TMS320C6455

Technical Reference

Literature Number: SPRU965A
May 2005–Revised August 2005

Texas Instruments

Contents

List of Figures

List of Tables

TMS320C6455 Technical Reference

1 Introduction

One of the goals of modern technology is to bring a little piece of home with us wherever we might be. We can call our family from the middle of nowhere via our cell phone or watch our favorite baseball team, even if we are half a world away. We carry our favorite movies and our favorite songs and we expect excellent video and audio quality. High-performance digital signal processors (DSPs) have enabled us to bring our familiar comforts everywhere.

This paper will overview a new DSP core and DSP device, and show how developers of telecommunication, network, and video infrastructure end-equipment and high-end imaging systems will see a system performance gain due to boots in DSP performance and I/O bandwidth. This will allow developers to integrate more high-bandwidth channels, achieve higher image definition, and produce more efficient software easily for faster time-to-market to meet the demands of the modern technology consumers like us.

In 1997, Texas Instruments introduced the TMS320C6000™ platform with the VelociTI™ architecture with the C62x™ and C67x™ cores. The C6000 platform uses an advanced very long instruction word (VLIW) architecture that utilizes multiple execution units executing in parallel. At a 200MHz clock rate and 1600 million instructions per second (MIPS), the original TMS320C6201 device achieved ten times the performance of earlier DSP solutions. From its beginnings, the C6000 architecture set the industry benchmark for MHz and parallelism.

In the year 2000, the next generation in the C6000 platform, the C64x™ core, dramatically extended these performance levels. By adding the VelociTI.2™ extensions, the C64x core doubled the multiply throughput, doubled the register file, and provided 4-10x cycle efficiency in video applications as compared to the original C62x core. In addition, a high-performance 130nm silicon manufacturing process and advanced design techniques tripled the clock rate to 600 MHz. Today, at clock rates of up to 1GHz, the C64x core can process information at a rate of 8000 MIPS or 8 billion instructions per second. These advances in performance were made possible not only by increasing the clock rate, but also by adding packed data processing to the VelociTI architecture.

Today, the new C64x+™ core extends the performance leadership by offering more multiplication bandwidth, higher data bandwidth, and smaller code size than the C64x core. The C64x+ CPU employs eight functional units, two register files and two data paths. Like the earlier C6000 devices, two of these eight functional units are multipliers or .M units. Each C64x+ .M unit doubles the multiply throughput versus the C64x core by performing four 16-bit x 16-bit multiplies every clock cycle. Thus, eight 16-bit x 16-bit multiplies can be executed every cycle on the C64x+ core. At a 1GHz clock rate, this means 8 billion 16-bit multiplies can occur every second. Moreover, each multiplier on the C64x+ core can compute one 32-bit by 32-bit multiply or four 8-bit x 8-bit multiplies every clock cycle. Thirty-two bit multiplies are especially useful in audio applications. Eight-bit data is common in the fields of video and imaging which are other application areas served by the C64x and C64x+ CPUs. On average, the C64x+ core will improve cycle performance by approximately 20% or more as compared to the C64x core.

Clock rate and CPU throughput are only part of the answer. Processing data at these high rates increases the need for I/O bandwidth. The TMS320C6455 DSP, one of the first devices to employ the C64x+ core, has several high bandwidth external interfaces. The C6455 device is also the first broad market DSP to incorporate a serial rapid I/O (sRIO) interface to provide inter-DSP communication. The sRIO interface can operate as a 4x serial bi-directional link at 12.5 Gbits/sec which is fast enough to transmit an uncompressed high-definition 1080p video stream between processors. There are two external memory interfaces; one is a 32-bit double data rate (DDR) interface that provides up to 2GB/sec of memory

bandwidth and the other is a 64-bit interface to both synchronous and asynchronous memories. Additionally, C6455 DSP developers can use a 1Gbit Ethernet interface to network to an IP backplane with eight independent transmit and receive channels. A 66MHz PCI interface is also provided to connect to a PCI host. Data movement in the system is facilitated by an internal enhanced direct memory access (EDMA) engine capable of providing over 5GB/sec with 64 independent channels.

The C6455 DSP integrates a large amount of on-chip memory organized as a two-level memory system. The level one (L1) program and data memories on the C6455 device are 32KB each, double the size of the L1 memories on the C64x devices. Furthermore, this memory can be configured as mapped RAM, cache, or some combination of the two. The level two (L2) memory is 2MB or twice the size of the L2 memory available on the TMS320C6416T DSP.

While there is a large amount of on-chip memory, changes have been made to the C64x+ core to conserve program memory. Traditionally, software written or compiled to take advantage of the parallelism in conventional VLIW architectures often employed techniques that increased code size. One of these optimization techniques, called software pipelining, allows for multiple iterations of a loop to be worked on in parallel. It also introduces some code growth as the loop is prepared at the beginning and winds down at the end. This priming and draining loop code is called prolog and epilog code, respectively. The C64x+ core introduces a software pipeline loop (SPLOOP) buffer that eliminates the need for prolog and epilog code. Another code size reduction feature introduced in the C64x+ core is compact instructions. The native instruction size for the C6000 devices is 32-bits. Many common instructions such as MPY, AND, OR, ADD and SUB can be expressed as 16 bits if the C64x+ compiler can restrict the code to use certain registers in the register file. This compression is performed by the code generation tools. For many DSP applications which consist of both looped code and control (or non-looped) code, the code size reduction can be 20-30% as compared to the C64x core.

The C64x+ core also includes key features that enhance system development such as exception handling, memory protection and privilege. Privilege provides separate operating modes to restrict access to system resources such as memory. These features facilitate the use and/or creation of an operating system. Exceptions aid programmers in isolating defects in their system. Exceptions can be caused by use of illegal opcodes in the software or by hardware events such as a watchdog timer.

In summary, the C6455 DSP utilizing the C64x+ core is object code-compatible with the C64x generation, yet extends the performance leadership by offering greater multiplication bandwidth, higher I/O bandwidth, and larger and more flexible on-chip memory while addressing code size and system issues. These new features will be covered in greater detail in the architectural overview section of this document.

1.1 Applicaton Areas

Manufacturers of telecommunications, network, and video infrastructure end-equipment and high-end imaging systems need flexible, scalable silicon choices and tools to help them keep pace with market trends. Consumers want high quality video, voice, and images. So, end equipment makers need to support streaming video at higher resolutions such as D1 and high definition (HD) resolutions such as 720P, 1080i, or 1080p. The infrastructure providers are also demanding higher channel density in the same physical space. However, since they often cannot increase the board size nor increase the number of boards in the system to accommodate this need, the performance of each chip needs to increase. This added performance will, in effect, supply more channels without increasing the number of chips. To enable product choice and innovation, OEMs need to support an increasing number of standard formats such as MPEG-4, H.264 and WMV9™, in addition to the traditional voice codecs GSM AMR, G.729AB, and G.726. And as standards change, OEMs must still find a way to get to market quickly with new products.

One way we can illustrate the benefits of the C64x+ core architectural enhancements in the applications mentioned above, is to provide benchmark results for our current performance. Table 1 shows the cycle count performance on some key kernels in DSP communications and video/image processing. To illustrate the improvements, Table 1 shows the kernel, number of data inputs, and cycle counts for both the C64x and C64x+ cores. Ratios for cycle count improvement on C64x+ core relative to C64x core are also shown in the table.

The benchmark code can be downloaded from this link:
http://www-s.ti.com/sc/psheets/sprc135/sprc135.zip

Table 1. DSP Kernels/Image Processing Kernel Benchmarks - Cycle Count

Kernel	#Data Inputs	Cycle Count C64x+	Cycle Count C64x	Cycle Count Ratio C64x:C64x+
Real block FIR filter	256 input samples, 32 taps	1046	2080	1.99x
Complex block FIR filter	256 input samples, 32 taps	4112	8222	2.00x
Vector maximum value	256	45	74	1.64x
Complex FFT (16-bit)	1K	3878	6002	1.55x
Complex FFT (32-bit)	1K	7839	11895	1.52x
IDCT 8x8 IEEE 1180-1990 compliant	6 blocks	495	614	1.24x

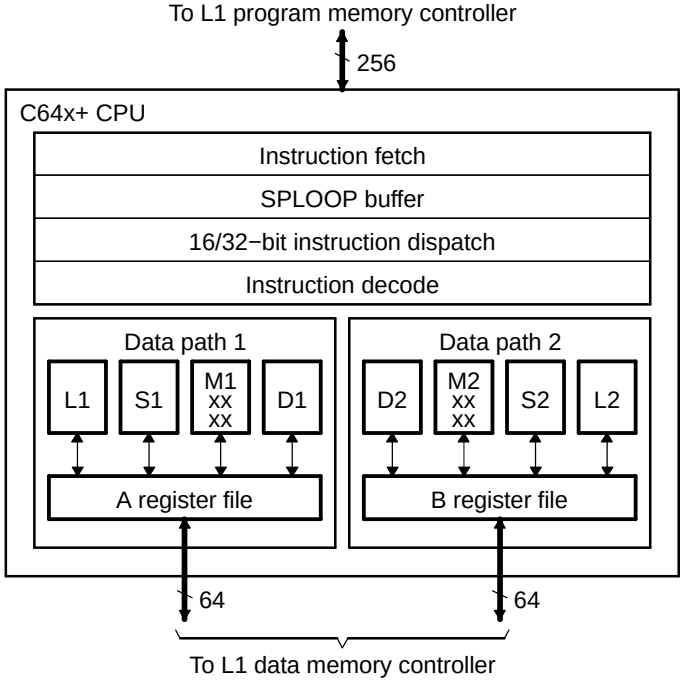
2 Architectural Overview

The C64x+ CPU is a key enabler of high bandwidth processing and is illustrated in a block diagram in Figure 1. The CPU contains:

- Two general-purpose register files (A and B) which each contain 32 32-bit registers for a total of 64 32-bit registers
- Eight functional units (.L1, .S1, .M1, .D1, .D2, .M2, .S2 and .L2)
- Two 64-bit paths to the L1 data memory controller
- One 256-bit path to the L1 program memory controller

Also shown is the instruction fetch and decode blocks highlighting the new SPLOOP buffer and compact instructions. The major changes to the CPU as compared to the C64x core are to the .M1 and .M2 functional units, in addition to the new SPLOOP buffer. Minor changes were made to .L1, .S1, .L2, .S2, and the instruction dispatch unit, as compared to the C64x CPU.

Figure 1. C64x+ CPU Block Diagram



2.1 Functional Units

As just stated, the most significant changes were made to the .M or multiply functional unit. As shown in Table 2, each C64x+ .M unit can perform one of the following at each clock cycle: one 32x32 bit multiply, one 16x32 bit multiply, two 16x16 bit multiplies, two 16x32 bit multiplies, two 16x16 bit multiplies with add/subtract capabilities, four 8x8 bit multiplies, four 8x8 multiplies with add operations, and four 16x16 multiplies with add/subtract capabilities, including a complex multiply. There is also support for Galois field multiplication for 8-bit and 32-bit data. The new operations as compared to the C64x CPU are shown in bold.

Table 2. C64x+ .M Unit Operations

.M unit (.M1, .M2)	16 x 16 multiply operations
	16 x 32 multiply operations
	Quad 8 x 8 multiply operations
	Dual 16 x 16 multiply operations
	Dual 16 x 16 multiply with add/subtract operations
	Quad 8 x 8 multiply with add operations
	Quad 8-bit Galois field multiply
	Quad 16 x 16 multiply with add/subtract operations
	Dual 16 x 32 multiply operations
	32 x 32 multiply operations
	Galois field multiply

The four (or quad) 16x16 bit multiplies with add/subtract capabilities help support fast implementation for many filtering algorithms. For the real block and complex block filters shown in Table 1 the additional multiply capability improves the cycle count performance by 2x as compared to the C64x CPU. Many communications algorithms such as FFTs and modems require complex multiplication. The CMPY instruction takes four 16-bit inputs and produces a 32-bit real and a 32-bit imaginary output. There are also complex multiplies with rounding capability that produces one 32-bit packed output that contain 16-bit real and 16-bit imaginary values. The 32x32 bit multiply instructions provide the extended precision necessary for audio and other high-precision algorithms on a variety of signed and unsigned 32-bit data types.

The .L or arithmetic logic unit, now incorporates the ability to do parallel add/subtract operations on a pair of common inputs. Versions of this instruction exist to work on 32-bit data or work on pairs of 16-bit data performing dual 16-bit add and subtracts in parallel. There are also saturated forms of these instructions. These instructions improve the performance of both fast fourier transforms (FFTs) and discrete cosine transforms (DCTs). In the FFT performance shown in Table 1, the C64x+ CPU performance is 55% higher than that of the C64x CPU for 16-bit data and is 52% higher for 32-bit data. The C64x+ CPU IDCT performance is 24% faster than the C64x CPU for an IEEE 1180-1990 compliant implementation.

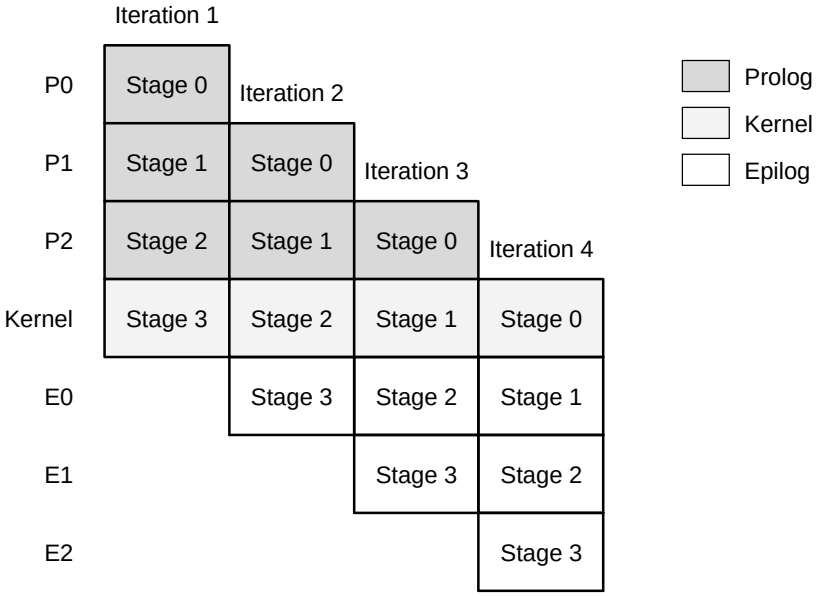
The C64x+ core enhances the .S unit in several ways. In the C64x core, dual 16-bit MIN2 and MAX2 comparisons were only available on the .L units. On the C64x+ core they are also available on the .S unit, which increases the performance of algorithms that do searching and sorting. In the vector maximum search benchmark shown in Table 1, we see the C64x+ core is 64% faster than the C64x core at the same clock rate. Finally, to increase data packing and unpacking throughput, the .S unit allows sustained high performance for the quad 8-bit/16-bit and dual 16-bit instructions. Unpack instructions prepare 8-bit data for parallel 16-bit operations. Pack instructions return parallel results to output precision including saturation support.

2.2 Unique Features

2.2.1 SPLOOP Buffer

As previously mentioned, software pipelining is a technique used to schedule instructions from a loop so that multiple iterations of the loop execute in parallel. Figure 2 illustrates an example of a software pipeline loop. In this figure, a maximum of four iterations can execute at one time. The row labeled Kernel represents the loop kernel. In the loop kernel, all four stages execute in parallel. The area above is the loop prolog and the area below it the loop epilog. The prolog is needed to prime the loop; the first result does not occur until you reach the kernel. The epilog is needed to drain the loop. However, both the prolog and epilog add to code size. The prolog and epilog are merely copies of instructions executed in the loop itself.

Figure 2. Software Pipeline Loop



Another side effect of working on multiple iterations at the same time is there are live variables that need to be preserved if an interrupt occurs. Allowing a software pipeline loop in the C64x or C62x core to be interruptible will potentially affect its performance and increase its code size to preserve the software environment.

The C64x+ CPU has added a software pipeline buffer. The compiler or the programmer creates code for one scheduled loop iteration with some “framing” SPLOOP control instructions. The loop’s instructions and pipeline characteristics build up in an internal buffer. This removes prolog and epilog code. It also reduces program fetches and thus reduces power. Furthermore, loops in the SPLOOP buffer are fully interruptible. No special care is needed to maintain the state.

2.2.2 Compact Instructions

The native instruction size for the C6000 DSP is 32 bits. Instructions are fetched eight instructions at a time, forming a fetch packet of 256 bits. As mentioned previously, up to eight 32-bit instructions can be executed in parallel on the eight functional units per clock cycle. Each 32-bit instruction includes information on which register(s) contain the source operand(s), which register to store the result, what functional unit will perform the operation, the actual opcode and whether the instruction is conditional, and if so, where does it find the condition to execute or not.

Many instructions commonly used in non-looped portions of code can be compressed to 16 bits if the

C64x+ compiler can restrict the code to use a subset of the registers in the register file and make execution of these instructions unconditional. These compact instructions include load instructions (for loading data from memory), store instructions (for loading data to memory), logicals (OR, AND, NEG, XOR), bit manipulation instructions (extract bit fields, set bit fields), shifts, compares (greater than, less than, equal to), arithmetic (ADD and SUB) and single 16x16 bit multiply (MPY) instructions.

Note that compact instructions is not a mode set in the compiler as is done with other processors. Fetch packets can contain both 32-bit and 16-bit instructions. Every fetch packet that contains 16-bit instructions uses a 32-bit header that encodes the layout of the fetch packet and how the 16-bit instructions should be interpreted. The other seven 32-bit values (or words) in the fetch packet can be a mix of 32-bit and 16-bit instructions.

The use of SPLOOP and compact instructions can reduce code size by 20-30% on a wide range of compiled DSP application code including MP3, MPEG4, TCP/IP and GSM AMR when comparing C64x+ to C64x code size using the same compiler revision and options.

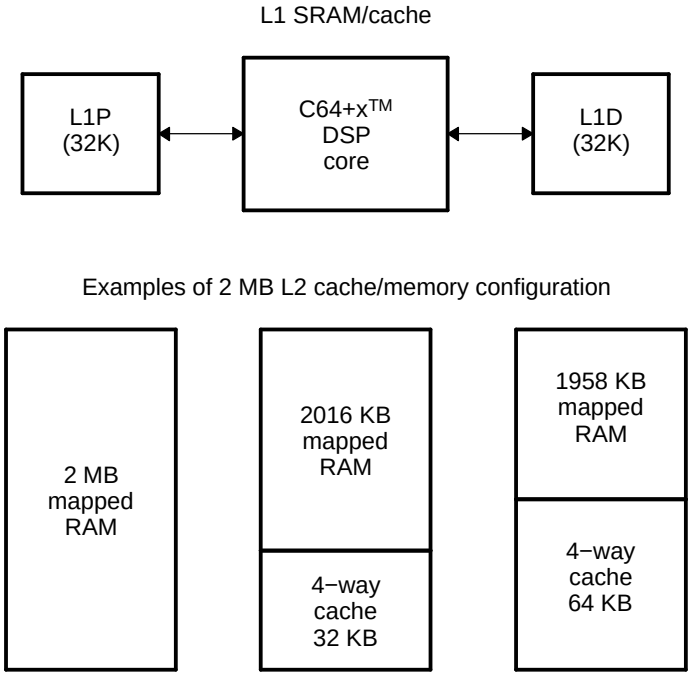
2.3 Chip Level Features

2.3.1 Two-Level Memory Architecture

On initial devices containing the C64x+ core, the CPU interfaces directly to dedicated level-one program (L1P) and data (L1D) memories. These memories operate at the full speed of the CPU. In C64x cores, these memories were limited in size to 16KB and were cache memories only. The C64x+ level one memories can be mapped SRAM, cache or a combination of mapped SRAM and cache. In the C6455 device, the L1P and L1D memories are each 32KB can be configured as 4KB, 8KB, 16KB or 32KB of cache. L1P is a direct mapped cache where as L1D is a two-way set associative cache. The flexibility to configure L1s as RAM allows the user to lock critical code such as interrupt service routines into on-chip memory in the case of L1P and allows critical data sections such as the software stack to be locked on-chip in the case of L1D.

A second level of unified L2 program/data memory provides flexible storage. Figure 3 depicts an example L2 of 2MB, the actual size found in the C6455. One configuration for L2 is to be entirely mapped SRAM. The other configurations have both mapped SRAM and a four-way set associative cache of various sizes (32KB, 64KB, 128KB and 256KB).

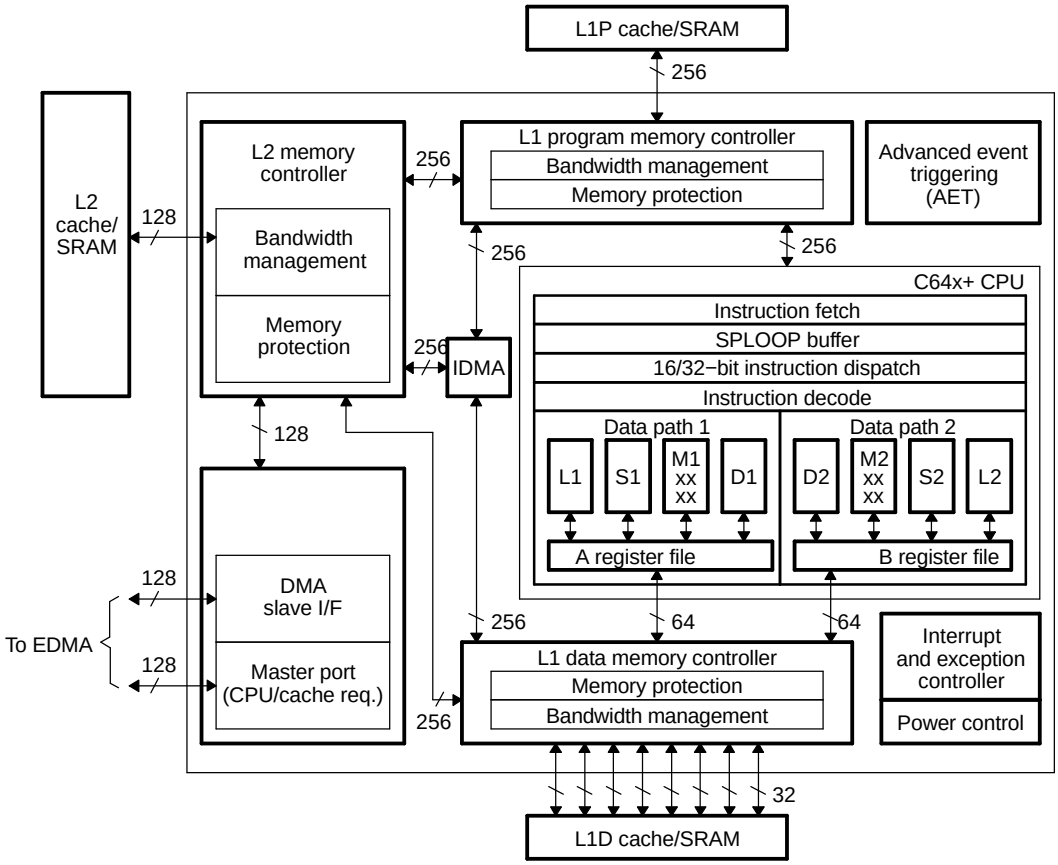
Figure 3. C6455 L1/L2 Memory



2.3.2 IDMA/EDMA

Figure 4 shows the block diagram of the C64x+ megamodule that consists of the CPU, the internal memory blocks and their memory controllers, and the data management engines that handle data movement within the megamodule, including data movement through the peripherals. In the C64x architecture, data movement between L1 and L2 memories was done through a cache controller. The C64x+ architecture has an internal DMA (IDMA) that was created to move data between L1 and L2, as well as perform background peripheral configuration. There are two channels associated with IDMA. Channel 0 configures on-chip peripherals. IDMA channel 1 transfers data between the on-chip memories. It allows movement of data and program sections in the background of CPU operation to set up processing from L1 and to eliminate any cache overhead.

Figure 4. C64x+ Megamodule Block Diagram

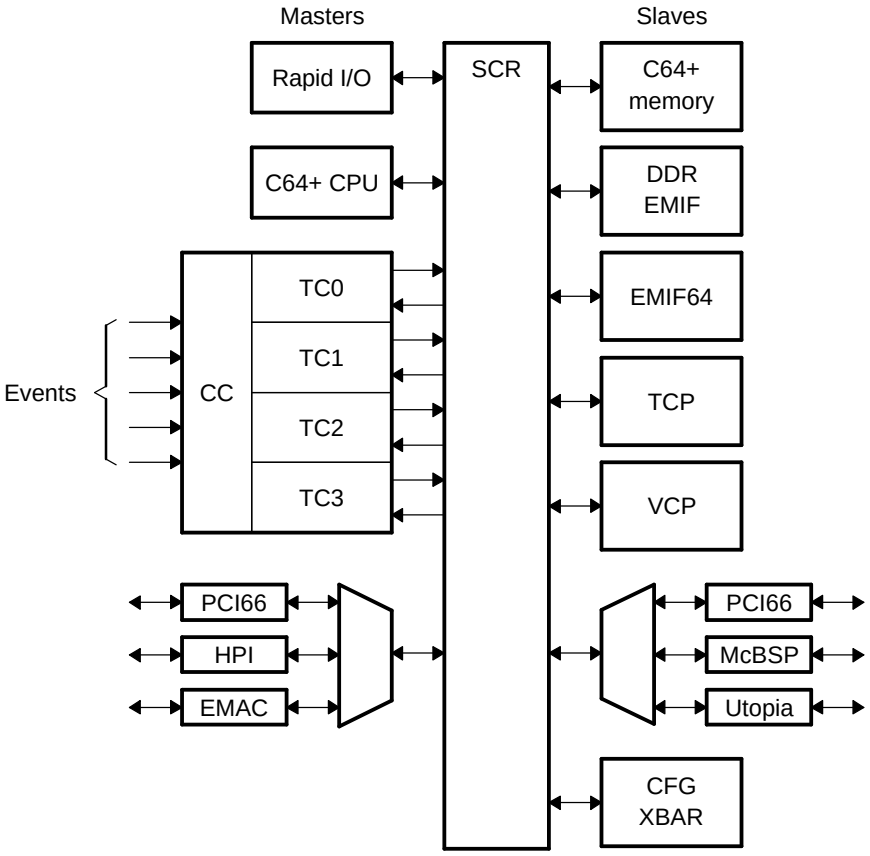


The C6455 EDMA can provide over 5GB/sec of data bandwidth and supports up to 64 DMA channels triggered by independent events (primarily used for hardware events) and four QDMA channels which can be utilized for CPU-submitted algorithmic transfers. A total of 256 parameter sets are available, allowing the user to create ping-pong buffers, circular buffers, and other user-defined data structures at initialization time, with minimal run-time CPU intervention. Linking allows a DMA channel to be reloaded (i.e., linked) with a new parameter set upon completion of the current parameter set. Chaining allows multiple DMA transfers to take place upon receipt of a single trigger event. Linking and chaining allow flexible user-defined data structures and continuous DMA operation without CPU intervention.

The C6455 EDMA as shown in Figure 5 separates the functionality into three distinct components: a switched-central resource (SCR), a channel controller, and a transfer controller. The SCR allows N master peripherals to connect to M slave peripherals. The SCR allows concurrent transfers and minimizes the latency between the master and the slave. The channel controller contains the 64 channels synchronized to the system events noted above. Each channel supports one-, two-, and three-dimensional transfers. The transfer controller (TC) is the part of the EDMA engine that actually performs the reads and writes and buffers the data being transferred. The C6455 device has four transfer controllers allowing four

concurrent data transfers between independent masters and slaves. For example the C64x+ CPU can send data to the Viterbi co-processor (VCP) and at the same time the PCI bus sends data to the DDR interface. In the C64x design there was a single high-speed bus that was time-multiplexed. In this C64x+ chip design, interactions between channels do not affect performance as much as in traditional DMA implementations. This allows data to flow through the system to keep the processing engines fed.

Figure 5. C6455 EDMA



2.3.3 SRIO

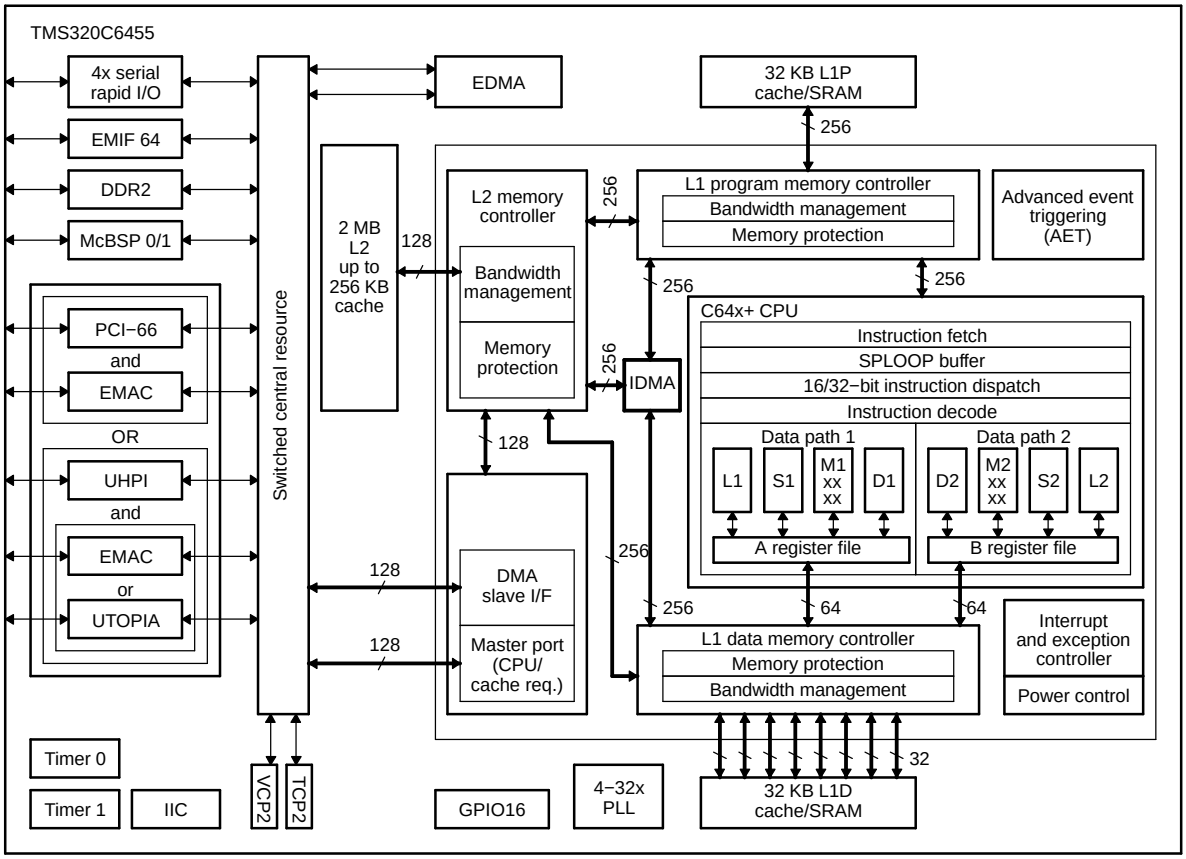
SRIO (IEEE 1149.6) is a worldwide standard with broad adoption suitable for multiple applications needing high-performance and large channel density. It provides flexible / scaleable rates and widths (1x or 4x) with having a low pin count and well defined, scaleable board level interface. For high performance, the sRIO interface provides a high throughput messaging passing capability, achieving nearly 95% of the available data bandwidth of the physical interface. This compares with only 77% for standards such as TCP/IP in the best case. In addition, RapidIO is based on the memory and addressing concepts of processor busses where the data transfer process is controlled by hardware. This enables the interface to offer low-latency and high-scalability system bandwidth as a 1x serial bi-directional link up to 3.125 Gbits/sec, or a 4x serial bi-directional link at 12.5 Gbits/sec. High data rate is achieved through differential source synchronous serializer/deserializer (SERDES) technology. Additionally, sRIO is a low pin count, low power link; one 3.125 Gbits/sec link consumes only 180mW.

2.3.4 External Memory Interfaces

The C6455 device has two memory interfaces. There is a 32-bit DDR2 memory interface providing 2GB/sec throughput at 500MHz based on a 250MHz clock rate. The DDR2 external bus only interfaces to JEDEC DDR2 devices; it does not share the bus with any other types of peripherals. The decoupling of DDR2 memories from other devices both simplifies board design and provides I/O concurrency from a

second external memory interface (EMIF). The 64-bit EMIF supports both asynchronous and synchronous peripherals at a maximum bus rate of 133MHz. The EMIF has four chip-enable spaces that supports synchronous devices like synchronous burst SRAMs and asynchronous devices like SRAMs, FIFOs, and peripheral devices. This EMIF can operate with a dedicated external clock input that decouples the CPU operating frequency from the bus frequency. Figure 6 shows a block diagram of the C6455 device.

Figure 6. Block Diagram of C6455 Device



2.3.5 Ethernet MAC

The Ethernet media access controller (EMAC) provides an efficient interface between the C6455 device and the network. The C6455 EMAC supports 10-baseT, 100-baseTX and 1000-baseTX or 10Mbits/sec, 100Mbits/sec or 1000Mbits/sec. The C6455 EMAC also supports four media-independent interfaces to the physical layer device (PHY): MII, RMII, GMII and RGMII; this includes standard and gigabit functionality in both full and reduced pin counts. The EMAC provides eight independent transmit and eight independent receive channels. Transmit channels can be configured as priority-based or as equal priority (round-robin) queues. Incoming packets can be classified based on up to eight different MAC addresses, and also supports multicast and broadcast modes. The EMAC can directly access both internal and external memory without the need for the CPU to maintain an EDMA channel. The EMAC also has a local 8KB descriptor memory which allows it to send and receive multiple network packets without CPU intervention.

2.4 Ease of Development

The C6000 device remains a very friendly, high-level language compiler target. The CPU architecture and the compiler development continue to be closely coupled. The C64x+ core continues the load/store architecture found in the C6000 platform. By separating arithmetic and memory operations, processor throughput is maximized. The RISC like-instruction set and extensive use of pipelining allow many instructions to be scheduled and executed in parallel.

But the C64x+ core is more than a good C engine; it also includes key features that enhance system

development. Memory protection provides many system benefits and can allow an operating system to create boundaries by defining who/what is or is not authorized to access the memory. The C64x+ core implements this by creating two modes of operation: supervisor mode and user mode. In supervisor mode, the privilege level is highest. The supervisor has access to all resources including memory. The user only gets to access resources for which he has permission. The C6455 internal memory is divided into pages. Each page defines who can access it: supervisor or user, and what level of access is permitted (i.e., read, write, or execute).

If an illegal memory access is attempted, an exception can be generated which notifies the user that something unintended has happened. Besides unauthorized memory access, exceptions can detect and indicate an illegal opcode or indicate that a peripheral experienced a data error. When exceptions occur, the code can be redirected to an error handler to recover from the condition. Memory protection, privilege, and exception handling are all components of a robust operating environment. Many multimedia applications require this level of support and in the absence of a host processor, the C6455 device can handle these system issues on its own.

2.5 Summary

The TMS320C6455 DSP is a programmable platform that addresses the needs of the developers of high-speed communications infrastructure, video infrastructure, and imaging equipment. At clock rates of 1GHz, the C6455 DSP can process information at a rate of 8000 MIPS. The multiplication bandwidth with 16-bit data is doubled as compared to the C6416T DSP. Eight 16-bit x 16-bit multiplies can occur every clock cycle. This capability can be leveraged in telecommunication, network, and video infrastructure applications.

As we have seen, clock rate and CPU throughput are only part of the communications processing solution. Processing data at these high rates increases the need for I/O bandwidth. The C6455 device has several high-speed external interfaces: serial rapid I/O to enhance inter-processor communication, DDR2 to provide 2GB/sec of memory bandwidth, and 1Gbit EMAC interface to network to an IP backplane with eight independent transmit and receive channels. In addition to having fast external buses, the internal enhanced DMA engine capable of providing over 5GB/sec with up to four concurrent transfers per clock cycle.

The C6455 device has doubled the amount of on-chip memory as compared to the C6416T. The L1 program and data memories are 32KB each and the L2 memory is 2MB. Furthermore, the memory system has the added flexibility of having the level one memories configured as RAM, cache, or some combination of the two.

While there is a large amount of on-chip memory, changes have been made to the C64x+ core to conserve program memory. With the introduction of the SPLOOP buffer and compact instructions which are automatically utilized by the code generation tools, the code size for many DSP applications can be 20-30% smaller as compared to the C64x core.

In summary, the C6455 device utilizing the C64x+ core is object-code compatible with the C64x generation, yet extends the performance leadership by offering greater multiplication bandwidth, higher I/O bandwidth, larger and more flexible on-chip memory while addressing code size, and system issues such as memory protection and privilege. The C6455 DSP enables developers of telecommunications, network and video infrastructure end-equipment, and high-end imaging systems to see a system performance gain. This gain will allow them to integrate more high-bandwidth channels, achieve higher image definition, and produce more efficient software easily for faster time-to-market. Thus, end-equipment manufacturers can keep pace with the never-ending and ever-increasing demand for high quality video, voice, and data by consumers.

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