	DUM	External Memory Transfer Duration (CPU cycles)	Transfer Bandwidth (Mb/sec)	SBSRAM Bandwidth Utilization
L2: Inc, Dec, or Fixed	EMIF: Increment	76.5	376.5	94.1%
L2: Inc, Dec, or Fixed	EMIF: Fixed	76.5	376.5	94.1%
L2: Inc, Dec, or Fixed	EMIF: Decrement	96.8	297.7	74.4%
L2: Inc, Dec, or Fixed	EMIF: Index	245.7	117.2	29.3%
L2: Index	EMIF: Increment	92.3	312.2	78.1%
L2: Index	EMIF: Fixed	90.9	316.8	79.2%
L2: Index	EMIF: Decrement	139.5	206.5	51.6%
L2: Index	EMIF: Index	245.3	117.4	29.4%
EMIF: Inc, or Fixed	L2: Inc, Dec, or Fixed	74.25	387.9	97.0%
EMIF: Inc, or Fixed	L2: Index	78.75	365.7	91.4%
EMIF: Decrement	L2: Inc, Dec, or Fixed	83.7	344.1	86.0%
EMIF: Decrement	L2: Index	92.7	310.7	77.7%
EMIF: Index	L2: Any	355.5	81.0	20.3%

Note that in this case, the L2 update mode generally doesn't affect transfer bandwidth (except under the worst case where the update mode is index). This is because L2 memory has much greater bandwidth than the destination. At lower clock ratios this will not be the case, and decrement and index update modes will lower performance.

2.2.3 2−D Transfers − Element Count

Two-dimensional transfers offer designers a great deal of flexibility. Two-dimensional transfers allow a complex memory transfer (such as extracting a sub-region from an image in memory) to be submitted to the EDMA easily and accomplished automatically. See the EDMA section of the *TMS320C6000 Peripherals Reference Guide* (SPRU190) for more information on two-dimensional EDMA transfers.

The EDMA and the EMIF are optimized to facilitate large transfers in contiguous bursts. This is optimal for most EDMA traffic, but the performance of two-dimensional EDMA transfers with small element and frame counts suffer as a result. Smaller element counts experience the greatest performance degradation. The following charts summarize this trend; 1 to 128 32-bit words were transferred to/from an external SDRAM running at 100MHz, using various frame/element counts to achieve the total number of words. Note that write transfers with only 1 frame complete in one burst, and therefore achieve maximum transfer bandwidth, which then degrades for multiple frame transfers. This data is collected from the pin side of the transfer; adding the initial latency for TR submission will increase the transfer time for short transfers significantly.

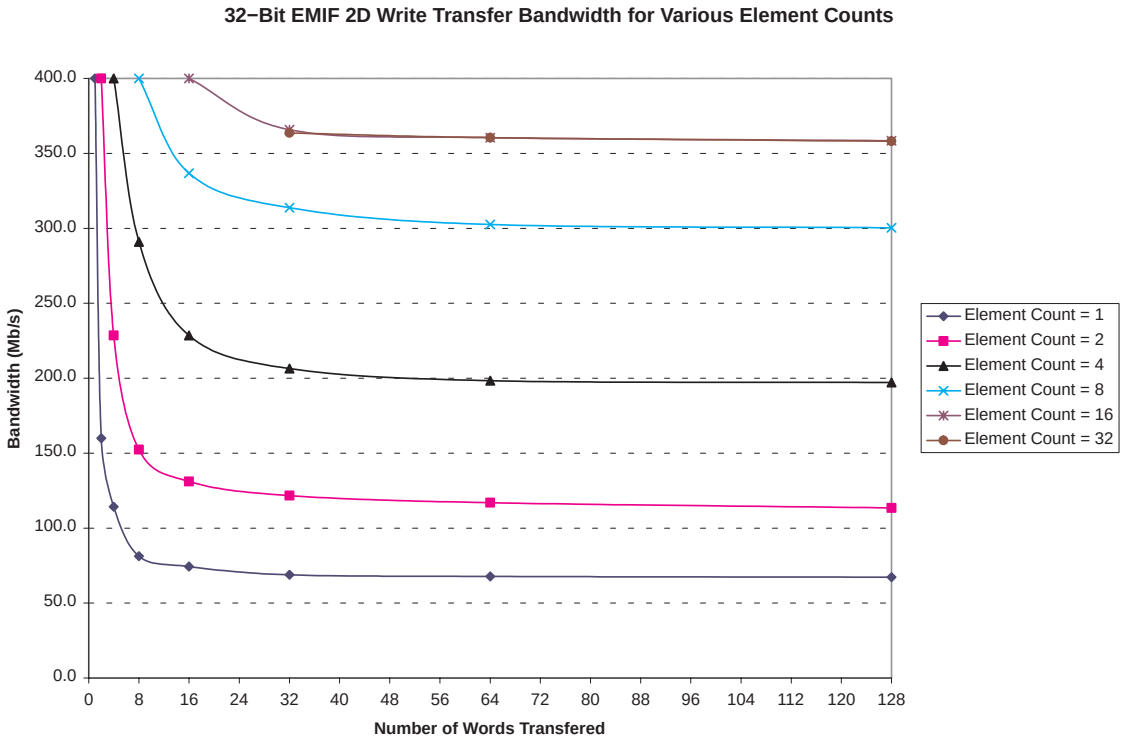

Figure 2. Transfer Bandwidth for 2-D Reads from a 32-bit SDRAM

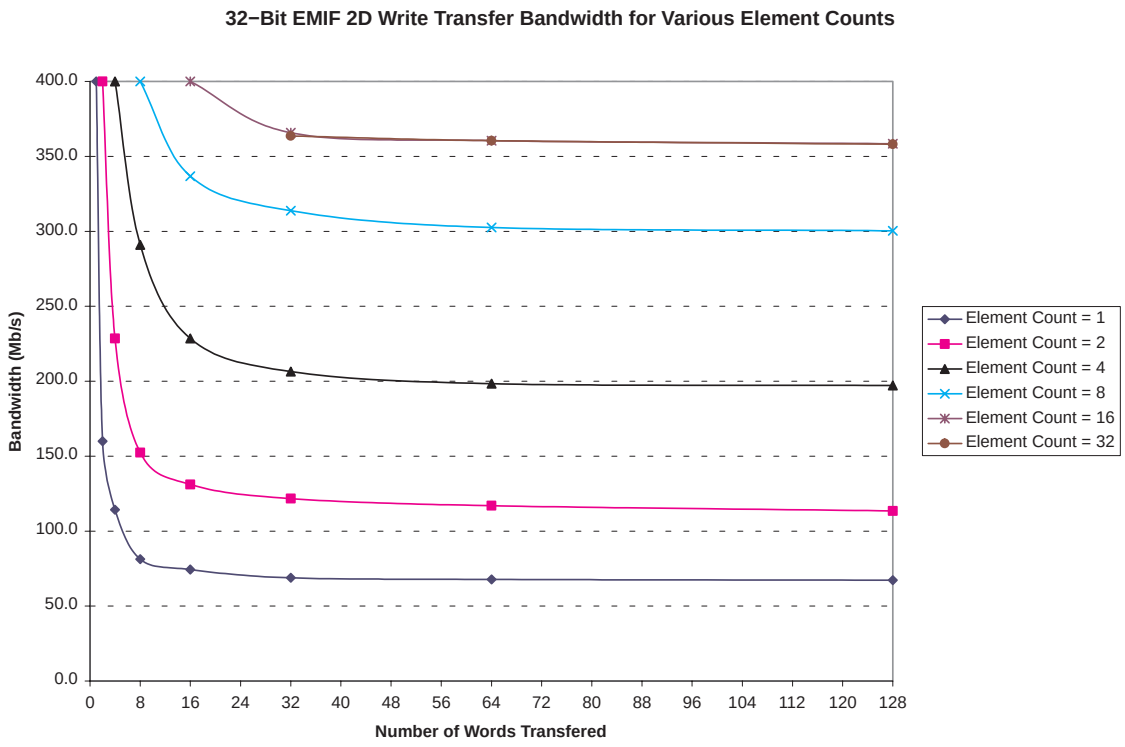

Figure 3. Transfer Bandwidth for 2-D Writes to 32-bit SDRAM

2.2.4 Simultaneous (2-D) EDMA Transfer Requests

Two-dimensional transfers offer a great deal of flexibility, and are generally used in systems with multiple transfers competing for resources. Figure 4 compares total EMIF bandwidth for several 2-D EDMA transfers occurring simultaneously. Various frame and element count combinations were investigated, however, the overall transfer size (element count * frame count) was found to have more bearing than the particular combination settings. Two situations are compared, one where all of the transfers are on the same priority queue, and another where the priority of the transfers is alternated. The results are given below representing the expected range of performance for various element/frame count combinations.

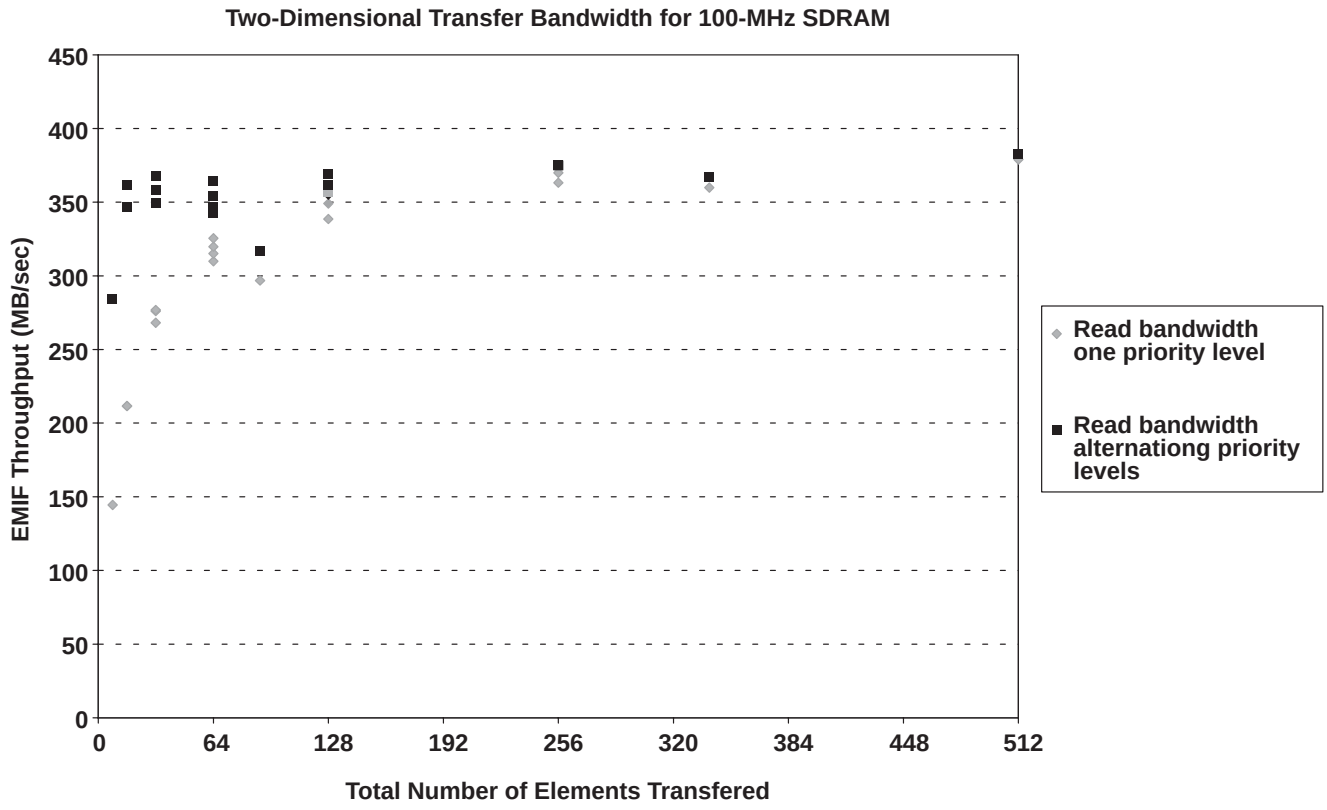

Figure 4. Read Throughput for Multiple Simultaneous 2-D EDMA Transfers

The above data clearly shows the benefit of using more than one priority level for multiple small transfers. For the larger transfers using multiply priority levels is less important. However, keep in mind that transfers on one priority level are processed serially, which may not be desired for large transfers. Simultaneous 2-D write transfers (not presented here) performed closer to ideal transfer rates (400 MB/s), do to write driven processing of the TMS320C621x/TMS320C621x DSP, and the larger write command buffers of the EMIF. See *TMS320C621x/TMS320C621x EDMA Architecture* (SPRA996) for more details.

2.2.5 Simultaneous EDMA Transfer Requests to External Memory

The EDMA is capable of servicing simultaneous transfers on different priorities as long as the transfers are to/from different destination/source ports. The independent source and destination pipelines, however, allow this rule to be broken for the EMIF port. See *TMS320C621x/TMS320C671x EDMA Architecture* (SPRA996) for details. Transfers to external memory on different priority levels and in different directions (read/write), are interleaved by the EMIF and are processed simultaneously along the following division: reads from external memory receive 66% of the bandwidth, and writes receive 33% of the bandwidth, irregardless of priority levels. There is also a switching penalty of 1−4 EMIF clock cycles (every 8 to 16 words) that marginally lowers overall performance as well.

2.3 Element Count

SDRAM has internal paging mechanisms which can cause delays in transfers that tend to increase with element count. The affects these delays have on bandwidth are examined by comparing SBSRAM (no paging) to SDRAM transfers. Table 4 compares 32-bit, 100 MHz SDRAM (ideal bandwidth 400 MB/s) accesses to 32-bit, 100 MHz SBSRAM (ideal bandwidth 400 MB/s) accesses on a 225 MHz TMS320C6713 DSP. Element counts of 4, 64, 256, and 1024 were used. Note that reads are less affected than writes.

Table 4. SBSRAM vs. SDRAM

Access Type	Memory Type	Total Data (bytes)	Source/Destination† Duration(CPU cycles)	Transfer Bandwidth (Mb/sec)
Read	SBSRAM	16	11.3	320.0
Read	SBSRAM	256	146.7	392.6
Read	SBSRAM	1024	577.8	398.8
Read	SBSRAM	4096	2304.9	399.8
Read	SDRAM	16	13.5	266.7
Read	SDRAM	256	148.5	387.9
Read	SDRAM	1024	580.5	396.9
Read	SDRAM	4096	2385.9	386.3
Write	SBSRAM	16	11.3	320.0
Write	SBSRAM	256	153.0	376.5
Write	SBSRAM	1024	612.0	376.5
Write	SBSRAM	4096	2447.6	376.5
Write	SDRAM	16	10.8	333.3
Write	SDRAM	256	159.8	360.6
Write	SDRAM	1024	646.2	356.5
Write	SDRAM	4096	2616.3	352.3

† Note that transfer bandwidth is calculated from this duration and is limited by the slower port (source or destination).

2.4 L2 Bandwidth

The ideal L2 bandwidth of a TMS320C621x/C671x DSP is set by the clock speed of the DSP and the width of the L2 Internal port. To improve internal transfers the L2 is connected to the EDMA by a 64-bit interface. Taking this 64-bit (8 byte) interface and multiplying by a 225 MHz clock speed gives the ideal bandwidth,1800 MB/sec. However, there are many factors that may affect the actual bandwidth attained:

- Ideal bandwidth is determined by CPU frequency − lowering the CPU frequency will lower bandwidth.
- The L2 memory is accessed from the CPU side and from the EDMA side − CPU access to L2 can take L2 bandwidth away from EDMA transfers.
- The L2 memory is cached by the L1 cache − before any data is sent to the EDMA, it must be verified that it is coherent with the L1 cache.

Each of these considerations is examined in the following subsections.

2.4.1 CPU and EMIF Frequency

For some CPU and EMIF frequencies, the ideal L2 bandwidth can become the limiting factor in a transfer. Table 5 explores this. A 32-bit SBSRAM was used to gather data on a TMS320C6713 DSP transferring 1024 bytes to/from external memory.

Table 5. CPU to EMIF Frequency Ratios

CPU Frequency (MHz)	EMIF Frequency (MHz)	Ideal Bandwidth		Actual Transfer Bandwidth (MB/sec)
		EDMA/L2 (MB/sec)	EMIF (MB/sec)	
225	100	1800	400	398.8
166	100	1328	400	398.8
125	100	1000	400	352.1
110	100	880	400	319.2
225	66	1800	266.7	265.7
166	66	1328	266.7	265.8
125	66	1000	266.7	265.7
110	66	880	266.7	265.6

2.4.2 CPU Activity Affecting L2 Bandwidth

Table 6 summarizes CPU effects on EDMA transfers from L2 to EMIF. The CPU executes a typical (incremental) series of load or store instructions to L2 memory attempting to reduce L2 bandwidth from the EDMA transfer. These tests were performed on a TMS320C6713 DSP transferring 4 KB of data to/from a 32-bit SBSRAM.

Table 6. CPU Effects on L2 EDMA Transfers

CPU Frequency	EMIF Frequency	CPU Activity	Ideal Bandwidth (MB/sec)		Actual Transfer Bandwidth (Mb/sec)
			L2	EMIF	
225	100	None	1800	400	400
225	100	LDW	1800	400	400
225	100	STW	1800	400	280
125	100	None	1000	400	330
125	100	LDW	1000	400	320
125	100	STW	1000	400	160

In some cases CPU activity did not reduce L2 bandwidth enough to affect the EDMA transfer.

Some rare types of CPU access to L2 can have negative effects on EDMA transfers to L2. The results shown in following explore the worst-case access pattern. These tests were performed on a TMS320C6713 DSP transferring 4 KB of data to/from a 32-bit SBSRAM.

Table 7. EDMA WEIGHT Effects on EDMA Transfers with Worst-Case CPU Activity (access on every cycle, accesses spaced 8 words apart)

CPU Frequency	EMIF Frequency	CPU Activity	Ideal Bandwidth (MB/sec)		Actual Transfer Bandwidth (Mb/sec)
			L2	EMIF	
225	100	LDW	1800	400	350
225	100	STW	1800	400	Blocked
125	100	LDW	1000	400	200
125	100	STW	1000	400	Blocked

The CPU performs a 8 word spaced store word or load word on every cycle for this worst case test; adding a few delay slots between the accesses makes a substantial increase in performance. This access pattern is rare and generally is not a concern, rather, it is something to look into during debugging. Using the new p-bit in the CCFG of some devices (see device data sheet for details) prevents this EDMA lockout condition. Other solutions include varying the stride pattern to avoid the 8 word stride, adding delay slots, or to write/read to locations allocated in the L1D cache (note that the L1D is a read allocated).

2.4.3 Cache Effects on L2 Bandwidth

L2 data is either invalid (unallocated), clean (allocated and unchanged), or dirty (allocated and changed) in L1 cache. Maintaining the coherency of the data in the L1 cache requires transfers to quickly check the status of the cache prior to actually moving data. This check is very quick if the data that is being moved corresponds to a cache line that is currently invalid. If the cache line is not invalid more time is required to check to see if the cache holds the data that is to be transferred, and if it has been modified. This extra time can lower the transfer bandwidth if the cache is clean or dirty. However, the L2 bandwidth of the TMS320C621x/TMS320C671x DSP is high enough, due to the internal 64-bit bus width, and single cycle transfer rate, that cache status was not observed to have an effect on transfers to external memory − even at low CPU to EMIF clock ratios. On other DSPs the bandwidth difference between L2 and EMIF may not be as large and cache status may have an effect.

2.4.4 L2-to-L2 Transfers and Cache Effects

Utilizing the EDMA (or QDMA) to transfer data from the L2 to the L2 isn't the most efficient way to perform internal data transfers. This is because the data is passed from the L2 to the EDMA, and back to the L2. Since the EDMA is a higher level in the memory hierarchy, the L2 controller must verify the L1 cache coherency before sending data to the EDMA, further reducing performance.

The most efficient way to perform L2-to-L2 transfers is to keep the data in the lower memory levels for the duration of the transfer. This can easily be accomplished with the memcpy() function (located in the rts6200.lib/ rts6700.lib library). With this type of transfer the cache works to increase efficiency, rather then adding overhead as in the EDMA case. There is a trade-off to consider when using the memcpy() function to transfer L2 data, however. The L2 data will be transferred faster, but the CPU is busy transferring data, rather than performing computations.

Table 8 contains data from a 1024 byte, L2-to-L2 transfer on a 225 MHz TMS320C6713 DSP with ideal L2 bandwidth of 1800 MB/sec. However, the memcpy() function only transfers 32-bit elements, halving L2 bandwidth to 900 Mb/s. Additionally, back-to-back L2 SRAM accesses to the same SRAM bank take 2 cycles to complete. See *TMS320C621x/C671x DSP Two-Level Internal Memory Reference Guide* (SPRU609) for details. The cache status column represents the status of the L2 data as cached by L1D.

Table 8. L2-to-L2 Transfers

Transfer Type	Cache Status	Actual Transfer Bandwidth (MB/sec)
Memcpy()	Allocated (clean) in L1D	460
Memcpy()	Allocated (dirty) in L1D	455
Memcpy()	Invalid in L1D	311
EDMA/QDMA	Invalid (unallocated) in L1D	444
EDMA/QDMA	Allocated (clean) in L1D	305
EDMA/QDMA	Allocated (dirty) in L1D	281

2.5 CPU Accesses and Caching

The following sections examine the access time for various CPU accesses to external memory. It should be noted that using the CPU to access data is generally a bad use of resources and should be avoided. Instead the EDMA should be given the task of transferring data so the CPU is free to perform actual computation.

When the CPU accesses external memory spaces a TR may be generated (depending on whether the data is cached). The TR will be for one of the following: a single element – if the memory space is non-cacheable, an L1 cache line – if the memory space is cacheable and the L2 cache is disabled, or an L2 cache line – if the memory space is cacheable and L2 cache is enabled. No TR is generated in the case of an L1 or L2 cache hit.

Incoming and outgoing data to and from the CPU are routed through the L2 controller, and thus utilize the L2 port to submit a TR.

An external memory can be cached by L1 cache, L2 cache, or neither. If the appropriate MAR bit for a memory space is not set, it is not cacheable. If the MAR bit is set and L2 cache size is zero (all L2 is defined as SRAM), the external memory space is cached by L1. If the MAR bit is set and L2 cache size is greater than 0, the external memory space is cached by L2. Remember that L2 SRAM is always cached by L1.

The address increment (or memory stride) controls cache utilization. Contiguous LDW or STW accesses (a stride of one word or 4 bytes) utilize cache memory to the fullest. A memory stride of 32 bytes or more causes every access to miss in the L1 cache because the L1 line size is 32 bytes. A memory stride of 128 bytes causes every access to miss in L2 because the L2 line size is 128 bytes. Access patterns exploring each of these conditions are covered in the next two sections.

2.5.1 CPU Reads (Load Instructions)

For reads, the CPU will stall waiting for the return data. The length of the stall is equal to the sum of the transfer latency, transfer duration, data return time, and some small cache request overhead. For these tests, cache utilization is controlled by the memory access pattern.

Figure 5 and Figure 6 show data collected from a 225 MHz TMS320C6713 reading from 32-bit, 100-MHz external memories. The time required for 128 LDW instructions was measured, and the average time for each instruction is reported.

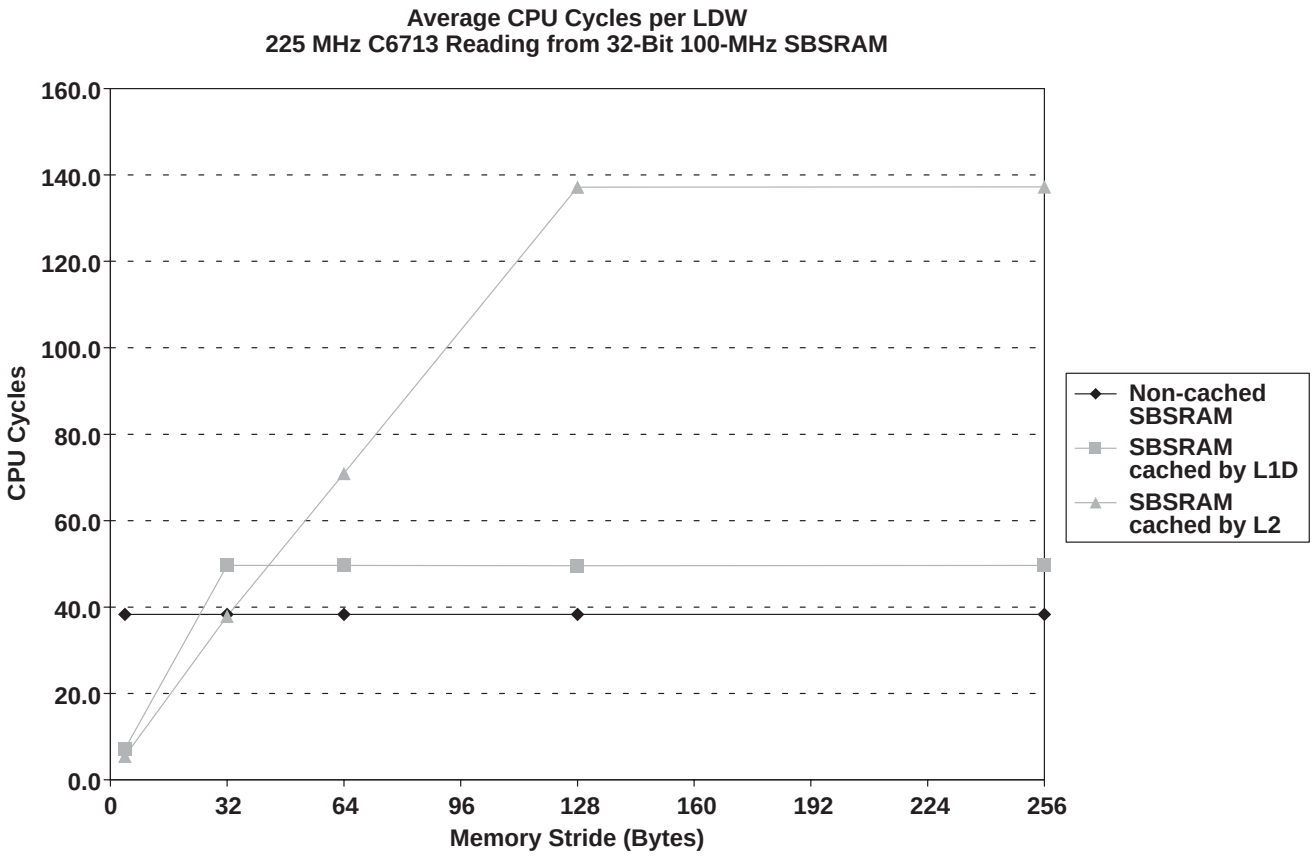

Figure 5. CPU Reads from SBSRAM

Delays caused by SDRAM accesses differ from those caused by SBSRAM accesses because of the additional page misses incurred for each TR (cache miss).

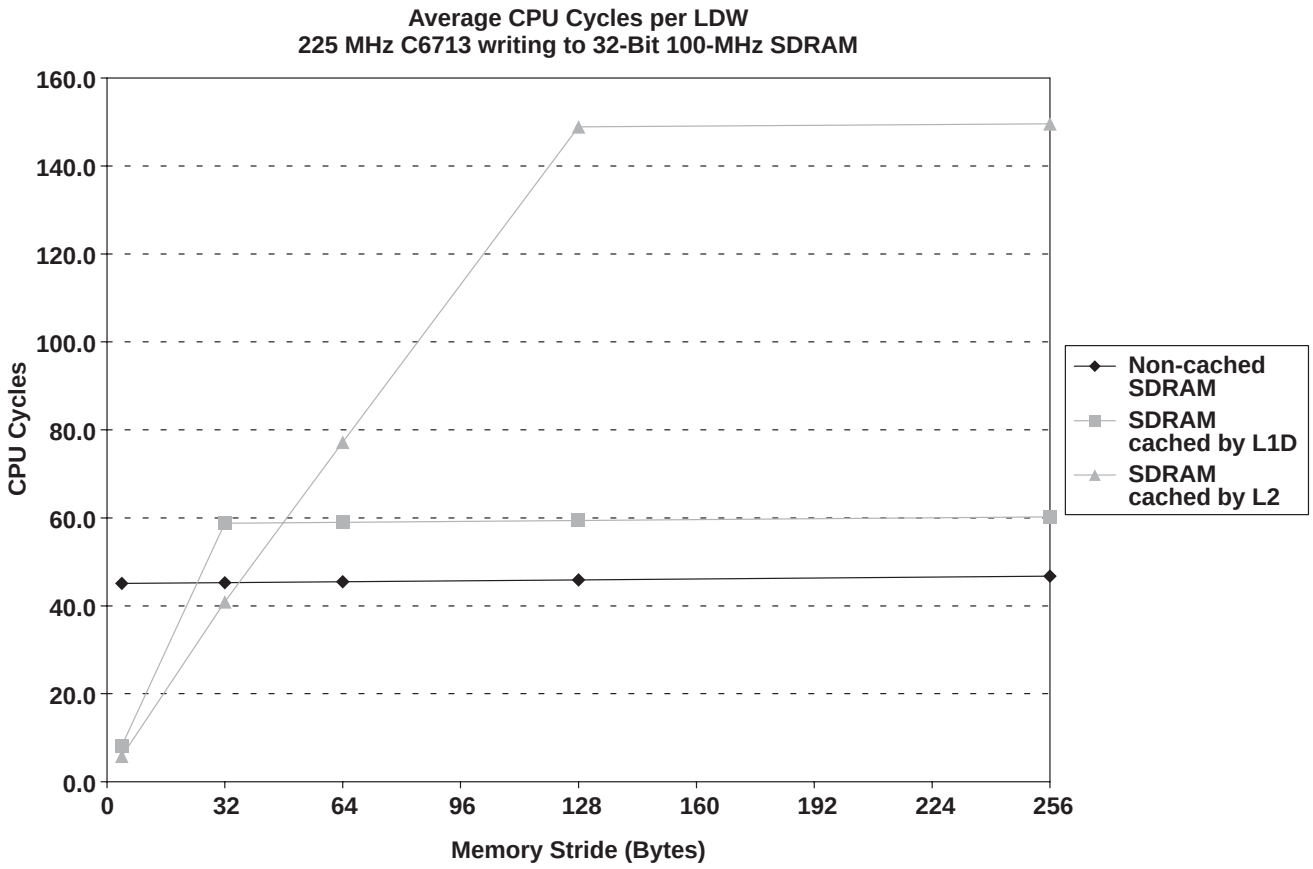

Figure 6. CPU Reads from SDRAM

2.5.2 CPU Writes (Store Instructions)

For writes, a stall may occur if more than four writes are submitted in quick succession. The length of the stall is equal to the sum of the transfer latency, transfer duration, and some small cache overhead. No data return delay is necessary.

The following figure show data collected from a 225 MHz TMS320C6713 writing to 32-bit, 100 MHz external memories. The time required for 128 STW instructions was measured, and the average time for each instruction is reported. For these tests, cache utilization is controlled by the memory access pattern. Since the L1 is a write-through cache, it has no effect on STW instructions. Therefore, only L2 and non-cached values are shown. Also note that cache writebacks are not included in these measurements.

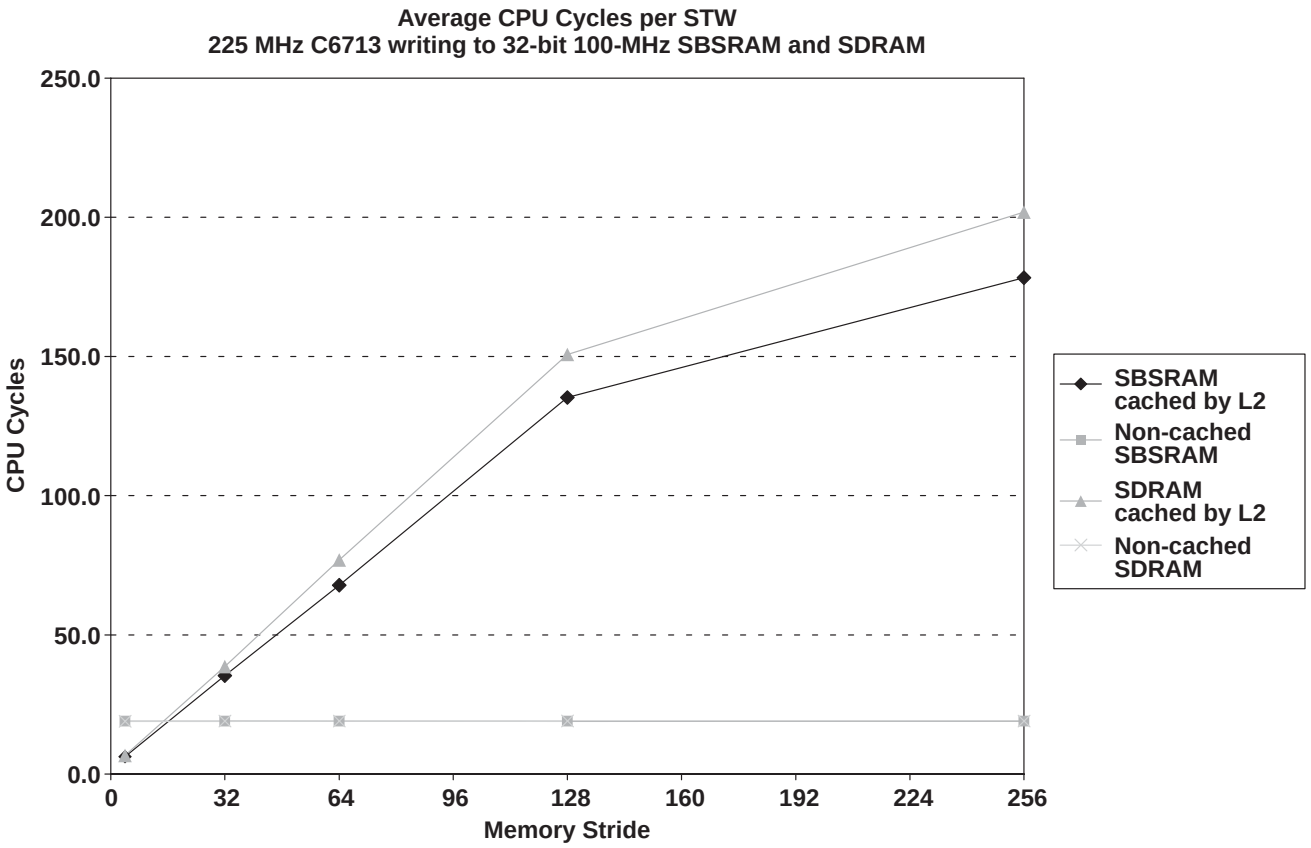

Figure 7. CPU Writes to EMIF

3 Summary

This document presented data from EDMA throughput trials from a variety of tests and provided insight into the cause of certain performance degradation. The numbers presented represent measured delays under the circumstances described. These should give designers a good starting point for estimating system performance. Actual performance is application specific and may vary from these numbers, refer to *TMS320C6000 EDMA IO Scheduling and Performance* (SPRAA00), and *TMS320C621x/TMS320C671x EDMA Architecture* (SPRA996) for more information on creating efficient data transfers in a system.

4 References

1. *TMS320C6000 DSP Peripherals Reference Guide* (SPRU190)
2. *TMS320C6000 EDMA Controller Reference Guide* (SPRU234)
3. *TMS320C621x/TMS320C671x Two-Level Internal Memory Reference Guide* (SPRU609)
4. *TMS320C621x/TMS320C671x EDMA Architecture* (SPRA996)
5. *TMS320C6000 EDMA IO Scheduling and Performance* (SPRAA00)

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265