TEXAS INSTRUMENTS

TMS320C62xx

*Technical
Brief*

TEXAS
INSTRUMENTS

TMS320C62xx
Technical Brief

Preliminary

PRINTED WITH
SOY INK™

TEXAS INSTRUMENTS

Printed on Recycled Paper

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Read This First

About This Manual

This manual introduces the TMS320C62xx devices. The TMS320C6201 device is the most powerful general-purpose programmable digital signal processor (DSP) available. The information in this manual describes the devices and provides a basic overview of how to use them. For more detailed information, see the related documentation.

How to Use This Manual

This document contains the following chapters:

Chapter 1, Introduction, describes the main features of the TMS320C62xx devices, the history of TI DSPs, and typical applications.

Chapter 2, CPU Architecture, describes the architecture of the TMS320C62xx devices, with a block diagram and brief introduction to the parts of the device.

Chapter 3, Memory, describes the on-chip memory and the external memory interface.

Chapter 4 Peripherals, describes the peripherals available for the 'C62xx devices, such as ports, timers, direct-memory access, and power-down logic.

Chapter 5, Development Support, describes the tools, documentation, Web site, and third-party support for the TMS320C6x.

Related Documentation From Texas Instruments

The following books describe the TMS320C62xx devices and related support tools. To obtain a copy of any of these TI documents, call the Texas Instruments Literature Response Center at (800) 477–8924. When ordering, please identify the book by its title and literature number.

TMS320C6x Assembly Language Tools User's Guide (literature number SPRU186) describes the assembly language tools (assembler, linker, and other tools used to develop assembly language code), assembler directives, macros, common object file format, and symbolic debugging directives for the 'C6x generation of devices.

TMS320C62xx CPU and Instruction Set Reference Guide (literature number SPRU189) describes the 'C62xx CPU architecture, instruction set, pipeline, and interrupts for the TMS320C62xx digital signal processors.

TMS320C6x C Source Debugger User's Guide (literature number SPRU188) tells you how to invoke the 'C6x simulator versions of the C source debugger interface. This book discusses various aspects of the debugger interface, including window management, command entry, code execution, data management, and breakpoints.

TMS320 DSP Designer's Notebook: Volume 1 (literature number SPRT125) presents solutions to common design problems using 'C2x, 'C3x, 'C4x, 'C5x, and other TI DSPs.

TMS320C6x Optimizing C Compiler User's Guide (literature number SPRU187) describes the 'C6x C compiler. This C compiler accepts ANSI standard C source code and produces assembly language source code for the 'C6x generation of devices. This book also describes the assembly optimizer, which helps you optimize your assembly code.

TMS320C62xx Peripherals Reference Guide (literature number SPRU190) describes common peripherals available on the TMS320C62xx digital signal processors. This book includes information on the internal data and program memories, the external memory interface (EMIF), the host port, serial ports, direct memory access (DMA), clocking and phase-locked loop (PLL), and the power-down modes.

TMS320C62xx Programmer's Guide (literature number SPRU198) describes ways to optimize C and assembly code and includes application program examples.

TMS320C6x Software Tools Getting Started Guide (literature number SPRU185) describes how to install the TMS320C6x assembly language tools, the C compiler, the simulator, and the C source debugger. Installation instructions for SunOS™, Solaris™, Windows™ 95, and Windows NT™ systems are given.

TMS320C6201 Digital Signal Processor Data Sheet (literature number SPRS051) describes the features of the TMS320C6xx and provides pinouts, electrical specifications, and timings for the device.

If You Need Assistance . . .

❏ **World-Wide Web Sites**	
TI Online	http://www.ti.com
Semiconductor Product Information Center (PIC)	http://www.ti.com/sc/docs/pic/home.htm
DSP Solutions	http://www.ti.com/dsps
320 Hotline On-line ™	http://www.ti.com/sc/docs/dsps/support.htm

❏ **North America, South America, Central America**		
Product Information Center (PIC)	(972) 644-5580	
TI Literature Response Center U.S.A.	(800) 477-8924	
Software Registration/Upgrades	(214) 638-0333	Fax: (214) 638-7742
U.S.A. Factory Repair/Hardware Upgrades	(281) 274-2285	
U.S. Technical Training Organization	(972) 644-5580	
DSP Hotline	(281) 274-2320	Fax: (281) 274-2324 Email: dsph@ti.com
DSP Modem BBS	(281) 274-2323	
DSP Internet BBS via anonymous ftp to ftp://ftp.ti.com/mirrors/tms320bbs		

❏ **Europe, Middle East, Africa**		
European Product Information Center (EPIC) Hotlines:		
Multi-Language Support	+33 1 30 70 11 69	Fax: +33 1 30 70 10 32 Email: epic@ti.com
Deutsch	+49 8161 80 33 11 or +33 1 30 70 11 68	
English	+33 1 30 70 11 65	
Francais	+33 1 30 70 11 64	
Italiano	+33 1 30 70 11 67	
EPIC Modem BBS	+33 1 30 70 11 99	
European Factory Repair	+33 4 93 22 25 40	
Europe Customer Training Helpline		Fax: +49 81 61 80 40 10

❏ **Asia-Pacific**		
Literature Response Center	+852 2 956 7288	Fax: +852 2 956 2200
Hong Kong DSP Hotline	+852 2 956 7268	Fax: +852 2 956 1002
Korea DSP Hotline	+82 2 551 2804	Fax: +82 2 551 2828
Korea DSP Modem BBS	+82 2 551 2914	
Singapore DSP Hotline		Fax: +65 390 7179
Taiwan DSP Hotline	+886 2 377 1450	Fax: +886 2 377 2718
Taiwan DSP Modem BBS	+886 2 376 2592	
Taiwan DSP Internet BBS via anonymous ftp to ftp://dsp.ee.tit.edu.tw/pub/TI/		

❏ **Japan**		
Product Information Center	+0120-81-0026 (in Japan)	Fax: +0120-81-0036 (in Japan)
	+03-3457-0972 or (INTL) 813-3457-0972	Fax: +03-3457-1259 or (INTL) 813-3457-1259
DSP Hotline	+03-3769-8735 or (INTL) 813-3769-8735	Fax: +03-3457-7071 or (INTL) 813-3457-7071
DSP BBS via Nifty-Serve	Type "Go TIASP"	

❏ **Documentation**
When making suggestions or reporting errors in documentation, please include the following information that is on the title page: the full title of the book, the publication date, and the literature number.
Mail: Texas Instruments Incorporated   Email: comments@books.sc.ti.com
Technical Documentation Services, MS 702
P.O. Box 1443
Houston, Texas   77251-1443

Note: When calling a Literature Response Center to order documentation, please specify the literature number of the book.

Trademarks

Classico, MicroLite, and Virtuoso Nano are trademarks of Eonic Systems, Inc.

EVP is a trademark of D2 Technologies.

InvisiLink is a trademark of ViaDSP, Inc.

PC is a trademark of International Business Machines Corporation.

Solaris, SunOS, and Sun-3 are trademarks of Sun Microsystems, Inc.

TI, cDSP, VelociTI, and XDS510 are trademarks of Texas Instruments Incorporated.

Windows, Windows 95, and Windows NT are registered trademarks of Microsoft Corporation. (Windows™, Windows™ 95, Windows NT™).

Contents

Figures

Tables

Introduction

The TMS320C62xx devices feature VelociTI™, an advanced very long instruction word (VLIW) architecture developed by Texas Instruments. VelociTI, together with the development tool set and evaluation tools, provides faster development time and higher performance with increased instruction-level parallelism.

1.1 Introduction to the TMS320C6x Generation

With performance of up to 1600 million instructions per second (MIPS) and a complete set of development tools, the 'C62xx devices offer cost-effective solutions to high-performance DSP programming challenges. The 'C6x development tools include a new C compiler, an Assembly optimizer that simplifies programming and scheduling, and a Windows-based debugger interface. VelociTI combines advanced VLIW architecture with a high degree of parallelism to produce a device that enables applications such as:

❑ Unlimited Internet bandwidth
❑ Universal wireless communications
❑ Radical new telephony features
❑ Remote medical diagnostics
❑ Ultimate automated cruise control
❑ Personal home base station
❑ Personalized home security

The 'C62xx devices also can be used for improved performance on existing applications, such as:

❑ Wireless base stations
❑ Pooled modems and remote access servers
❑ Next-generation xDSL modems and cable modems
❑ Multichannel telephony platforms including central office switches, PBXs, and voice-messaging systems
❑ Multimedia systems

1.2 The TMS320 Family of DSPs

The TMS320 family consists of both 16-bit fixed-point and 32-bit floating-point devices. These DSPs possess the operational flexibility of high-speed controllers and the numerical capability of array processors. The following characteristics make this family the ideal choice for a wide range of processing applications:

❑ Very flexible instruction set
❑ Inherent operational flexibility
❑ High-speed performance
❑ Innovative, parallel architectural design
❑ Cost-effectiveness

1.2.1 History, Development, and Advantages of TMS320 DSPs

In 1982, Texas Instruments introduced the TMS32010 — the first fixed-point DSP in the TMS320 family. Before the end of the year, the *Electronic Products* magazine awarded the TMS32010 the title "Product of the Year". The TMS32010 became the model for future TMS320 generations.

Today, the TMS320 family consists of nine generations: the 'C1x, 'C2x, 'C2xx, 'C5x, and 'C54x are fixed-point, the 'C3x and 'C4x are floating-point, the 'C8x is a multiprocessor, and the 'C6x will offer both fixed-point and floating-point devices. The first device in the 'C6x generation is the TMS320C6201, which is a fixed-point DSP.

Each generation of TMS320 devices has a central processing unit (CPU) and a variety of on-chip memory and peripheral configurations. These spin-off devices satisfy a wide range of needs in the worldwide electronics market. When memory and peripherals are integrated into one processor, the overall system cost is greatly reduced, and circuit board space is saved. Figure 1–1 shows the progress of the TMS320 family of devices.

Figure 1–1. The TMS320 Family of DSPs
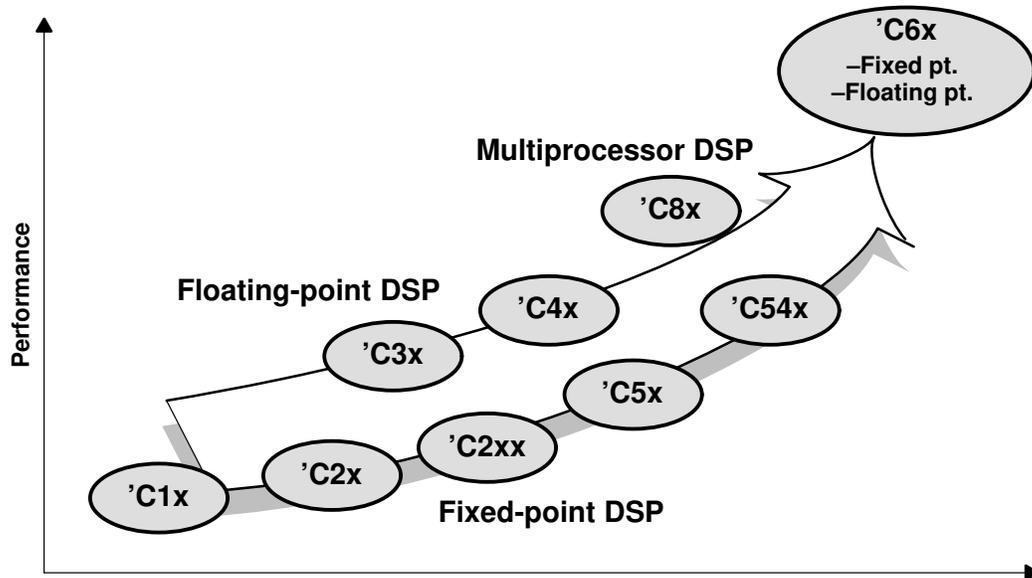
1.2.2 Typical Applications

The TMS320 family of DSPs offers better, more adaptable approaches to traditional signal-processing problems, such as vocoding, filtering, and error coding. Furthermore, the TMS320 family supports complex applications that often require multiple operations to be performed simultaneously. Table 1–1 shows many of the typical applications of the TMS320 family.

Table 1–1. Typical Applications for the TMS320 Family

Automotive	Consumer	Control
Adaptive ride control	Digital radios/TVs	Disk drive control
Antiskid brakes	Educational toys	Engine control
Cellular telephones	Music synthesizers	Laser printer control
Digital radios	Pagers	Motor control
Engine control	Power tools	Robotics control
Global positioning	Radar detectors	Servo control
Navigation	Solid-state answering machines	
Vibration analysis		
Voice commands		

General-Purpose	Graphics/Imaging	Industrial
Adaptive filtering	3-D rotation	Numeric control
Convolution	Animation/digital maps	Power-line monitoring
Correlation	Homomorphic processing	Robotics
Digital filtering	Image compression/transmission	Security access
Fast Fourier transforms	Image enhancement	
Hilbert transforms	Pattern recognition	
Waveform generation	Robot vision	
Windowing	Workstations	

Instrumentation	Medical	Military
Digital filtering	Diagnostic equipment	Image processing
Function generation	Fetal monitoring	Missile guidance
Pattern matching	Hearing aids	Navigation
Phase-locked loops	Patient monitoring	Radar processing
Seismic processing	Prosthetics	Radio frequency modems
Spectrum analysis	Ultrasound equipment	Secure communications
Transient analysis		Sonar processing

Telecommunications		Voice/Speech
1200- to 56 600-bps modems	Faxing	Speaker verification
Adaptive equalizers	Future Terminals	Speech enhancement
ADPCM transcoders	Line repeaters	Speech recognition
Base Stations	Personal communications	Speech synthesis
Cellular telephones	systems (PCS)	Speech vocoding
Channel multiplexing	Personal digital assistants (PDA)	Text-to-speech
Data encryption	Speaker phones	Voice mail
Digital PBXs	Spread spectrum communications	
Digital speech interpolation (DSI)	xDSL	
DTMF encoding/decoding	Video conferencing	
Echo cancellation	X.25 packet switching	

1.3 Key Features of the TMS320C62xx Devices

The TMS320C62xx devices are fixed-point processors based on the advanced VLIW CPU with eight functional units, including two multipliers and six arithmetic logic units. The CPU can execute up to eight instructions per cycle for up to ten times the performance of typical DSPs. The advanced VLIW architecture allows designers to develop highly effective reduced instruction-set computer (RISC)-like code for fast development time. Features common to all the devices in the 'C62xx series are listed in Table 1–2.

Table 1–2. Key Features of the TMS320C62xx Devices

Feature	Benefit
Advanced VLIW CPU with eight functional units including two multipliers and six arithmetic logic units	Executes up to eight instructions per cycle for up to ten times the performance of typical DSPs. Allows designers to develop highly effective reduced instruction-set computer (RISC)-like code for fast development time
Instruction packing	Code size equivalence for eight instructions executed serially or in parallel. Reduces code size, program fetches, and power consumption
100% conditional instructions	Reduces costly branching Increases parallelism for higher sustained performance.
Code executes as programmed on highly independent functional units	The most efficient C compiler in the industry on DSP benchmark suite and industry's first assembly optimizer for fast development time
8-/16-/32-bit data support	Efficient memory support for a variety of applications
40-bit arithmetic options	Extra precision for vocoders and other computationally intensive applications
Saturation and normalization	Support for key arithmetic operations
Bit-field manipulation and instruction: extract, set, clear, bit counting	Supports common operations found in control and data manipulation applications

The first device in the family is the TMS320C6201. The early release of this device includes memory, the external memory interface (EMIF), direct memory access (DMA) with two channels, the host-port interface (HPI), and a flexible phase-locked loop (PLL) clock generator. The production version of this device also will have two enhanced-buffered serial ports and two 32-bit timers.

Table 1–3 summarizes the key features of the TMS320C6201 device.

Table 1–3. Features of the TMS320C6201

Feature	Benefit
Bit-field manipulation and instruction: extract, set, clear, bit counting	Supports common operation found in control and data manipulation applications
1M-bit on-chip memory (512K-bit program, 512K-bit data)	Fast algorithm execution with fewer components per system
32-bit external memory interface supports synchronous dynamic random access memory (SDRAM), synchronous burst static RAM (SBSRAM), and static RAM (SRAM)	High speed connections to external memory for maximum sustained performance
Two enhanced-buffered serial ports (EBSPs)	Glueless interface to high-bandwidth telecommunications trunks Provides high-speed interprocessor communication
16-bit host access port	Host processor access to on-chip data memory
Two data memory access channels with boot loading capability	Efficient access to external memory/peripherals while minimizing CPU interrupts
Flexible PLL clock generator	Multiplies external clock rate for two or four for maximum CPU performance
352-lead ball grid array package	Ultra-thin package minimizes board space

CPU
Architecture

The VelociTI architecture makes the 'C62xx the first off-the-shelf DSP to use advanced VLIW to achieve high performance through increased instruction-level parallelism. A traditional VLIW architecture consists of multiple execution units running in parallel that perform multiple instructions during a single clock cycle. Parallelism is the key to extremely high performance, taking these next-generation DSPs well beyond the performance capabilities of traditional superscalar designs. VelociTI is a highly deterministic architecture, with few restrictions on how or when instructions are fetched, executed, or stored. This architectural flexibility is key to the break-through efficiency levels of the 'C6x compiler.

2.1 TMS320C62xx Block Diagram

The C62xx processor consists of three main parts - CPU (or the "core"), peripherals, and memory. The first device in the series, the TMS320C6201, is a fixed-point DSP using the VelociTI VLIW architecture. Eight functional units operate in parallel, with two identical sets of the basic four functional units. The units communicate through two register files, which each contain 16 32-bit registers. Program parallelism is defined at compile time since there is no data dependency checking done in hardware during run time. The 256-bit-wide program memory fetches eight 32-bit instructions every single cycle.

Figure 2–1 shows the block diagram for the TMS320C6201 digital signal processor (DSP). 'C62xx DSPs are based on the 'C62xx CPU. 'C62xx devices come with program memory which on some devices can be used as a program cache. The devices also have varying sizes of data memory. Peripherals such as a DMA controller, power-down logic, and EMIF usually come with the CPU, and peripherals such as serial ports and timers are available on certain devices. Check the data sheet for your device to determine the specific peripheral configurations you have.

Figure 2–1. TMS320C6201 CPU Core With Peripherals
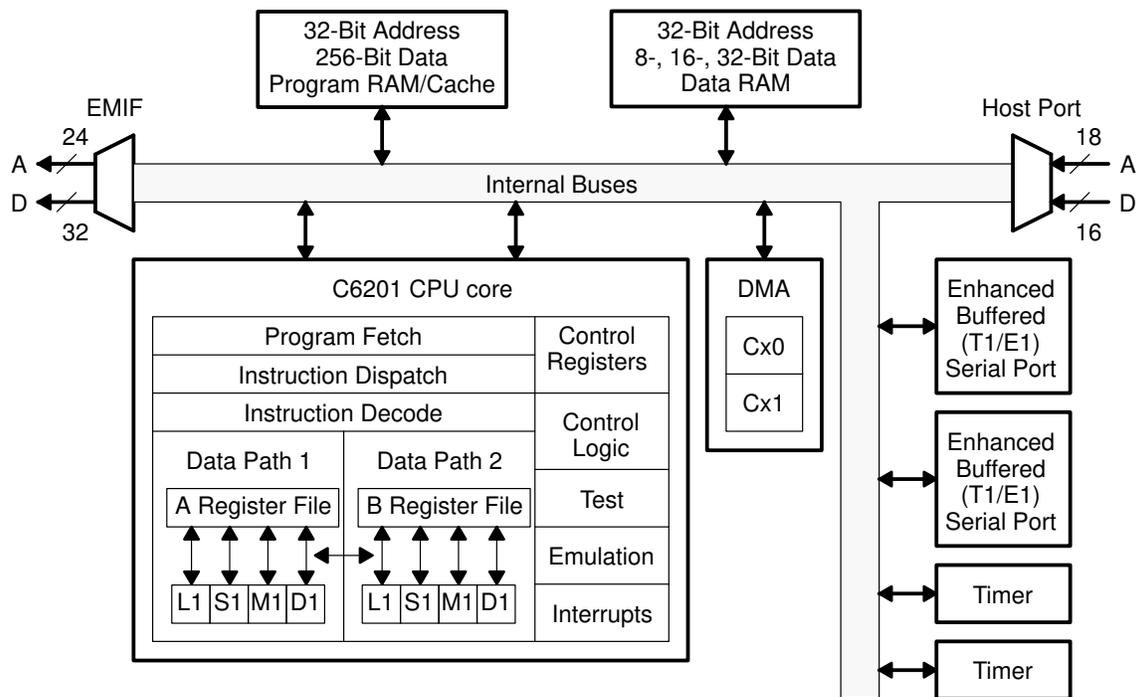
2.2 Central Processing Unit (CPU)

The 'C62xx central processing unit (CPU) is the central building block of all the TMS320C62xx devices. The CPU contains:

❑ Program fetch unit
❑ Instruction dispatch unit
❑ Instruction decode unit
❑ 32 registers
❑ Two data paths, each with four functional units
❑ Control registers
❑ Control logic
❑ Test, emulation, and interrupt logic

The CPU has two data paths where processing occurs. Each data path has four functional units (.L, .S, .M, .D) and a register file containing 16 32-bit registers. The functional units execute logic, shifting, multiply, and data address operations. All instructions operate on the registers. The two sets of data-addressing units (.D1 and .D2) are exclusively responsible for all data transfers between the register files and the memory.

The four functional units on each side of the CPU share the control register files. Each side also has a single data bus connected to registers on the other side of the CPU so that the units can cross–exchange data from register files on opposite sides. Register access across the CPU supports only one read and write operation per cycle.

 The two sets of functional units include the following:

❑ Two multipliers
❑ Six arithmetic logic units (ALUs)
❑ 32 registers with 32-bit word length each

Each functional unit is controlled by a 32-bit instruction. The instruction fetch, instruction dispatch, and instruction decode blocks can deliver up to eight 32-bit instructions from the program memory to the functional units every cycle. The control register file provides methods to configure and control various aspects of processor operation.

The VLIW processing flow begins when a 256-bit-wide instruction fetch packet is fetched from the internal program memory. The instructions are linked together by the least significant bit (LSB) positions of the instruction. The instructions linked together for simultaneous execution (up to eight in total) comprise an execute packet. For more details on the processing, see the *TMS320C6201 Digital Signal Processor* data sheet (literature number SPRS051).

The program fetch, instruction dispatch, and instruction decode units can deliver up to eight 32-bit instructions from the program memory to the functional units every cycle. Processing occurs in each of the two data paths (A and B). Each data path has four functional units (.L, .S, .M, and .D) and a register file containing 16 32-bit registers. Each functional unit is controlled by a 32-bit instruction. To understand how instructions are fetched, dispatched, decoded, and executed in the data path, refer to the chapter on pipeline operation in the *TMS320C62xx CPU and Instruction Set Reference Guide* (literature number SPRU189).

2.3   CPU Data Paths

Figure 2–2 shows the 'C62xx CPU data paths, which consists of:

❏   Two general purpose register files (A and B)
❏   Eight functional units (.L1, .L2, .S1, .S2, .M1, .M2, .D1, and .D2,)
❏   Two load-from-memory paths (LD1 and LD2)
❏   Two store-to-memory paths (ST1 and ST2)
❏   Two register file cross paths (1X and 2X)

2.3.1   General-Purpose Register Files

There are two general-purpose register files (A and B) in the 'C62xx data paths. Each of these files contains 16 32-bit registers (labeled A0–A15 for file A and B0–B15 for file B). The general purpose registers can be used for data or data-address pointers. Registers A1, A2, B0, B1, and B2 can be used for condition registers.

2.3.2   Functional Units

The eight functional units in the 'C62xx data paths can be divided into two groups of four, each of which is virtually identical for each register file. The functional units are described in Table 2–1.

Most data lines in the CPU support 32-bit operands, and some support long (40-bit) operands. Each functional unit has its own 32-bit write port into a general-purpose register file. All units ending in 1 (for example, .L1) write to register file A and all units ending in 2 write to register file B. Each functional unit has two 32-bit read ports for source operands *src*1 and *src*2. Four units (.L1, .L2, .S1, .S2) have an extra 8-bit wide port for 40-bit long writes as well as an 8-bit input for 40-bit long reads. Because each unit has its own 32-bit write port, all eight units can be used in parallel every cycle.

Table 2–1. Functional Units and Descriptions

Functional Unit	Description
.L Unit (.L1,.L2)	32/40-bit arithmetic and compare operations Left most 1, 0, bit counting for 32 bits Normalization count for 32 and 40 bits 32-bit logical operations
.S Unit (.S1, .S2)	32-bit arithmetic operations 32/40-bit shifts and 32-bit bit-field operations 32-bit logical operations, Branching Constant generation Register transfers to/from the control register file
.M Unit (.M1, .M2)	16 x 16-bit multiplies
.D Unit (.D1, .D2)	32-bit add, subtract, linear and circular address calculation

2.3.3 Register File Cross Paths

Each general-purpose register file is connected to the opposite register file's functional units by the 1X and 2X paths. These paths allow the .S, .M, and, .L units from each side to access operands from either file.

Four units (.M1, .M2, .S1, .S2), have one 32-bit input mux selectable with either the same side register file (A for units ending in a 1 and B for units ending in a 2), or the opposite file via the cross paths (1X and 2X). The 32-bit inputs on the .L1 and .L2 units are both mux selectable via the cross paths.

2.3.4 Memory, Load, and Store Paths

There are two 32-bit paths for loading data from memory to the register file: one (LD1) for register file A, and one (LD2) for register file B. There are also two 32-bit paths, ST1 and ST2, for storing register values to memory from each register file. The store paths are shared with the .L and .S long read paths.

2.3.5 Data-Address Paths

The data-address paths (DA1 and DA2) coming out of the .D units allow data addresses generated from one register file to support loads and stores to memory from the other register file.

Figure 2–2. TMS320C62xx CPU Data Paths
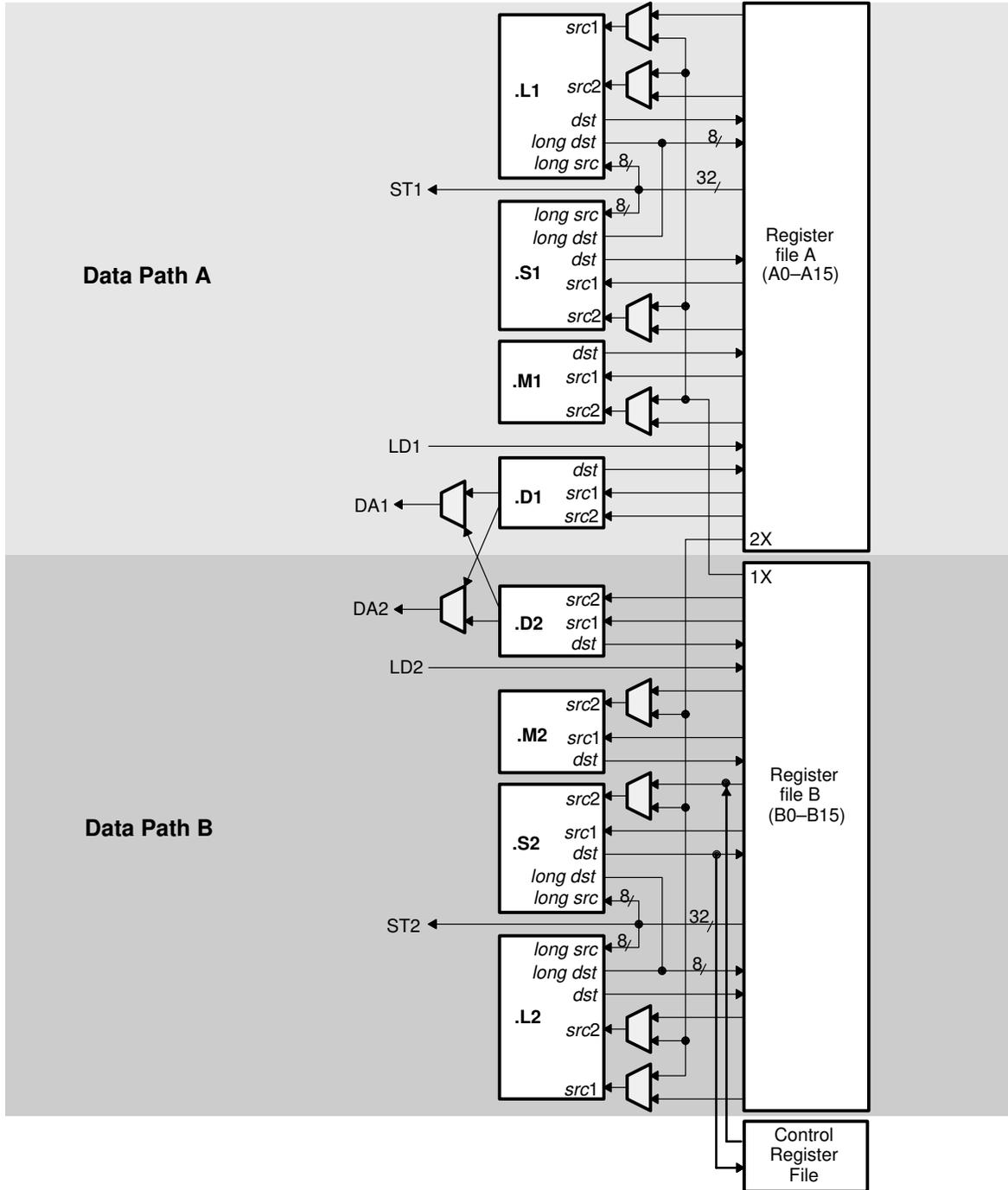
2.4 Mapping Between Instructions and Functional Units

Table 2–2 and Table 2–3 define the mapping between instructions and functional units. The first table lists the instructions that can be used on each functional unit. The second table lists the instructions alphabetically with the functional unit where each instruction can be used checked under the units.

Table 2–2. Instruction to Functional Unit Mapping

.L Unit	.M Unit	.S Unit	.D Unit
ABS	MPY	ADD	ADD
ADD	SMPY	ADDK	ADDA
AND		ADD2	LD mem
CMPEQ		AND	LD mem (15-bit offset)‡
CMPGT		B disp	MV
CMPGTU		B IRP†	NEG
CMPLT		B NRP†	ST mem
CMPLTU		B reg	ST mem (15-bit offset)‡
LMBD		CLR	SUB
MV		EXT	SUBA
NEG		EXTU	ZERO
NORM		MVC†	
NOT		MV	
OR		MVK	
SADD		MVKH	
SAT		NEG	
SSUB		NOT	
SUB		OR	
SUBC		SET	
XOR		SHL	
ZERO		SHR	
		SHRU	
		SSHL	
		STP†	
		SUB	
		SUB2	
		XOR	
		ZERO	

† .S2 only
‡ .D2 only

Table 2–3. Functional Unit to Instruction Mapping

Instruction	C62xx Functional Units			
	.L Unit	.M Unit	.S Unit	.D Unit
ABS	✔			
ADD	✔		✔	✔
ADDA				✔
ADDK			✔	
ADD2			✔	
AND	✔		✔	
B			✔	
B IRP			✔†	
B NRP			✔†	
B reg			✔†	
CLR			✔	
CMPEQ	✔			
CMPGT	✔			
CMPGTU	✔			
CMPLT	✔			
CMPLTU	✔			
EXT			✔	
EXTU			✔	
IDLE				
LD mem				✔
LD mem (15-bit offset)				✔‡
LMBD	✔			
MPY		✔		
MVC†			✔	
MV	✔		✔	✔
MVK			✔	

† .S2 only
‡ .D2 only

Table 2–3. Functional Unit to Instruction Mapping (Continued)

	C62xx Functional Units			
Instruction	**.L Unit**	**.M Unit**	**.S Unit**	**.D Unit**
MVKH			✔	
NEG	✔		✔	✔
NOP				
NORM	✔			
NOT	✔		✔	
OR	✔		✔	
SADD	✔			
SAT	✔			
SET			✔	
SHL			✔	
SHR			✔	
SHRU			✔	
SMPY		✔		
SSHL			✔	
SSUB	✔			
ST mem				✔
ST mem (15-bit offset)				✔‡
STP			✔†	
SUB	✔		✔	✔
SUBA				✔
SUBC	✔			
SUB2			✔	
SWI				
XOR	✔		✔	
ZERO	✔		✔	✔

† .S2 only
‡ .D2 only

2.5 Addressing Modes

The addressing mode options on the C62xx are linear, circular using BK0, and circular using BK1. The mode is specified by the addressing-mode register (AMR).

Eight registers can perform circular addressing. A4-A7 are used by the .D1 unit and B4-B7 are used by the .D2 unit. No other units can perform circular addressing modes. For each of these registers, the AMR specifies the addressing mode.

LD(B)(H)(W), **ST(B)(H)(W)**, **ADDA(B)(H)(W)**, and **SUBA(B)(H)(W)** instructions all use the AMR to determine what type of address calculations are performed for these registers. All registers can perform linear mode addressing.

For more information on addressing modes, see the *TMS320C62xx CPU and Instruction Set Reference Guide* (literature number SPRU189).

2.6 Interrupts

The 'C6200 CPU has 14 interrupts. These are reset, the non-maskable interrupt (NMI), and interrupts 4–15. These interrupts correspond to the $\overline{RESET}$, NMI, and INT4–INT15 signals on the CPU boundary. In some 'C62xx devices these signals may be tied directly to pins on the device, connected to on-chip peripherals, or may be disabled permanently by being tied inactive on-chip. Generally, RESET and NMI are connected directly to pins on the device.

For more information on interrupts, see the *TMS320C62xx CPU and Instruction Set Reference Guide* (literature number SPRU189).
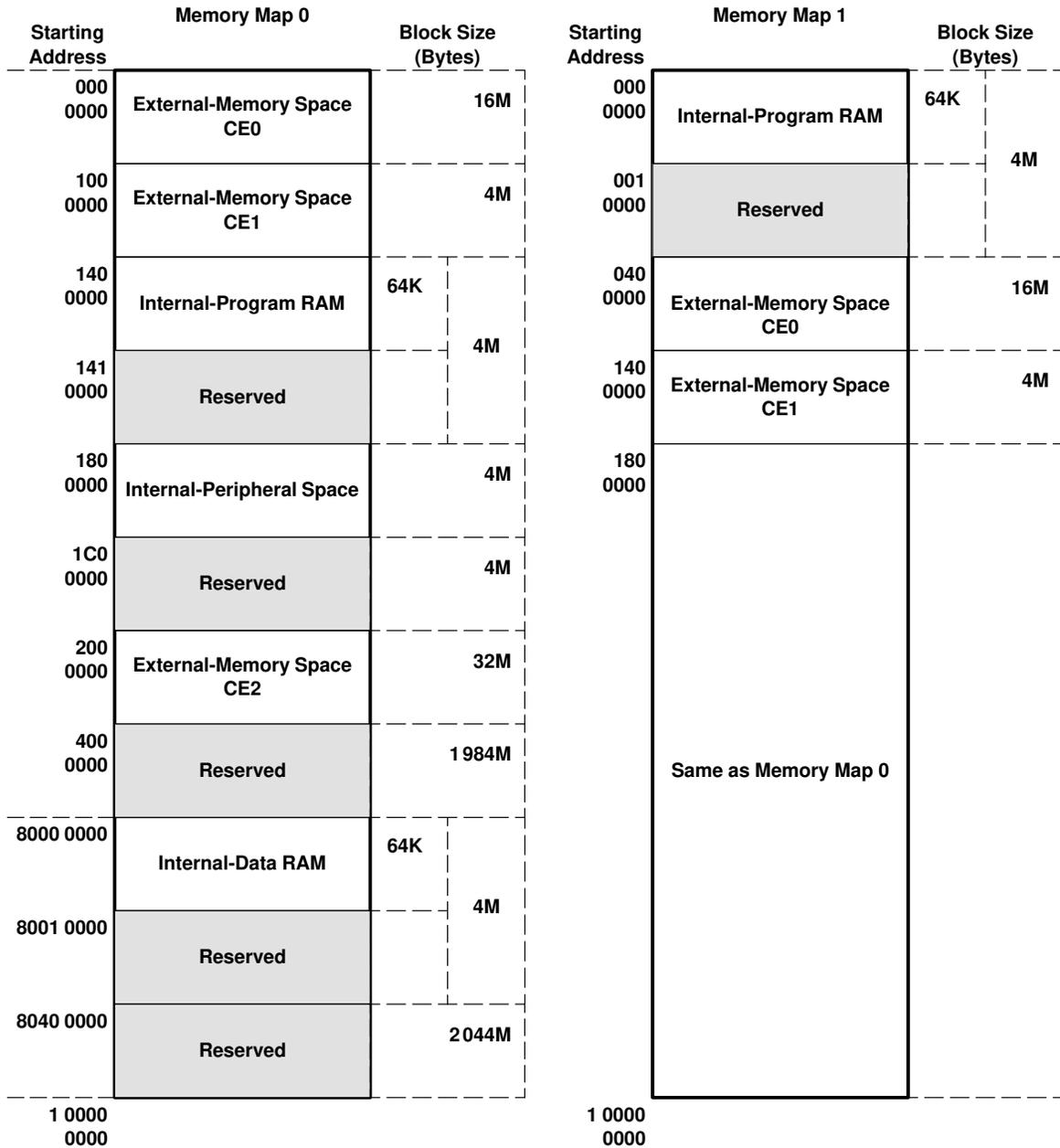
Chapter 3

Memory

The TMS320C62xx devices come with on-chip memory that can be selected for use as program memory or program cache. The device is available with varying sizes of data memory. When off-chip memory is used, the external memory interface (EMIF) can unify these spaces to a single memory space on most devices.

Topic	Page

3.1 Memory Map

Figure 3–1 shows the memory map of the TMS320C6201 DSP. The total memory address range of the 'C6201 is 4M bytes (corresponding to 32-bit internal address representation). The memory map is divided between the internal-program memory, internal-data memory, three external memory spaces, and internal-peripheral space.

Figure 3–1. Memory Map of the TMS320C6201

Memory Map 0		Memory Map 1	

Memory Map 0

Starting Address		Block Size (Bytes)
000 0000	External-Memory Space CE0	16M
100 0000	External-Memory Space CE1	4M
140 0000	Internal-Program RAM	64K
141 0000	Reserved	4M
180 0000	Internal-Peripheral Space	4M
1C0 0000	Reserved	4M
200 0000	External-Memory Space CE2	32M
400 0000	Reserved	1984M
8000 0000	Internal-Data RAM	64K
8001 0000	Reserved	4M
8040 0000	Reserved	2044M
1 0000 0000		

Memory Map 1

Starting Address		Block Size (Bytes)
000 0000	Internal-Program RAM	64K
001 0000	Reserved	4M
040 0000	External-Memory Space CE0	16M
140 0000	External-Memory Space CE1	4M
180 0000	Same as Memory Map 0	
1 0000 0000		

3.2 Internal Memory

The internal (on-chip) memory is organized into separate data and program spaces. The 'C62xx has two internal ports to access data memory, each with a 32-bit data and 32-bit byte-address reach. It has a single port to program memory, with an instruction fetch width of 256 bits and a 30-bit word (four byte) address, equivalent to a 32-bit byte address.

3.2.1 Data-Memory System

The TMS320C6201 data-memory system includes a 64K-bytes of SRAM and a memory controller. The TMS320C6201 CPU can access data memory in 8-bit byte, 16-bit halfword, and 32-bit word-lengths. The data memory system supports two memory accesses in a cycle. These accesses can be any combination of loads and stores from the two data buses of the CPU. Similarly, a simultaneous internal and external memory access is supported by the data memory system. The TMS320C6201 data memory system also supports direct memory access (DMA) and external host accesses. For more information on the DMA operation, see the *TMS320C62x CPU and Reference Guide*.

The data memory is organized into four banks of 16-bit wide memory. This interleaved memory organization provides a method for two simultaneous memory accesses. Occurring in one cycle, two simultaneous accesses to two different internal memory banks provide the fastest access speed.

3.2.2 Program-Memory System

The TMS320C6201 program-memory system includes 64K bytes of on-chip SRAM and a memory/cache controller. The program memory can operate as either a 64K-byte internal program memory or as a directly mapped program cache. There are four modes under which the TMS320C6201 program memory system operates:

❑ Program-memory mode
❑ Cache-enable mode
❑ Cache-freeze mode
❑ Cache-bypass mode

The DMA can write data into an addressed space of program memory. The DMA cannot read from the internal program memory in program memory mode.

When the program memory is used to cache external program data, the memory is no longer in valid memory space and cannot be directly addressed; therefore, the DMA cannot write or read the internal program memory in any

cache mode. The caching scheme implemented in the TMS320C6201 program cache is a direct mapping of external program memory addresses. This means that any external address map to only one cache location, and addresses which are 64K bytes apart map to the same cache location. The program cache is organized into 256-bit frames. Thus, each frame holds one fetch packet. The cache stores 2048 fetch packets.

A program store to external memory in any cache mode first flushes the data in the cache frame that is mapped to the target address directly to ensure data coherency in the cache. The data then is written to the external memory at the addressed location. When that address is again accessed a cache miss occurs causing the new data to be loaded from external memory.

On the change from program memory mode to cache-enabled mode, the program cache is flushed. During a cache freeze, the cache retains its current state. A program read to a frozen cache is identical to a read to an enabled cache with the exception that the data read from the external interface is not stored in the cache on a cache miss. When the cache is bypassed, any program read fetches data from external memory. The data is not stored in the cache memory. Like cache freeze, in cache bypass the cache retains its state. For details on cache modes, see the *TMS320C62xx Peripherals Reference Guide*

3.3   External Memory Interface (EMIF)

All external data accesses by the CPU or DMA pass through the external memory interface (EMIF). The EMIF is the interface between the CPU and external memory such as synchronous dynamic random-access memory (SDRAM), synchronous-burst static RAM (SBSRAM), and asynchronous memory. The EMIF also provides 8-bit and 16-bit wide memory read capability to support low-cost boot ROM memories (flash, EEPROM, EPROM, and PROM). The production version of the EMIF will support higher throughput interfaces to SDRAM, including burst capability.

The interface is programmable to adapt to a variety of setup, hold, and strobe widths for asynchronous devices. SBSRAM supports zero-wait state external access once bursts have begun.

In all of these types of access, the EMIF supports 8-bit, 16-bit, and 32-bit addressability for writes. All reads are performed as 32-bit transfers.

The EMIF can receive three types of requests for access. The three types are prioritized in this order: CPU data accesses, CPU program fetches, and DMA data accesses. When available to service another access, the EMIF services the request type of highest priority. For example, DMA requests are not serviced until the CPU ceases requesting external data and program fetches.

The major functions implemented by the EMIF are the following:

❑   Steering incoming data bytes and half-words to form word data (when reading from byte and half-word memories)

❑   Interfacing to the 'C6xx internal peripheral bus

❑   Interfacing to SBSRAM and SRAM, including computation of the next address

❑   Interfacing to asynchronous memories, using programmable timing

❑   Interfacing to SDRAM memories

❑   Handshaking with internal and external modules

The characteristics of the EMIF are as follow:

❑   Zero wait state operation, after an initial two clock cycle latency, with synchronous burst SRAM (currently available in 125 MHz speed grades).

❑   Support for little or no glue logic interface to asynchronous memory. Timing parameters can be programmed to match various asynchronous memories.

❑ Serialization between program memory system, data memory system, and DMA system.

❑ Support for sharing external memory with another processor.

❑ Support for reading byte and half-word memory devices (ROM) to facilitate boot up from these low cost devices.

The exact level of throughput to the 'C62xx is determined by the type of memory used, and the clock rate of the 'C62xx. For example, when the 'C62xx is running at 200 MHz, the maximum throughput would be 800 M-byte/second - assuming memory supporting that throughput.

Figure 3–2 shows a diagram of the 'C62xx external memory signals that are common to all interfaces.

Figure 3–2. External Memory Interface (EMIF) Block Diagram
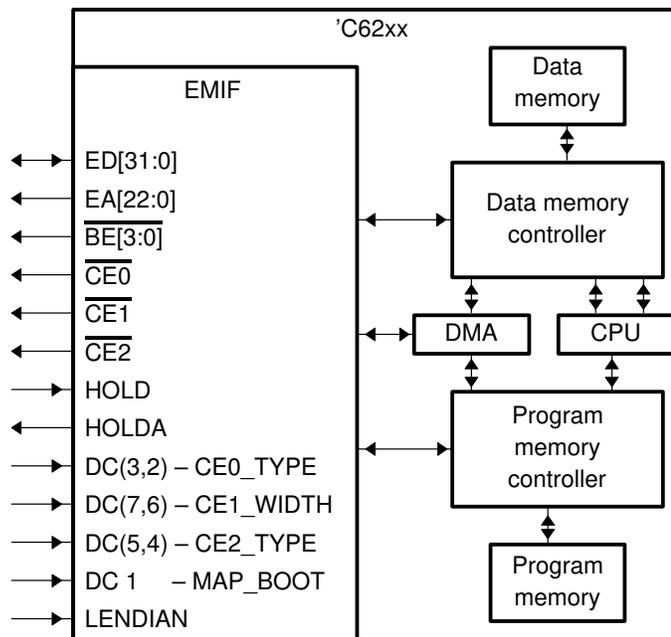
For more information on memory, see the *TMS320C62xx CPU and Instruction Set Reference Guide* (literature number SPRU189). For more information on the EMIF, see the *TMS320C62xx Peripherals Reference Guide* (literature number SPRU190).

Peripherals

In addition to on-chip memory, the TMS320C62xx devices also contain several peripherals for communication with off-chip memory, co-processors, and serial devices.

Peripherals for the TMS320C62xx devices may include:

❏ External memory interface (EMIF)

❏ Direct-memory access (DMA) controller

❏ Host-port interface (HPI)

❏ Power-down logic

❏ Enhanced-buffered serial ports (EBSPs)

❏ 32-bit timers

The first device in the TMS320C62xx series is the TMS320C6201. The production release of this device will include the EBSPs supporting multivendor interface protocol (MVIP) and timers to allow easy algorithm integration. The EBSP is based on the standard TMS320C2x/C5x/C54x serial port. In addition, it has the ability to buffer serial samples in memory automatically with the aid of the DMA. It also has multichannel capability compatible with the T1, E1, and MVIP standards.

Figure 4–1 shows the peripherals for the TMS320C62xx devices.

Figure 4–1. Peripherals Overview

EMIF
Glueless access to 1M/4M/8M blocks
EPROM, SRAM, SDRAM, SBSRAM

DMA
Two–channel transfer

Boot Loader
Pin–select at reset, done by DMA
Boot via SP or 1M external block

Serial Port
High speed sync serial comm
T1/E1 interface supporting MVIP

HPI
16-bit access to internal data RAM

Power-Down
Three IDLE modes

32-Bit Timer
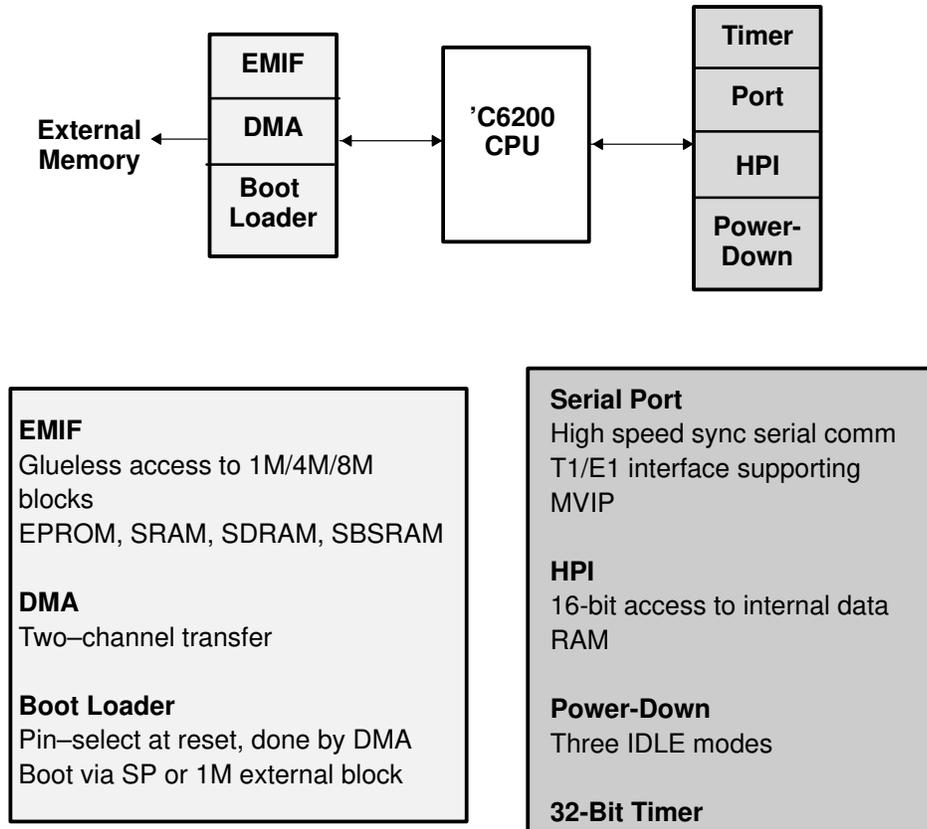
4.1 External Memory Interface (EMIF)

The EMIF is the interface between the CPU and external memory such as synchronous dynamic random-access memory (SDRAM), synchronous burst static RAM (SBSRAM), and asynchronous memory. The EMIF also provides 8-bit and 16-bit wide memory read capability to support low-cost boot ROM memories (flash, EEPROM, EPROM, and PROM). The final revision of the EMIF will support higher throughput interfaces to SDRAM including burst capability.

More information on the EMIF is provided in Chapter 3, Memory.

For more information on memory, see the *TMS320C62xx CPU and Instruction Set User's Guide*. For more information on the EMIF, see the *TMS320C62xx Peripherals Reference Guide*.

4.2 **Direct-Memory Access (DMA)**

The on-chip DMA offers two channels which can be configured to transfer information from one location in the memory map to another without interfering with the operation of the CPU. This allows interfacing to slow external memories and peripherals without reducing the throughput to the CPU. The DMA controller contains its own address generators, source and destination registers, and transfer counter. The DMA has its own bus for addresses and data. This keeps the data transfers between memory and peripherals from conflicting with the CPU. A DMA operation consists of a 32-bit word transfer to or from any of the three 'C62xx modules (see Figure 4–2):

❏ Internal Data Memory

❏ Internal Program Memory that is not configured as cache as a destination of a transfer

❏ EMIF

One of the channels can be used by the processor during the boot load startup procedure to initialize the internal program memory after reset. The DMA channels can be used to write to Internal program memory.

The boot loader uses the DMA to boot load code from off-chip memory to the internal program memory area. An external pin (sampled at reset) selects whether this boot load is performed. The serial port can also be used for booting.

The DMA controller can access all internal program memory, all internal data memory, and all devices mapped to the EMIF. An exception is that the DMA cannot use program memory as the source of a transfer. Also, it cannot access memories configured as cache or memory-mapped on-chip peripheral registers.

The DMA controller has the following features:

❏ Two independent channels

❏ Source and destination addresses may be within the same or different modules. These addresses are programmable independently, and can remain constant, increment, or decrement on each transfer.

❏ The transfer count is programmable. Once the transfer count has completed, the DMA can send an interrupt to the CPU.

Figure 4–2. DMA Controller Interconnect to 'C62xx Memory Mapped Modules
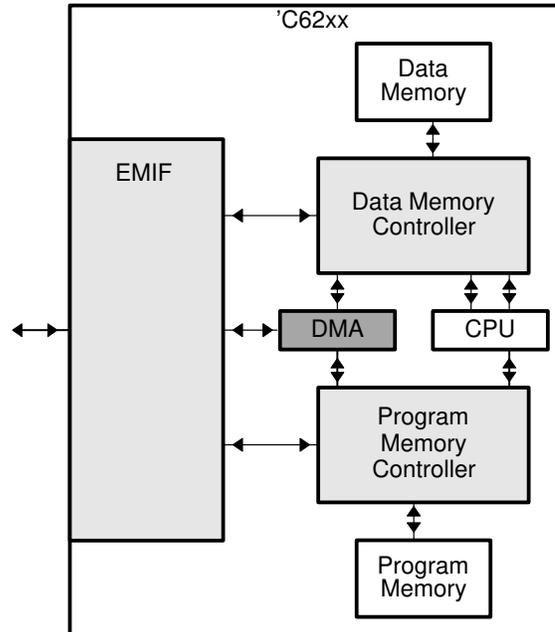
The DMA has lowest priority to all modules it accesses and it must wait until no transfers are being initiated to the internal data and program memory it intends to access. DMA accesses to internal memory perform cycle stealing; therefore, no subsequent CPU accesses of internal memory are hampered by a DMA access. However, if the CPU accesses the EMIF while a multi-cycle DMA access is in progress, it must wait until that access completes.

Each DMA channel has an independent set of registers that must be programmed to control the operation of the DMA.

See the data sheet for the specific device to find the memory mapping of DMA control registers. These registers are 2-bits wide and must be accessed through 32-bit accesses from the CPU.

4.3 Host-Port Interface (HPI)

The host-port interface (HPI) operates as a straightforward asynchronous interface. The HPI is a 16-bit wide access port through which a host (external) processor can read from, and write to, the 'C62xx's internal data memory.

A host processor access to the 'C62xx internal data memory through the HPI consists of two operations, which follow:

1) The host must gain control over the HPI by performing the request/acknowledge handshake through the HREQ/HACK signals.

2) Once access has been granted, the host may perform read and write operations to the 'C62xx internal data memory.

The mapping of host-port address to the 'C62xx internal memory address is described in the data sheet for your 'C62xx device.

The HPI on the production release of the TMS320C6201 will have the ability to boot load the CPU as well as access the full range of the 'C6201's memory. Also, the HPI will offer improved performance and will be capable of operating without impacting CPU performance.

For more information on the host port, see the *TMS320C6xx Peripheral Reference Guide* (literature number SPRU190).

4.4 Power-Down Logic

The 'C62xx supports three power-down modes that can reduce system power requirements significantly. The modes are as follow:

❑ Idle1
❑ Idle2
❑ Idle3

The three lower bits of the power-down field in the control status register (CSR) are used to initiate the three power-down modes. If more than one of these power-down bits are set, the power-down mode is selected by the most significant bit enabled.

When in a power-down mode, the 'C62xx can be reactivated by a reset, an enabled interrupt, or any interrupt. Bits three and four of the power-down field in the control status register set the wake-up condition.

The power-down mode bit and wake up bit must be set by the same instruction to ensure proper power-down operation. See the *TMS320C6xx Peripheral Reference Guide* (literature number SPRU190) for more details on the power-down logic

Development Support

The TMS320C6x design environment reflects the unique nature of the advanced VLIW architecture. The environment includes code-generation tools, evaluation tools, documentation, on-line help with various tools, and a Web site on the Internet (www.ti.com/sc/C6x) with complete technical documentation.

5.1 Code-Generation Tools

A complete development tool set for both the PC and Sun workstations includes the following:

❑ C Compiler
❑ Assembly optimizer
❑ Linker

The environment is founded on the generation's highly advanced C compiler and TI's revolutionary assembly optimizer. Figure 5–1 shows a flow of the process to develop code.

The 'C6x generation's C compiler eliminates the need for extensive knowledge of DSP architecture, allowing you to take full advantage of the world's most powerful DSP. This highly-structured, architecture-independent C code development environment dramatically reduces development time for new products. At the same time, it maintains the inherent performance benefits of the 'C6x generation's advanced VLIW architecture. The 'C6x compiler offers a 3X improvement in efficiency over existing fixed-point C compilers for DSP.

For application code sections that require the fine tuning of assembly code, the 'C6x generation's unique assembly optimizer provides the same transparent programming capability as the C compiler. The tool supports automatic scheduling, optimizing, and separation of fine-grained parallel tasks from serial, in-line assembly code – delivering a level of simplicity and power that is unprecedented in assembly-level tools.

The tools take C or assembly source code and implement many different optimizations, including software pipelining, to intelligently find and exploit the unique instruction-level parallelism of the 'C6x. After each step in the process, the 'C6x tools allow you to evaluate their results and take appropriate steps to achieve the most parallel code.

Initially, all C code — new or reused from other applications — is run through the C compiler for the 'C6x. Using the evaluation tools described in the following section, you can evaluate the code for efficiency. If the performance is sufficient for the particular application, then the application has been completed, achieving the fastest possible time-to-market and incurring minimal engineering cost.

Figure 5–1. Code-Development Flow Chart
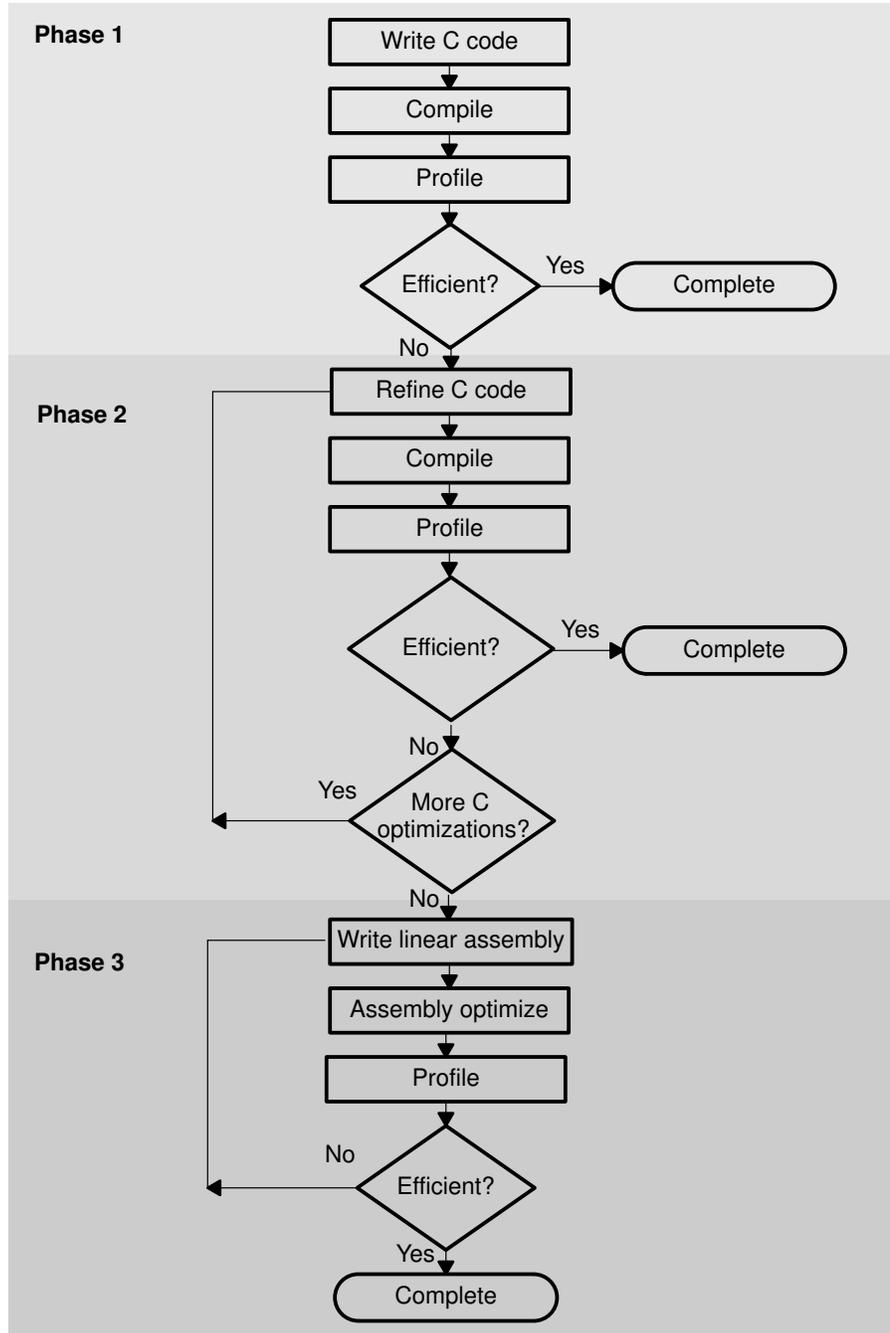
A designer who needs to improve code efficiency can use intrinsics, command-line options and source-code enhancements as a first step:

❑ The 'C6x design tools feature two sets of intrinsics. The first set includes intrinsics that perform DSP-specific operations on instructions that are not supported directly in C. The second set is designed to facilitate 16-bit operation on a 32-bit machine. These intrinsic functions can be invoked to tune the performance of the C code.

❑ The designer can experiment with several command-line options that cause the compiler to perform more aggressive optimization. One particularly useful option instructs the compiler to compile an entire application at once, giving the compiler visibility across program sections and more knowledge of the way in which variable and functions are used. Another option causes the compiler to perform global optimizations across an entire application.

❑ Source-code enhancements can be made to exploit specific features of the 'C6x architecture. For example, the 'C6x has support for operating on words containing two 16-bit quantities; therefore, you can utilize 32-bit loads and stores when operating on arrays containing 16-bit data and easily achieve a 2X performance improvement.

Taken together, these actions extract a large amount of parallelism from C code.

For ultra-high performance applications, extracting every last bit of throughput from the application code may be necessary. The profiler can identify critical code segments that might benefit most from being generated in assembly language.

For these program sections, the designer writes simple, linear 'C6x assembly code that is input to TI assembly optimizer. This assembly code is 'C6x instructions written without concern for parallel instructions, instruction latencies, or register usage.

The assembly optimizer tool schedules the instructions, taking into account the architectural parallelism. The tool honors 'C6x latency requirements, maximizes parallel code, and performs register allocation.

5.2 Evaluation Tools

The evaluation tools include the following:

❏ Windows-based debugger interface
❏ Simulator
❏ Hardware emulation board

The 'C6x development environment provides a new intuitive Windows™-based graphical user interface (GUI) for debugging. The debugger interface features windows for source, assembly, call stack, memory, registers, and watch expressions as well as menu and tool bars. The debugger offers one-click breakpoint setting and dialogs for editing breakpoint. The debugger also incorporates the dynamic profiler to help users find bottlenecks and improve code efficiency.

TI will provide 'C6201 scan-based emulation systems that support hardware and software debugging of target systems via a JTAG-emulation cable. Scan-based emulation is a unique, non-intrusive approach to system emulation, integration, and debugging.

Initially, TI is offering a stand-alone 'C6201 test and emulation board (TEB) that interfaces with the host platform through the XDS510™ and XDS 510WS emulators through the IEEE Standard 1149.1 (JTAG)-compliant port. The board features a prototyping area for adding user-defined peripherals. With the addition of other 'C6x generation members, TI will continually add functionality to the common development environment as well. Capabilities will ultimately include a PC plug-in evaluation module (EVM) board, a low-cost PC-based board that is well-suited for software algorithm development.

The dynamic profiler integrated into the 'C6x debugger creates cycle histograms that are continuously updated as the code runs. It can show graphically which functions, ranges and lines in an application are performance bottlenecks. The profiler can show:

❏ The percentage of total execution time spent in any function
❏ The number of times a function is called
❏ Total cycles in the application, a function, or a line

A timing display can be built into the application by inserting a few function calls in the code. The resulting simple cycle counts, obtained without using the profiler or the debugger, can be printed automatically to allow you to track the changes in execution speed of an algorithm over time. This output, while less sophisticated, is continuously available with no further action.

5.3 Third-Party Support

TI has a long history of strong third-party support and this continues with the 'C62xx devices. Table 5–1 lists some of the companies supporting the 'C62xx devices and the product areas. Table 5–2 lists the third-party support contacts with telephone numbers and electronic mail addresses.

Table 5–1. Third-Party Support Companies and Product Area Supported

Company	Product Area
Ariel Corporation	High–performance VME64 platform and computer telephony products
Cheops GmbH & Co KG	Industrial and medical imaging and high speed/high resolution videoconferencing
D2 Technologies, Inc.	Embedded Voice Processing (EVP™) computer telephony software
DSP Research, Inc.	TIGER development boards and OEM systems
DSP Software Engineering, Inc.	Multi-channel V.34bis soft–modem and telecom software
Eonic Systems, Inc.	Real-time operating systems – Virtuoso Nano™, Classico™, and MicroLite™
Go DSP Corporation	Code Composer™ support and next generation development tool, Code Maestro™
HotHaus Technologies, Inc.	HausWare – DSP software architecture for embedded telecommunications applications
Innovative Integration	PCI6201 DSP coprocessor for telecom, communications and data acquisition applications
Loughborough Sound Images	PCI/C6200 – signal processing platform and PCI/C6220 telecommunications/high density DSP telephony platform
Pentek, Inc.	Scaleable multi–processor board for the VMEbus – model 9134
Signals & Software Ltd. (SASL)	Very high density ISP modem solution
ViaDSP, Inc.	InvisiLink™ – line of software and firmware for high density computer telephony boards
White Mountain DSP, Inc.	Emulation and multiplatform debug support –Mountain–510, Mountain–510/WS and Mountain–510/LT PCMCIA Card

Table 5–2. Contacts for Third-Party Support

Third-Party Contact	Phone number	e-mail address
Ariel Corporation	609 860–2900	ariel@ariel.com
Cheops GmbH & Co KG	+49 8861 2369 0	100541.3370@compuserve.com
D2 Technologies, Inc.	805 564–3424	sales@d2tech.com
DSP Research, Inc.	408 773–1042	info@dspr.com
DSP Software Engineering, Inc.	617 275–3733	info@dspse.com
Eonic Systems, Inc.	301–572–5000	info@eonic.com
GO DSP Corporation	416 599–6868	gdasilva@go–dsp.com
HotHaus Technologies, Inc.	604–278–4300	info@hothaus.com
Innovative Integration, Inc.	818–865–6150	techsprt@innovative-dsp.com
Loughborough Sound Images	+44 0 1509 634444	
Pentek, Inc	201–818–5900	info@pentek.com
Signals & Software Ltd. (SASL)	+44 181 426 9533	davem@sasl.demon.co.uk
ViaDSP, Inc.	508–369–0048	dpenny@viadsp.com
White Mountain DSP, Inc.	603 883–2430	info@wmdsp.com

5.4 Web Site and Documentation

Visit the Web site at **www.ti.com/sc/C6x** for information, an interactive multimedia technical overview (MeTO), documentation, and schedule of 'C6x design workshops. The MeTO describes features of the devices in a visual way with graphics in a point-and-click display for ease of navigation. The Web site offers a complete training schedule of design workshops and seminars. Applications assistance and frequently asked questions (FAQ), are also on the Web site.

Documentation is available directly from the Web site in down-loadable files for printing. There is a complete list of documentation available in the Preface under *Related Documentation from Texas Instruments*.

Glossary

A

address: The number of a particular memory or peripheral storage location.

ALU: *Arithmetic logic unit*. The high-speed CPU circuit that does calculating and comparing. Numbers are transferred from registers into the ALU for calculation, and the results are sent back to a register.

ASIC: *Application–specific integrated circut*. A custom chip designed for a specific applicaiton. It is designed by integrating standard cells from a library.

Assembler: A software program that creates a machine-language program from a source file that contains assembly language instructions, directeives, and macro definitions. The assembler substitutes absolute opertaion dcodes for symbolic operation codes .

Assembly Optimizer: A software program that optimizes linerar assembly code, which is assembly code that has not been register-allocated or scheduled. The assembly optimizer is automatically invoked with the shell program, C/6x, when one of the input files has a .sa extension.

B

boot loader: A built-in segment of code that transfers code from an external source to program memory at power up.

C

clock cycles: Cycles based on the input from the external clock.

code: A set of instructions written to perform a task; a computer program or part of a program.

CPU: *Central processing unit*. The unit that coordinates the functions of a processor.

D

Data Memory: Memory accessed through the 'C6x's RAM interface.

DMA: *Direct memory access.* Specialized circuitry that transfers data from memory to memory without using the CPU.

DRAM: *Dynamic random access memory.* The most common type of computer memory.

E

EBSP: *Enhanced buffered serial ports*.

EMIF: *External memory interface*.

execute packet: A packet of instructions that execute in parallel.

external interrupt: A hardware interrupt triggered by a pin.

F

fetch packet: A packet containing up to eight instructions held in memory for execution by the CPU.

G

global interrupt enable (GIE): A bit in the control status register (CSR). Used to enable or disable maskable interrupts.

H

hardware interrupt: An interrupt triggered through physical connections with on-chip peripherals or external devices.

HPI: Host port interface

I

interrupt: A condition caused either by an event external to the CPU or by a previously executed instruction that forces the current program to be suspended and causes the processor to execute an interrupt service routine corresponding to the interrupt.

interrupt service fetch packet (ISFP): A fetch packet used to service interrupts. If 8 instructions are insufficient, the user must branch out of this block for additional interrupt service. If the delay slots of the branch do not reside within the ISFP, execution continues from execute packets in the next fetch packet (the next ISFP).

ISFP: *Interrupt service fetch packet.*

IFP: *Instruction fetch packet.*

L

latency: The delay between when a condition occurs and when the device reacts to the condition. Also, in a pipeline, the necessary delay between the execution of two instructions to ensure that the values used by the second instruction are correct.

LSB: *least significant bit*. The lowest order bit in a word.

M

maskable interrupt: A hardware interrupt that can be enabled or disabled through software.

memory interleaving: A category of techniques for increasing memory speed.

MIPS: *Million instructions per second*. The execution speed of a computer.

MSB: *most significant bit*. The highest order bit in a word.

N

NMI: Non–maskable interrupt

nonmaskable interrupt: An interrupt that can be neither masked nor disabled.

O

overflow: A condition in which the result of an arithmetic operation exceeds the capacity of the register used to hold that result.

P

pipeline: A method of executing instructions in an assembly-line fashion.

pipeline processing: A category of techniques that provide simultaneous, or parallel, processing within the computer. It refers to overlapping operations by moving data or instructions into a conceptual pipe with all stages of the pipe processing simultaneously.

PLL: *Phase-locked loop.*

program memory: Memory accessed through the C6x's program fetch interface.

R

RAM: *Random-access memory.*

register: A group of bits used for temporarily holding data or for controlling or specifying the status of a device.

reset: A means of bringing the CPU to a known state by setting the registers and control bits to predetermined values and signaling execution to start at a specified address.

RISC: *Reduced instruction set computing.* A computer architecture that reduces chip complexity by using simpler instructions.

S

SBSRAM: *Synchronous burst static random-access memory.*

SDRAM: *Synchronous dynamic random-access memory.*

shifter: A hardware unit that shifts bits in a word to the left or to the right.

V

VelociTI: Architecture developed by TI that features very long instruction words

VLIW: *Very long instruction word.*

Index

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.