TMS320C8x System-Level Synopsis

SPRU113B
September 1995

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

The TMS320C8x is Texas Instruments' first generation of single-chip multiprocessor digital signal processor (DSP) devices. A single 'C8x contains up to five powerful, fully programmable processors: a master processor (MP) and up to four parallel processors (PPs). The MP is a 32-bit RISC (reduced instruction set computer) processor with an integral, high-performance IEEE-754 floating-point unit. Each PP is a 32-bit advanced DSP that combines capabilities similar to those of conventional DSPs with advanced features to accelerate operation on a variety of data types.

The 'C8x supports a variety of parallel-processing configurations, which facilitate a wide range of multimedia and other applications that require high processing speeds. Applications include image processing, two-dimensional, three-dimensional, and virtual reality graphics, digital audio and video compression, and telecommunications.

This manual describes the 'C8x's features, architecture, and development environment.

Related Documentation From Texas Instruments

The following books describe the TMS320C8x 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.

TMS320C80 Multimedia Video Processor Data Sheet (literature number SPRS023) describes the features of the ’C80 device and provides pinouts, electrical specifications, and timings for the device.

TMS320C80 (MVP) C Source Debugger User’s Guide (literature number SPRU107) describes the ’C8x master processor and parallel processor C source debuggers. This manual provides information about the features and operation of the debuggers and the parallel debug manager; it also includes basic information about C expressions and a description of progress and error messages.

TMS320C80 (MVP) Code Generation Tools User’s Guide (literature number SPRU108) describes the ’C8x code generation tools. This manual provides information about the features and operation of the linker and the master processor (MP) and parallel processor (PP) C compilers and assemblers. It also includes a description of the common object file format (COFF) and shows you how to link MP and PP code.

TMS320C80 (MVP) Master Processor User’s Guide (literature number SPRU109) describes the ’C8x master processor (MP). This manual provides information about the MP features, architecture, operation, and assembly language instruction set; it also includes sample applications that illustrate various MP operations.

TMS320C80 (MVP) Multitasking Executive User’s Guide (literature number SPRU112) describes the ’C8x multitasking executive software. This manual provides information about the multitasking executive software features, operation, and interprocessor communications; it also includes a list of task error codes.

TMS320C80 (MVP) Parallel Processor User's Guide (literature number SPRU110) describes the 'C8x parallel processor (PP). This manual provides information about the PP features, architecture, operation, and assembly language instruction set; it also includes software applications and optimizations.

TMS320C80 (MVP) Transfer Controller User's Guide (literature number SPRU105) describes the 'C80 transfer controller (TC).This manual provides information about the TC features, functional blocks, and operation; it also includes examples of block write operations for big- and little-endian modes.

TMS320C80 (MVP) Video Controller User's Guide (literature number SPRU111) describes the 'C80 video controller (VC). This manual provides information about the VC features, architecture, and operation; it also includes procedures and examples for programming the serial register transfer (SRT) controller and the frame timer registers.

TMS320C80 to TMS320C82 Software Compatibility User's Guide (literature number SPRU154) describes programming differences in the 'C80 and the 'C82.

If You Need Assistance. . .

If you want to. . .	**Do this. . .**
Request more information about Texas Instruments Digital Signal Processing (DSP) products	Write to: Texas Instruments Incorporated Market Communications Manager, MS 736 P.O. Box 1443 Houston, Texas 77251–1443
Order Texas Instruments documentation	Call the TI Literature Response Center: **(800) 477–8924**
Ask questions about product operation or report suspected problems	Call the DSP hotline: **(713) 274–2320** **FAX: (713) 274–2324**
Report mistakes in this document or any other TI documentation	Fill out and return the reader response card at the end of this book, or send your comments to: Texas Instruments Incorporated Technical Publications Manager, MS 702 P.O. Box 1443 Houston, Texas 77251–1443 Electronic mail: **comments@books.sc.ti.com**

Trademarks

Windows NT is a trademark of Microsoft Corporation.

EPIC is a trademark of Texas Instruments Corporation.

Contents

Figures

Chapter 1

Overview of the TMS320C8x

The TMS320C8x digital signal processor (DSP) is a single-chip, multiprocessor device for use in image processing, two-dimensional, three-dimensional/virtual-reality graphics, audio/video digital compression, and many other applications. The ’C8x offers the power to support applications in the security, image recognition, digital telecommunications, and other markets.

The ’C8x architecture’s high degree of on-chip integration allows you to replace multiple ASICs, RISC processors, DSPs, and their corresponding SRAM and interface chips with one or more ’C8x devices.

Topics

1.1 What Is the TMS320C8x?

The TMS320C8x integrates several components onto a single chip:

- Up to five powerful, fully programmable processors
- Up to 50 KB of SRAM
- An intelligent DMA (direct memory access) controller with a direct external interface to DRAM, SDRAM, SRAM, and VRAM

Versions of the ’C8x are capable of performing the equivalent of over two billion RISC-like operations per second (BOPS). In some applications, one ’C8x can do the job of over ten of the most powerful DSPs or general-purpose processors. During each second of processing, the ’C80 can move up to 2.4 GB of on-chip data, up to 1.8 GB of on-chip instructions, and up to 400 MB of data to off-chip memory.

The ’C8x architecture contains up to four parallel processors and one master processor. The master processor (MP) is a 32-bit RISC (reduced instruction set computer) with an integral IEEE-754 floating-point unit. The MP’s general-purpose processing capabilities make it ideal for coordinating on-chip processing activities and for communicating with external devices.

Each parallel processor (PP) is an advanced 32-bit DSP. In addition to having the processing capabilities of conventional DSPs, a PP has special features to speed up operations on image data and to accelerate applications that manipulate data organized into bit fields. A PP’s wide instruction word, three-operand arithmetic logic unit (ALU), and single-cycle multiplier enable it to perform a number of RISC-like operations in a single clock cycle.

Communication between the MP and PPs takes place over a high-speed crossbar network that provides simultaneous access to multiple banks of on-chip RAM. Software tools allow you to write programs in C and assembly language for the MP and PPs.

In addition to the fully programmable processors, the chip contains an intelligent DMA controller, called the transfer controller (TC). The TC manages all memory traffic by performing several services. One service takes the form of packet transfers that move data between two memory regions. Some packet transfers are complex programmable byte-aligned array transfers that involve XY or linear addressing of a source or destination array.

To provide on-chip storage, the ’C80 offers 50 KB of SRAM and the ’C82 offers 44 KB of SRAM. On-chip RAM, with the exception of the caches, is shared among the processors to support a variety of parallel-processing configurations.

This unique combination of processing hardware facilitates a wide range of DSP applications that demand high processing speeds.

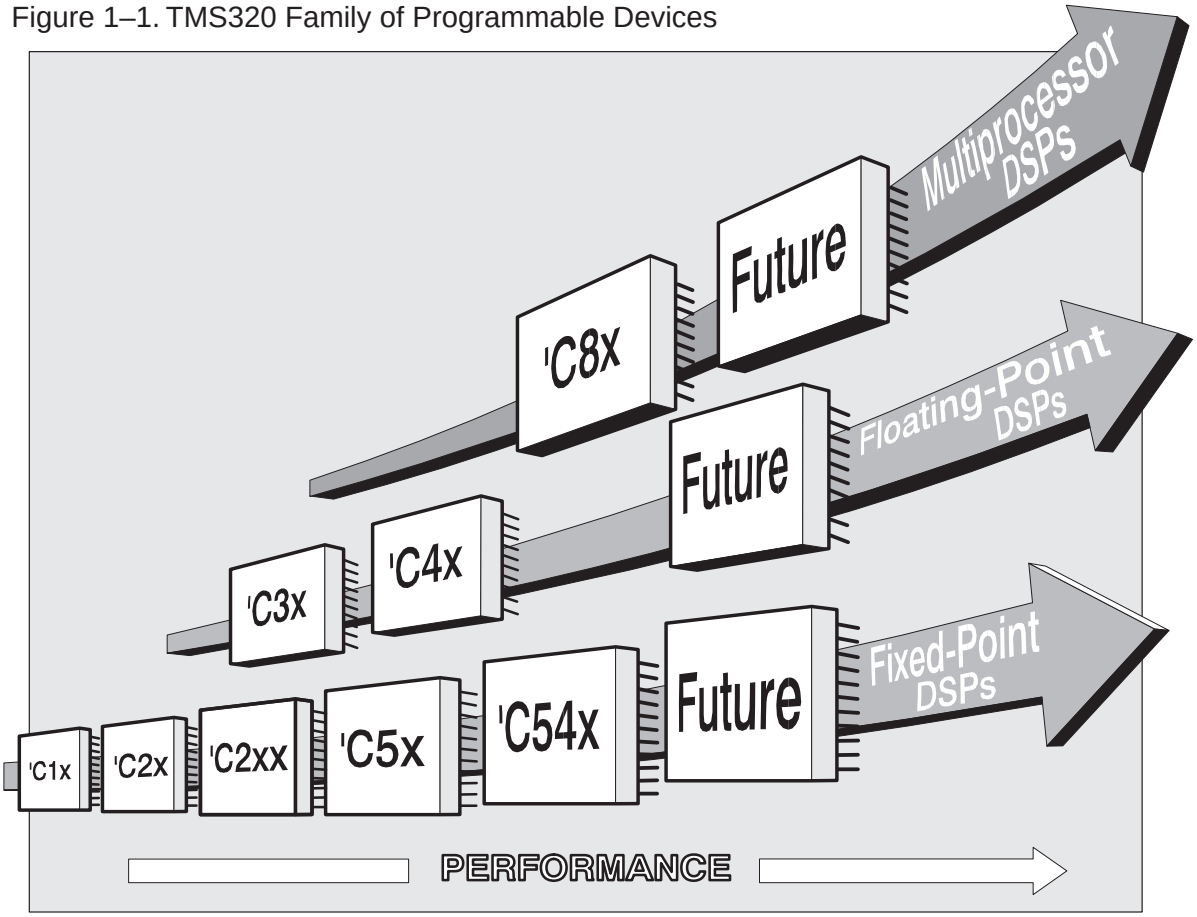
1.1.1 The TMS320 Family

The TMS320C8x generation is the flagship in the TMS320 family of DSPs. The family's scalable architecture allows Texas Instruments to offer a variety of devices at various cost and performance points. Eight generations make up the TMS320 family:

- The 'C1x, 'C2x, 'C2xx, C5x, and 'C54x generations offer a complete line of general-purpose and application-specific fixed-point DSPs.
- The 'C3x and 'C4x generations provide an ensemble of floating-point DSPs.
- The 'C8x generation offers multiprocessing capabilities.

Figure 1–1 illustrates the TMS320 family of devices.

Figure 1–1. TMS320 Family of Programmable Devices



The TMS320 family has grown from a single device introduced in 1982, the TMS32010, to over thirty different products across six CPU architectures. On-chip hardware multipliers, register files, barrel shifters, ALUs, ROMs, RAMs, caches, and I/O peripherals, along with massive internal buses, all within a product as programmable as a general-purpose microprocessor, make TMS320 devices ideal for a broad range of computation-intensive applications.

1.1.2 Key features of the TMS320C8x

The high-performance ’C8x architecture offers several key features:

- Versions capable of over two billion operations per second (BOPS)
- 40 or 50 MHz performance
- A 32-bit RISC master processor with an integrated IEEE-754 floating-point unit
- Multiple 32-bit parallel processors (advanced DSPs) with 64-bit instruction words
  - Four parallel processors in the ’C80
  - Two parallel processors in the ’C82
- 50 KB of on-chip RAM on the ’C80 and 44 KB of on-chip RAM on the ’C82
- An on-chip crossbar that allows multiple instruction fetches and parallel data accesses during each cycle to support high transfer rates:
  - up to 4.2 GB/s transfer rates on the ’C80
  - up to 2.6 GB/s on the ’C82
- Big-endian and little-endian byte-ordering modes
- A 4-Gbyte memory address space
- A 64-bit transfer controller capable of up to 400 MB/s in on-chip and off-chip memory transfers
  - Dynamic sizing of bus width for 64, 32, 16, or 8 bits
  - Access to 64-bit VRAM/DRAM/SDRAM/SRAM memory
- A video controller that contains dual frame timers for simultaneous image capture and display (TMS320C80 only)
- Four external interrupts (three edge-triggered and one level-triggered)

- A full-scan design (plus boundary scan), accessed via an IEEE1149.1-compliant test port that provides emulation support
- A 3.3-volt design
- TI EPIC™ 0.5/0.6-μm CMOS technology
- Efficient packaging
  - 305-pin ceramic PGA (TMS320C80)
  - 240-pin plastic QFP (TMS320C82)

1.2 Why Should I Use the TMS320C8x?

The ’C8x provides flexible support for evolving industry standards and readily adapts to the computational requirements of proprietary algorithms by combining exceptional processing speed with full programmability. This allows the ’C8x to support a multitude of video, graphics, audio, and telecommunications applications. In addition, the flexible ’C8x architecture, its fully scannable design, and its in-circuit emulation capability permit you to design a system that meets your specific needs and that can replace multiple boards and multiple processors.

- **The TMS320C8x fulfills multiply-intensive, pixel, and bit-field processing needs.**

  Image-processing applications require two main types of processing: multiply-intensive transforms and pixel/bitfield manipulations. Most DSPs are designed to perform only multiply-intensive operations, but are not designed for pixel and bit manipulations. The ’C8x, however, meets the needs of both types of processing; the hardware and instruction set of the ’C8x PPs differ greatly from those of traditional DSPs in their ability to manipulate bit fields and to process multiple pixels in parallel through the data path.

- **The TMS320C8x allows you to perform high-precision operations and floating-point computations.**

  Two-dimensional and three-dimensional graphics, audio processing, and general-purpose processing can require high-precision operations (such as 32×32 bit multiplies) and floating-point computations. The ’C8x master processor (MP) offers a special set of floating-point instructions to support graphics transforms and DSP-like floating-point operations.

- **The TMS320C8x provides the processing power to support telecommunications applications such as digital switches in telephone networks and cellular base stations.**

  The ’C8x is also ideal for applications in the telecommunications industry. Telephone networks and cellular base stations, for example, require tremendous processing power. A typical cellular base station today employs over 4000 DSPs in a single station. The ’C8x can reduce this number by an order of magnitude.

- **The TMS320C8x offers high data bandwidth and effective interprocessor communication.**

  Common to most applications is the need for high data bandwidth. A related necessity is the need for effective interprocessor communication. Through a combination of specialized features, including numerous shared 64-bit wide RAMs, a crossbar network, and an intelligent DMA controller, the ’C8x achieves high bandwidth, reducing the time that its processors spend waiting on data and ensuring that interprocessor communication does not bottleneck.

- **The TMS320C8x provides you with in-circuit emulation, allowing you to control and monitor the execution of each of the processors.**

  The ’C8x includes on-chip features that facilitate in-circuit emulation. These features allow you to control execution of the processors and to monitor each of the processor’s registers. You can control the execution of each processor independently of the other processors, or you can synchronize processor execution. The control and observation requirements for emulation take advantage of scan paths provided for testability. Communication with the on-chip emulation hardware is provided through an IEEE1149.1-compliant test access port.

- **The TMS320C8x allows you to test all registers and latches.**

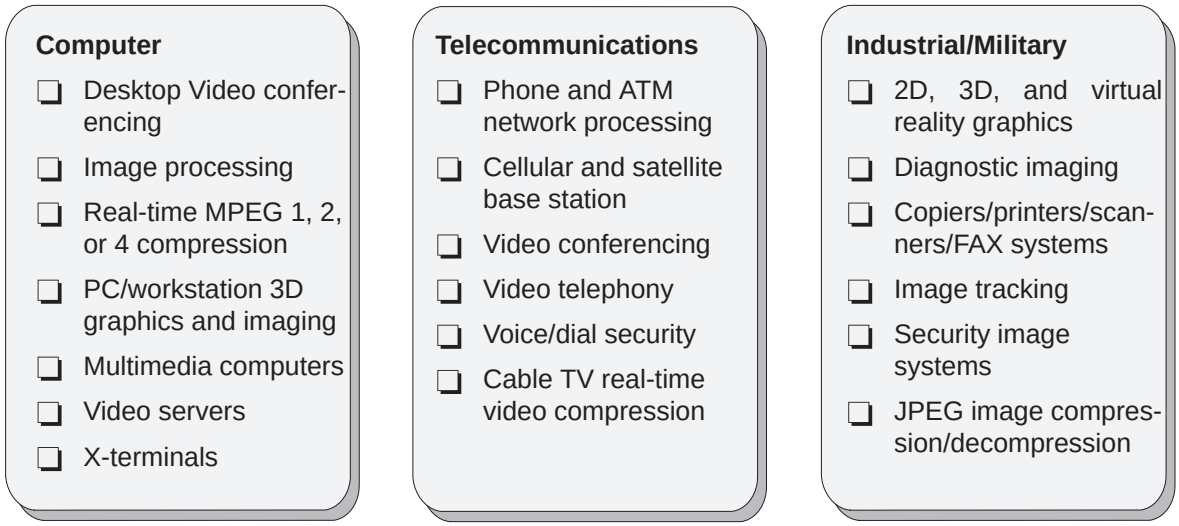
  The ’C8x is a fully scannable design that permits you to test the status of all registers and latches. The device uses the TI modular port scan design for the internal core logic. The I/O ports are on a boundary scan path that conforms to the IEEE1149.1 standard for boundary scan architecture. Both the internal scan paths and the boundary scan paths are accessed through an IEEE1149.1-compliant test access port.

1.3 Typical Applications

The ’C8x is ideal for applications in the computer, telecommunications, industrial, video, graphics, and military markets. Typical applications are shown in Figure 1–2.

In the past, if you wanted to create a system that combined video, audio, and telecommunication functions, for example, you needed to have a separate processor and board for each application of the system. The ’C8x is unlike any other DSP in that you can combine multiple applications on a single board.

Figure 1–2. Typical TMS320C8x Applications

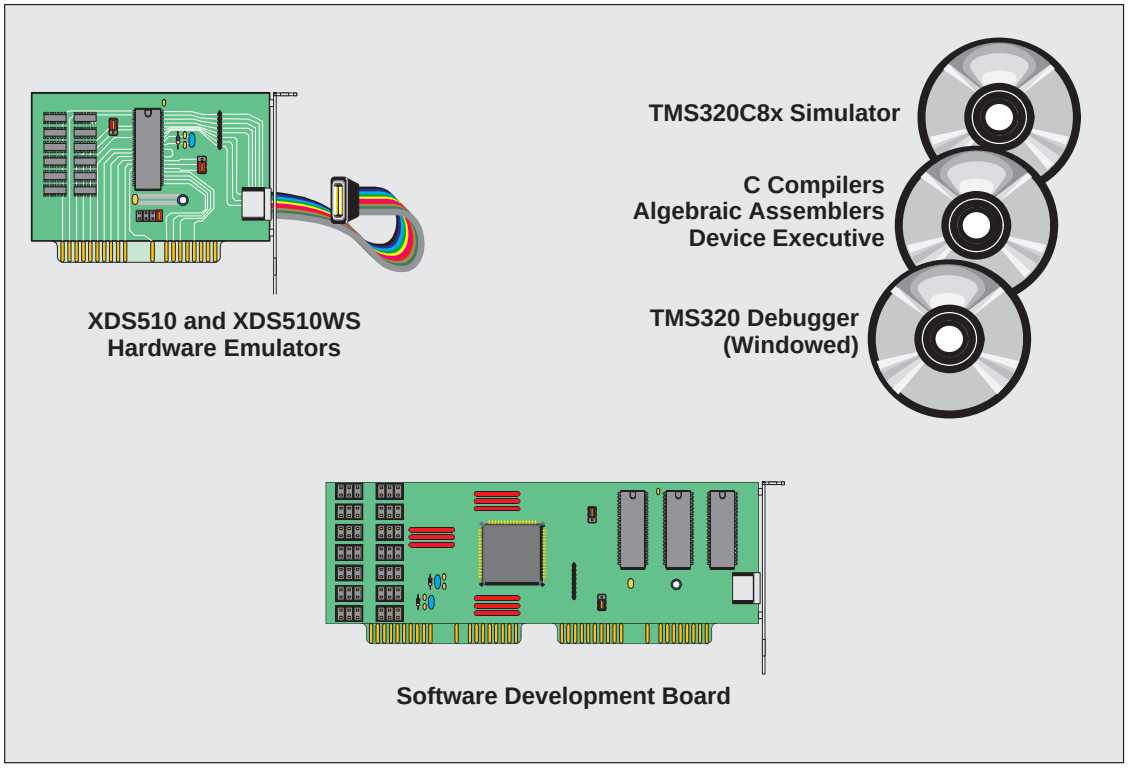


1.4 The TMS320C8x Development Environment

The ’C8x supports code development for parallel-processing applications by offering a wide range of development tools for SUNs and PCs. These tools include optimizing C compilers and assemblers for the MP and the PPs and debugging tools such as an in-circuit emulator and a software simulator.

Figure 1–3 shows the major parts of the ’C8x development environment.

Figure 1–3. The TMS320C8x Development Tools



There are eight primary ’C8x software development tools:

- Optimizing ANSI C compilers, which include several key features:
  - Separate compilers for the MP and PPs
  - Easy implementation of data and message-passing between tasks (or processors) in parallel-processing systems
  - C-source and target-specific optimizations
  - Plum-Hall validation to ANSI standard for code portability

  For more information about the C compilers, see the *TMS320C80 Code Generation Tools User’s Guide*.
- Algebraic assemblers and a linker, including:
  - Separate assemblers that support the MP and PP assembly languages, and a linker that links assembled code in common object file format (COFF) into common memory
  - Directives to map program and data code on specific processors for fast integration and debugging of parallel-processing code
  - Support for creating relocatable modules for maximum code flexibility

  For more information about the assemblers and the linker, see the *TMS320C80 Code Generation Tools User’s Guide*.
- Register allocator that:
  - Streamlines code development by efficiently managing usage of registers
  - Reduces the time designers normally spend on register management, which is generally a time-consuming task
- Code compactor that:
  - Examines serially-written code to identify instructions that can run in parallel
  - Maximizes system performance and simplifies development of efficient parallelized code
- Multitasking executive that:
  - Executes on the MP
  - Simplifies the synchronization and execution of multiple DSP tasks
  - Supports C8x-to-host processor communications and provides local control of on-chip parallel-processing tasks

- Software development board (SDB) with:
  - Full-featured PC add-in card that includes hook-ups for acquiring audio and video input and for developing, benchmarking, and debugging ’C8x code
  - Board includes:
    - A 40 MHz TMS320C80
    - A PCI bus master interface
    - A 16-bit stereo audio subsystem that supports a sampling rate of up to 48kHz
    - A 16-bit video acquisition/display subsystem
    - 2 MB of display VRAM to support 16-bit 1024x768 video applications
  - SDB device driver for Windows NT™, including a simple applications-programming interface to control data flow across the host bus
  - Full concurrent emulation support provided through a real-time debugger interface that allows for downloading, debugging, and benchmarking code on a ’C8x device
- Parallel-processing in-circuit emulator (XDS510/XDS510WS) that allows you to:
  - Debug C and assembly code simultaneously using the graphical source-level debugger
  - Debug any number of ’C8xs in a system with a single XDS510 or XDS510WS controller
  - Globally stop, start, and single-step all or any combination of ’C8x devices in a system

  For more information about the emulator and the debugger, see the *TMS320C80 C Source Debugger User’s Guide.*
- SUN-based Software simulator that provides:
  - Cycle-by-cycle simulation of the ’C8x
  - Effective tool to simulate key software kernels

For more information about the simulator, see the *TMS320C80 C Source Debugger User’s Guide.*

Chapter 2

Multiprocessing and System Architecture

This chapter describes the ’C8x architecture. The TMS320C8x architecture combines RISC processing and the processing power of multiple advanced DSPs (parallel processors), giving you the flexibility to design systems that support video, audio, graphics, imaging, and other processor-intensive applications. The TMS320C80, for example, provides you with the processing power to perform over two billion operations per second (BOPS).

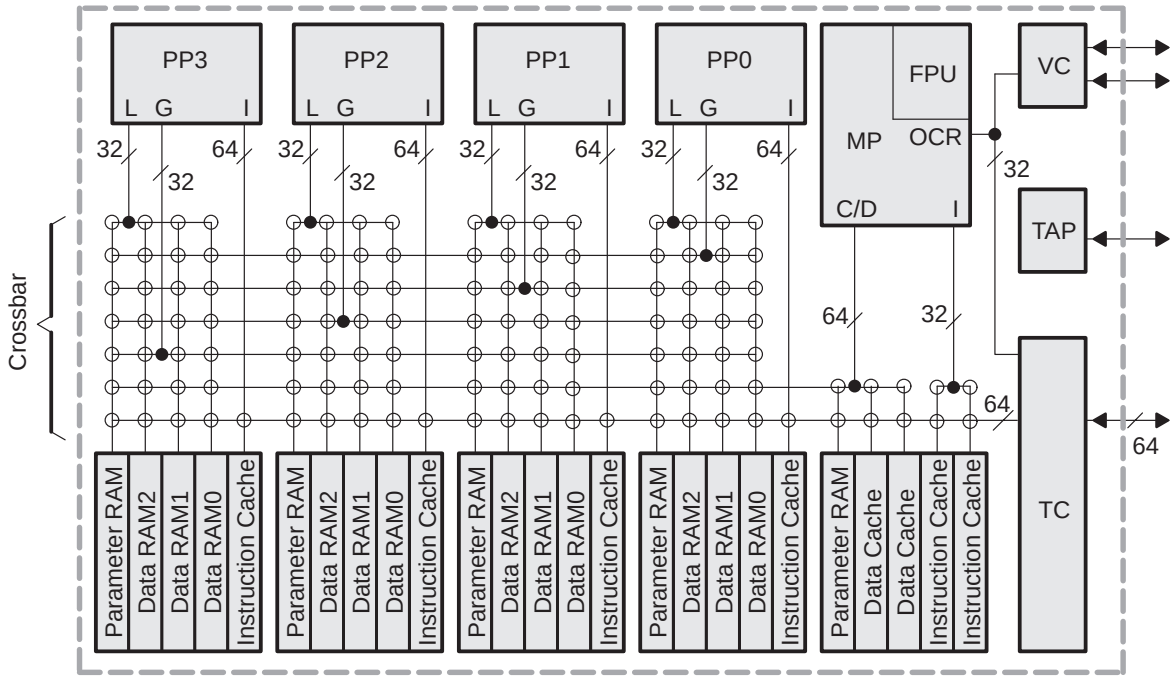
Topics

2.1 Architecture Overview

The ’C8x provides software flexibility and system adaptability with its fully programmable parallel-processing platform that integrates both advanced DSP and RISC architectures. The processors on the ’C8x can be configured for a variety of multiple-instruction, multiple-data (MIMD) operations and are connected by a crossbar network to on-chip SRAMs and to the external memory via the transfer controller. This allows you to use shared memory efficiently and to eliminate processing delays resulting from contention.

Figure 2–1 and Figure 2–2 illustrate the components of the ’C80 and the ’C82.

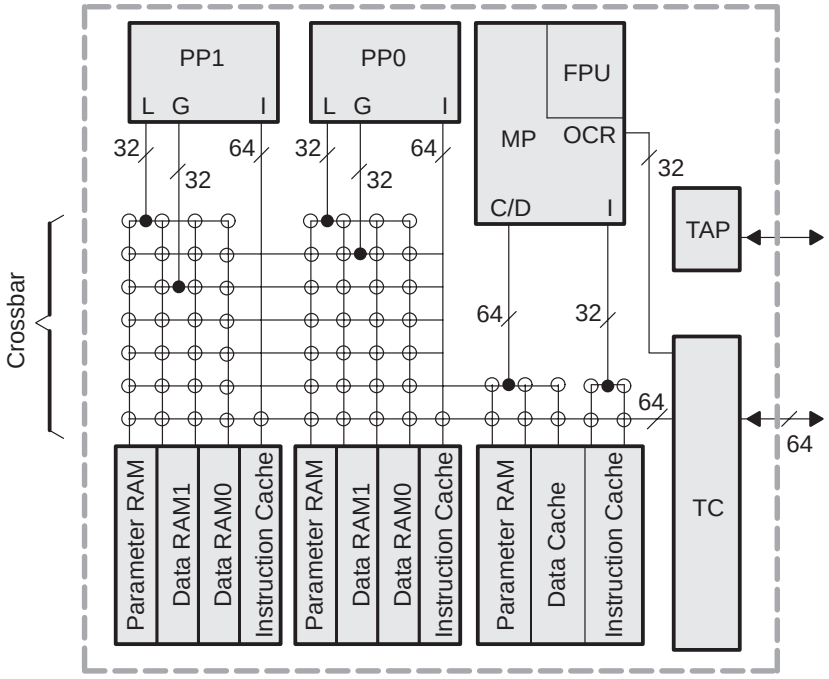
Figure 2–1. TMS320C80 Block Diagram



Legend:

L	Local port	G	Global port
I	Instruction port	FPU	Floating-point unit
OCR	On-chip register port	C/D	Cache/data port
TAP	Test access port	TC	Transfer controller
VC	Video controller	MP	Master processor
PP0–3	Parallel processors 0–3		

Figure 2–2. TMS320C82 Block Diagram



Legend:

L	Local port	G	Global port
I	Instruction port	FPU	Floating-point unit
OCR	On-chip register port	C/D	Cache/data port
TAP	Test access port	TC	Transfer controller
VC	Video controller	MP	Master processor
PP0–3	Parallel processors 0–3		

The following information describes each functional unit of the TMS320C8x:

- **Master processor (MP).** The MP is a 32-bit RISC processor with an IEEE-754 floating-point unit (FPU).
- **Parallel processors (PPs).** The PPs are advanced DSP, 32-bit integer units.
- **Test access port (TAP).** An IEEE1149.1-compliant test port provides access to internal emulation and boundary-scan paths.
- **Transfer controller (TC).** The TC services caches and transfers blocks of data between external memory and internal memory; external and internal memory accesses by the TC use a 64-bit port.
- **Local port (L).** A 32-bit local port on each PP provides access to on-chip SRAM data that is local to the PP.
- **Global port (G).** A 32-bit global port on each PP provides access to all on-chip shared SRAM data.

- **Instruction port (I).** Instruction ports on the PPs and on the MP provide access to on-chip instruction caches:
  - Each PP uses a 64-bit instruction port.
  - The MP uses a 32-bit instruction port.
- **Cache or data port (C/D).** A 64-bit cache or data port on the MP provides access to the data cache or on-chip SRAMs.
- **On-chip register port (OCR).** The MP accesses the memory-mapped TC and VC ('C80 only) registers through a 32-bit port.
- **Crossbar.** The on-chip processors use crossbar switching to access on-chip RAM. (In Figure 2–1 and Figure 2–2, the ⊕ symbol shows valid connections.) See Chapter 4, *The Crossbar*, for more information about the crossbar.
- **On-chip RAM.** The memory is divided into several smaller special-purpose RAMs.
  - **Parameter RAM.** Each processor (PPs and MP) has a parameter RAM that is partially dedicated to storing interrupt vector addresses and specifying parameters used by the TC. Each processor on the 'C80 has 2 KB of parameter RAM; each processor on the 'C82 has 4 KB. You can use the rest of this RAM for general data storage. Any processor can access the parameter RAM associated with any PP.
  - **Data RAM.** In the 'C80, each PP has three 2 KB data RAMs (6 KB total per PP) that any processor can access as shared memory. In the 'C82, each PP has two 4 KB data RAMs (8 KB total per PP). Any processor (PP or MP) can access any of the data RAMs through the crossbar.
  - **Instruction cache.** The PPs and the MP have their own instruction caches. The size of the MP's instruction cache is 4 KB in both the 'C80 and the 'C82. The size of each PP's instruction cache is 2 KB on the 'C80 and 4 KB on the 'C82. These caches are managed by an instruction-cache controller in each processor.
  - **Data cache.** In the 'C80 and in the 'C82, the size of the MP's data cache is 4 KB. The cache memory is managed by the MP's data-cache controllers. The PP's manage their on-chip data storage explicitly in software instead of through an automatic hardware cache.
- **Video controller (VC).** The 'C80's VC includes dual frame timers that provide simultaneous image capture and display. The 'C82 does not have a VC, but instead supports low-cost external DRAM-based capture/display systems.

2.2 The Master Processor (MP)

The master processor (MP) is a 32-bit RISC processor with an integral IEEE-754 floating-point unit. The MP architecture is designed for efficient execution of C code. As is the case with other RISC processors, the MP accesses memory with load and store instructions, and it performs most integer and logical operations on registers in a single cycle.

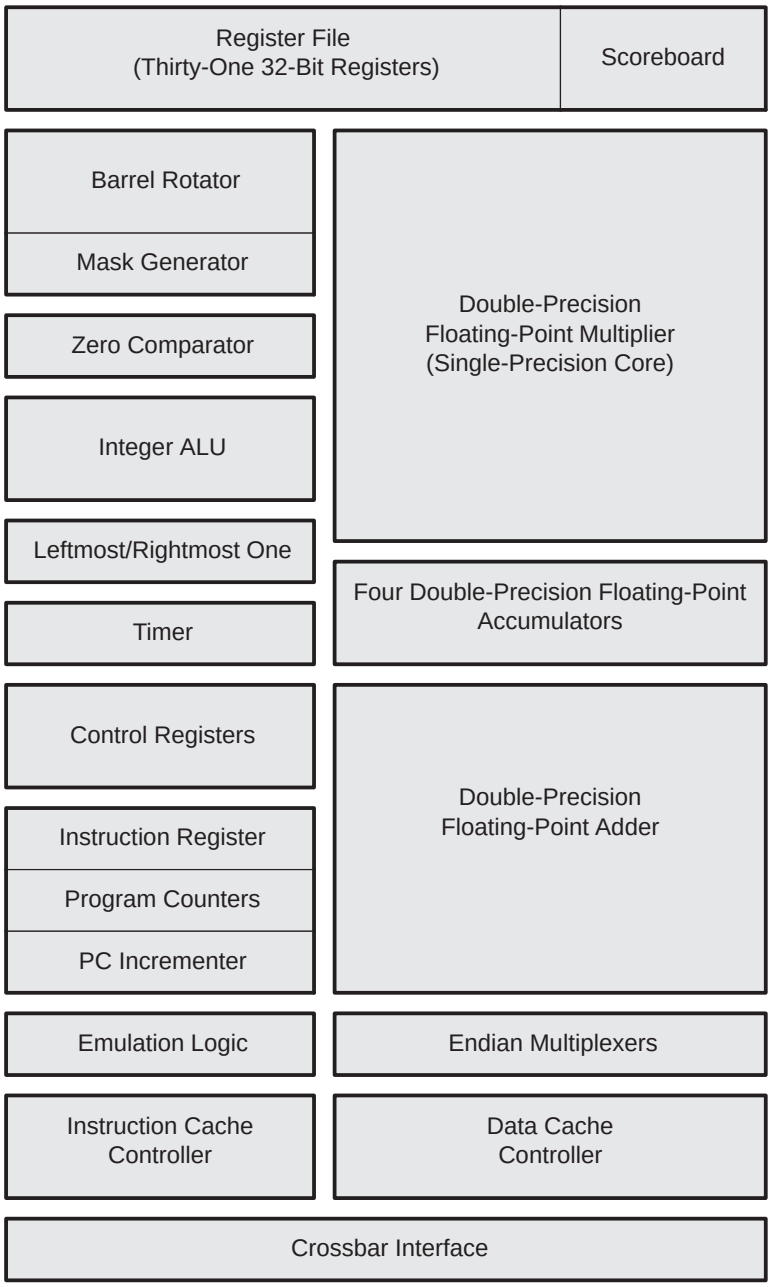
Floating-point instructions are pipelined; therefore, you can start a single-precision multiply or any floating-point add instruction on each clock cycle. Moreover, with a special set of parallel multiplication, addition, load, and store instructions, the floating-point unit is capable of up to 100 MFLOPS in performance at a 50 MHz internal clock rate.

Floating-point unit operations use the same register file as the integer and logic unit. A register scoreboard ensures that correct register-access sequences are maintained.

MP instructions and data are fetched from a pair of on-chip caches, each of which is 4 KB in size (for more information about the caches, see Section 3.2, *Accessing On-Chip Memory*). The control for these caches is an integral part of the MP design. Figure 2–3 shows the block diagram of the master processor.

For more information about the master processor, see the *TMS320C80 Master Processor User's Guide.*

Figure 2–3. Master Processor Block Diagram



2.3 The Parallel Processors (PPs)

The parallel processors (PPs) provide much of the TMS320C8x's computational power. The PPs perform digital signal processing and bit-field/multiple-pixel manipulation. These processors have advanced features that are not found in other DSPs or general-purpose processors; a single PP can perform the equivalent of ten RISC-like operations in each clock cycle.

Each PP's 64-bit wide instruction word allows multiple parallel operations to be specified in one instruction. Each instruction has fields that independently control the data unit (with its multiplier and ALU data paths) and the two address units. All instructions execute in a minimum of a single cycle.

Figure 2–4 shows the components of a PP. Each PP has a set of 44 user-visible registers. All registers can be the source or destination of ALU or memory operations. The **register set** is divided into files according to each register's function. Most of the registers support more than one access in a cycle; registers in the data unit support over eight accesses in a single cycle.

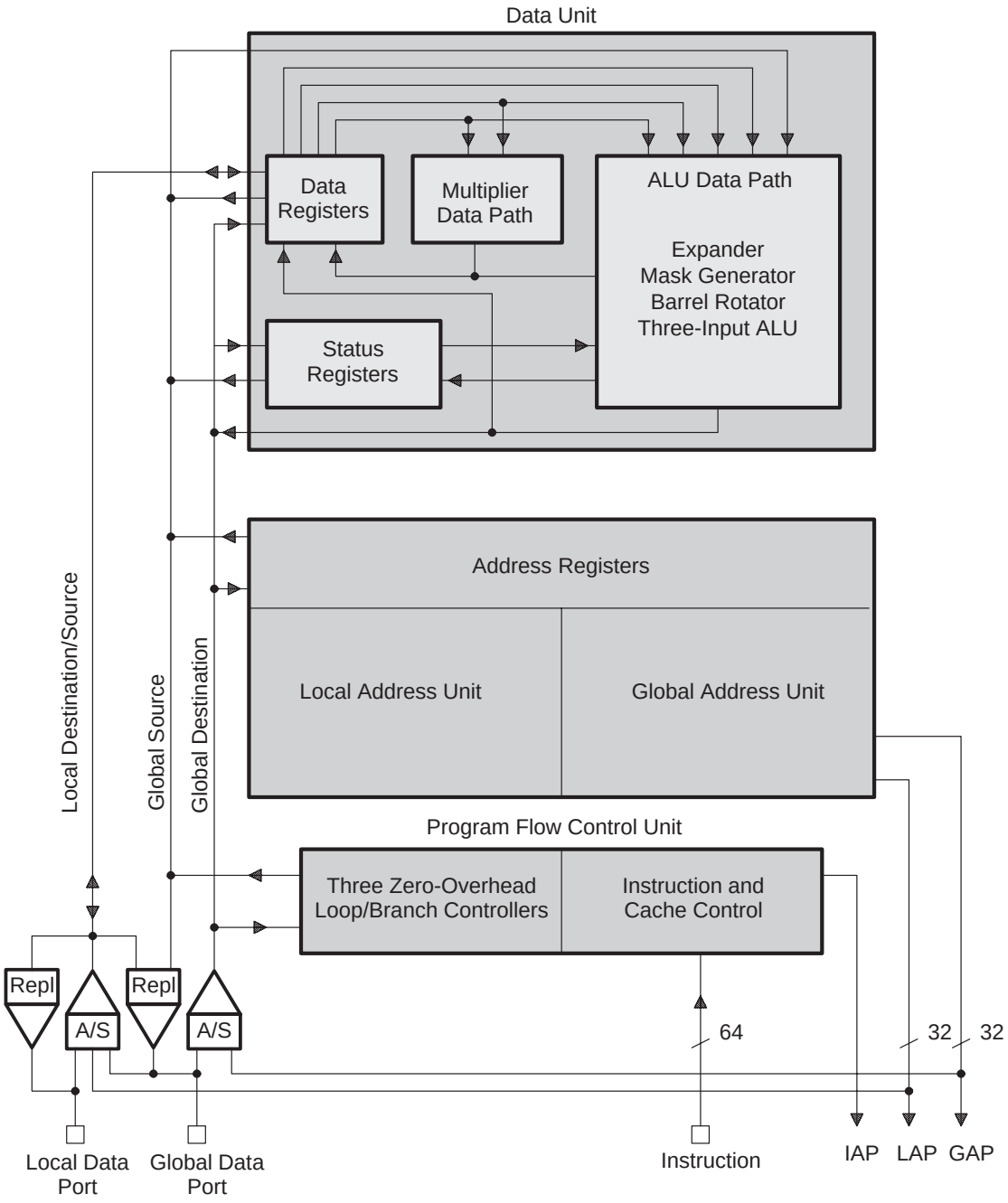
The **data unit** supports the massive processing associated with algorithms such as frequency domain transforms (for example, discrete cosine transforms), correlation, and filters. It also supports the bit-field and pixel manipulations required by image coding and computer graphics. The combination of special hardware and a flexible data path allows a single-cycle data unit operation to perform the equivalent of many general-purpose processor instructions.

The two **address units** (global and local) are nearly identical and together can perform two independent memory operations in each cycle. Each memory operation is a load or store that can be totally independent of the data unit operation. The address unit forms an address by adding an immediate index or a register index to an address register. The result of an address computation can be used to modify the address register to facilitate stepping through a memory array. During an instruction that does not need the address unit to perform a memory access, the unit is available to perform arithmetic operations on register data.

The **program flow control unit** controls the PP instruction pipeline, fetches and decodes instructions, performs any necessary handshaking with the TC, and handles interrupt response and prioritization. The major hardware elements of the program flow control unit are the instruction controller, the program counter registers, the cache controller, and the three hardware loop controllers.

For more information about the parallel processor, see the *TMS320C80 Parallel Processor User's Guide.*

Figure 2–4. Parallel Processor Block Diagram



Legend:

IAP	Instruction address port	LAP	Local address port
GAP	Global address port	Repl	Replicate hardware
A/S	Align/sign extend hardware		

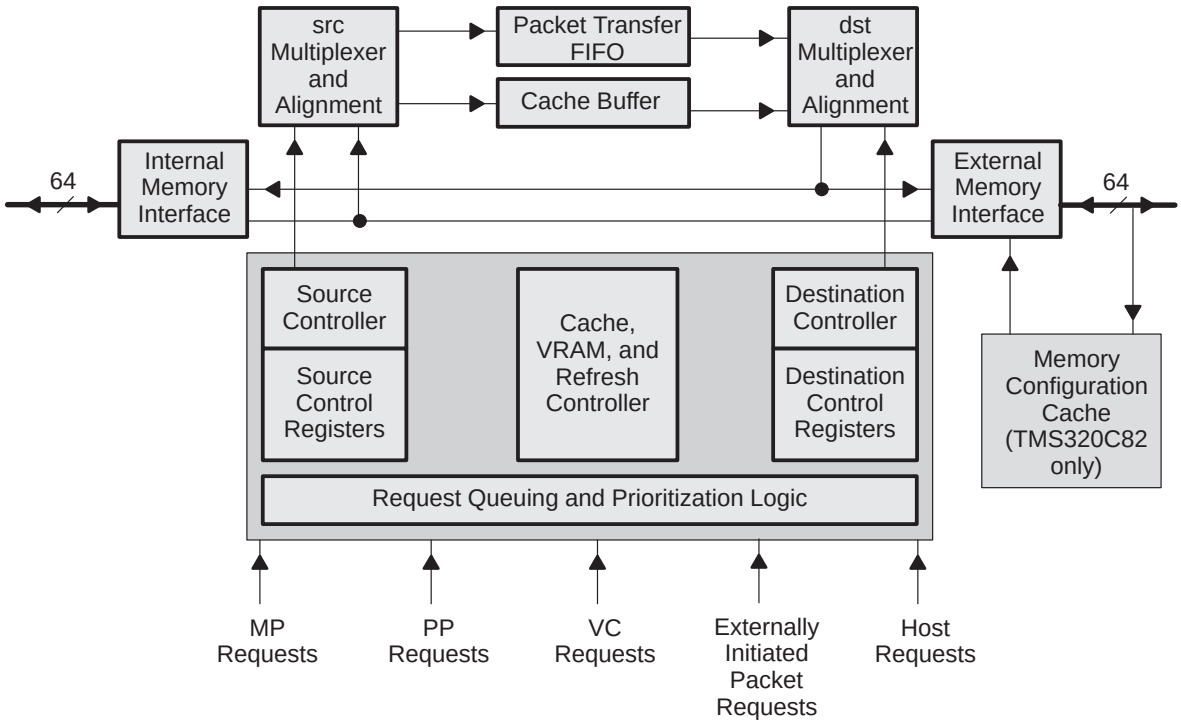
2.4 The Transfer Controller (TC)

The transfer controller (TC) is a combined direct memory access (DMA) controller and memory interface that intelligently queues, prioritizes, and services data requests and cache misses of the MP and the PPs. Accesses to external devices by the MP and the PPs take place through the TC.

The ’C8x TC, shown as a block diagram in Figure 2–5, has four main features:

- The TC processes source and destination addresses with independent controllers: the **source controller** and the **destination controller**.
- The **packet transfer FIFO** (first-in, first-out logic) supports DRAM page and burst modes and buffers between byte-misaligned accesses to access memory more efficiently.
- A separate **cache controller** can preempt program-controlled packet transfers to service cache misses. This controller uses the **cache buffer** to buffer incoming data.
- The **request queuing and prioritization logic** prioritizes active requests and starts transfers. The TC automatically suspends and later resumes lower-priority requests when a higher-priority request occurs.

Figure 2–5. TMS320C8x Transfer Controller Block Diagram



The ’C8x transfer controller has several key capabilities that make it an effective interface between memories:

- Flexible memory interfacing
- Dynamic bus sizing
- Memory configuration caching (TMS320C82 only)
- Flexible data transfers
- Multiple external packet transfers

2.4.1 Memory Interfacing

The TC’s off-chip memory interface supports SDRAMS, DRAMs, VRAMs, SRAMs, and ROMs. The support for DRAMs, including timing control and address multiplexing, is relatively new in DSPs. The combination of fast on-chip SRAM and off-chip DRAM improves performance and reduces system costs.

The TC also supports a host-interface mechanism that allows a host processor to gain access to the system. The host-interface mechanism causes all memory interface pins to be set to high impedance, allowing the host to drive the memory control signals. Additionally, the TC contains a DRAM refresh controller and interface logic to the VC (’C80 only) to perform VRAM serial register transfer operations.

2.4.2 Dynamic Bus Sizing

The TC allows you to dynamically size the bus on a page-by-page basis (row address), enabling the selection of 64-, 32-, 16-, and 8-bit transfers. Dynamic bus-sizing, coupled with external page-by-page memory device-type checking, almost eliminates the need for external interface hardware, regardless of whether or not only a single type of memory device is used.

2.4.3 Memory Configuration Caching (TMS320C82 only)

The ’C82 TC includes a memory configuration cache consisting of six 32-bit words that describe the properties of the six most-recently-used banks of memory. The cache automatically loads configuration words each time an access to a new bank is made. Each entry in the cache can be locked, set high-priority, or set low-priority. If the cache is full and an access to a new bank is made, the least-recently-used low-priority entry is replaced. If there are no low-priority entries, then the least-recently-used high-priority entry is replaced. If all six cache entries are locked, then a memory fault occurs. This method of operation allows you to minimize access times by locking the most important memo-

ries and peripherals into the cache and by assigning priorities to less-critical memories and memory-mapped peripherals. Additionally, caching the configuration information reduces the number of pins necessary in the ’C82 and in support chips.

2.4.4 Packet Transfers

Linked-list packet transfers are specifically requested by the MP, by the PPs, or by external devices to transfer data between two memory areas. In response to these requests, the TC transfers multidimensional blocks of information between a source and a destination, either of which can be on-chip or off-chip. In graphics and imaging, it is common for the TC to fetch data from an image region as a two-dimensional X/Y array and bring the data on-chip for processing as a linear array. After processing, the results can then be stored off-chip as an X/Y array. The ability of the TC to make these transformations autonomously greatly improves the efficiency of processing by the PPs and the MP.

Long-form packet transfers require at least 64 bytes of parameter RAM for the packet request data. Short-form packet transfers require only 16 bytes of parameter RAM. The ’C80 supports only long-form packet transfers. In contrast, the ’C82 TC supports long-form and short-form packet transfers.

2.4.5 Externally-Initiated Packet Transfers (XPTs)

External hardware receives or sends data through externally-initiated packet transfers (XPTs). One application of XPTs is to control DRAM-based or VRAM-based display buffers. In such an application, typically two or more XPTs initiate transfers to generate a two-dimensional display. For DRAM-based display buffers, the TC causes multiple transfers for each packet request. The TC also supports shift-register transfer to control VRAMs. The ’C80 supports seven XPTs and the ’C82 supports fifteen.

Long-form XPTs are high-priority and suspend any packet transfer other than another XPT. This form of XPT allows an external device to use the full flexibility of the TC.

On the ’C82, short form XPTs do not require in-progress packet transfers to be suspended, thus significantly reducing the worst-case latency compared to long-form XPTs. Generally, short-form XPTs should be used for timing-critical operations, such as display refresh requests.

For more information about the transfer controller, see the *TMS320C80 Transfer Controller User’s Guide.*

2.5 The Video Controller (VC) (TMS320C80 only)

The video controller (VC) is the interface between the ’C80 and image capture and display systems. The VC provides simultaneous control over two independent frame systems. The frame systems provide several features, including variable screen resolution and data capture. The independence of the two frame systems allows flexibility in design; you may, for example, configure them to control two displays, to control a display and an image-capture device, or to control two image-capture devices.

Four main sections make up the video controller:

- **Frame timers.** Two identical frame timers provide video timing control. Each frame timer has its own input frame clock (FCLK) and operates asynchronously with respect to the rest of ’C80 logic. Each frame timer can be programmed to generate timing pulses on five output signals that can be used to control an image capture or display device. These signals provide:
  - Separate or composite synchronization (sync) and blanking
  - Synchronization to internal or external signals
  - Interlaced or noninterlaced frame control
  - Limitless screen resolutions

- **Serial register transfer controller.** A serial register transfer (SRT) controller generates SRT cycle requests to the TC to transfer data into and out of frame memories. The frame timers indicate to the SRT controller when an SRT is necessary. Then, the SRT controller generates the required addresses and synchronizes and prioritizes the requests before passing them on to the TC. This controller can also initiate packet transfers on the TC to control DRAM-based display memories.

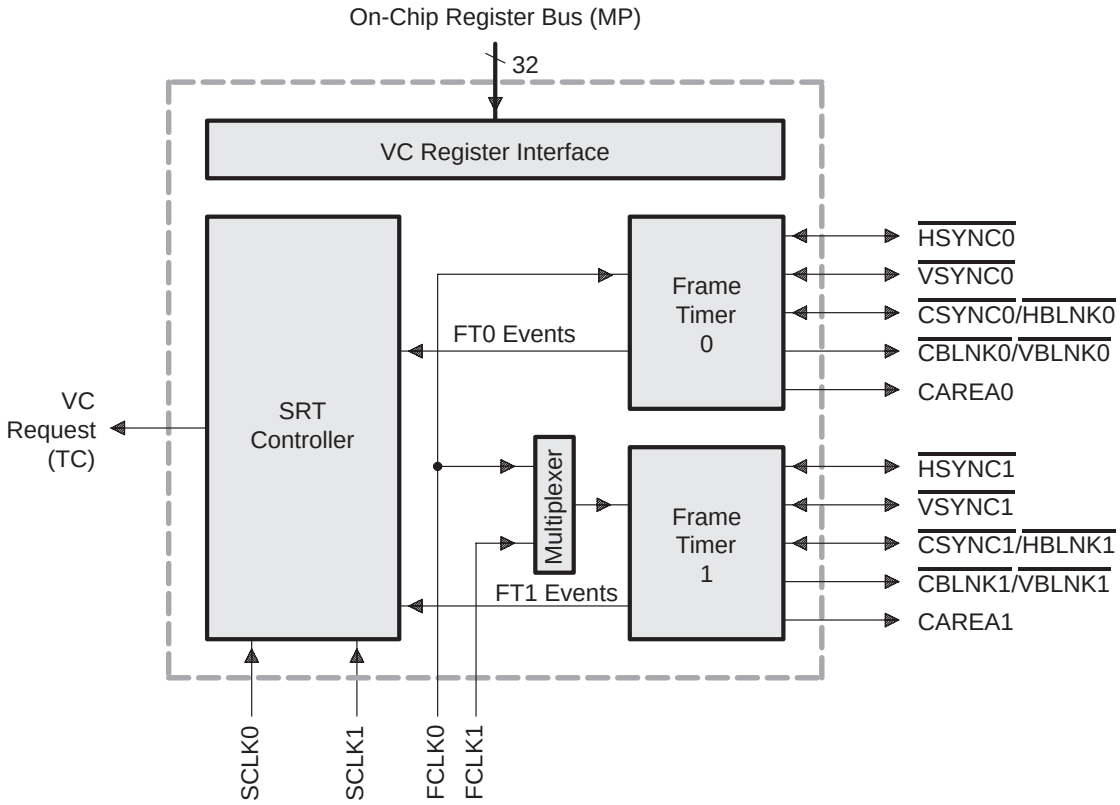
- **Register interface.** The register interface accesses the VC’s memory-mapped registers via the MP’s 32-bit register bus.

- **Multiplexer.** The multiplexer allows the synchronization of the two frame timers to a single FCLK (FCLK0).

The VC can be split into functional sections, as shown in Figure 2–6.

For more information about the video controller, see the *TMS320C80 Video Controller User’s Guide.*

Figure 2–6. Video Controller Block Diagram



2.6 The Multitasking Executive Software

The multitasking executive software runs on the master processor and provides local control of on-chip parallel-processing tasks to present a uniprocessor-like interface to the outside world. The executive has two primary components:

- The **kernel** consists of a software library of user-callable functions, providing basic program control, as well as intertask communication and synchronization. The kernel offers a foundation on which to build sophisticated system services.
- The **software interface** serves as a channel for MP tasks to issue commands to the parallel processors (PPs).

The multitasking kernel that runs on the MP is message-based; MP-resident tasks use variable-length messages to communicate with each other and to communicate with a general-purpose host processor. MP-resident tasks can also communicate with other tasks by signaling them through shared data objects called semaphores. Interrupt service routines typically use signals sent through semaphores, rather than messages sent through ports, to inform MP tasks of interrupt events.

A typical MP task is a server routine that processes a stream of requests received from a client process that runs on a host processor. The host may be a UNIX workstation, for example. An MP-resident server task might perform graphics, image processing, or video compression/decompression operations requested by a client.

Most of the code for the executive is written in C. The MP's C compiler can be configured to generate either big- or little-endian code. User-callable functions within the kernel are designed to be called either by C programs or by programs that follow C conventions regarding function calls.

The multitasking executive includes source code, as well as object code. The source code is included not only to help you better understand how the executive operates, but also to serve as a starting point for customizing the executive to meet your specific requirements.

For more information about the multitasking executive, see the *TMS320C80 Multitasking Executive User's Guide.*

Chapter 3

The TMS320C8x Memory Organization

This chapter describes the organization of the TMS320C8x's 4-GB memory space.

Topics

3.1 Overview of the TMS320C8x Memory Organization

The TMS320C8x is a byte-addressable device with a single 4-GB memory space common to its processors. Each address refers to a specific byte in the address space. Addresses less than 0x0200 0000 are reserved for on-chip memory, and addresses from 0x0200 0000 to 0xFFFF FFFF are assigned to off-chip memory. The memory map is shown in Figure 3–1 for the ’C80 and in Figure 3–2 for the ’C82.

Each of the processors on the ’C8x can access on-chip memory simultaneously. The TC can make one access to on-chip memory, the MP can make two accesses to on-chip memory, and each PP can make three accesses to on-chip memory in any given instruction cycle, for a total of 15 (’C80) or 9 (’C82) possible accesses in a single cycle. The ’C8x supports these accesses with the crossbar, a high-speed switch network between the processors, the TC, and on-chip RAMs. (For more information about the on-chip RAMs, see Section 3.2. For more information about the TC, see Section 2.4. For more information about the crossbar, see Chapter 4.)

Figure 3–1. TMS320C80 Memory Map

Starting Address (hex)	Ending Address (hex)	Bank size (bytes)	Memory or Device
0000 0000	0000 07FF	2K	PP0 Data RAM 0
0000 0800	0000 0FFF	2K	PP0 Data RAM 1
0000 1000	0000 17FF	2K	PP1 Data RAM 0
0000 1800	0000 1FFF	2K	PP1 Data RAM 1
0000 2000	0000 27FF	2K	PP2 Data RAM 0
0000 2800	0000 2FFF	2K	PP2 Data RAM 1
0000 3000	0000 37FF	2K	PP3 Data RAM 0
0000 3800	0000 3FFF	2K	PP3 Data RAM 1
0000 4000	0000 7FFF	16K	Reserved
0000 8000	0000 87FF	2K	PP0 Data RAM 2
0000 8800	0000 8FFF	2K	Reserved
0000 9000	0000 97FF	2K	PP1 Data RAM 2
0000 9800	0000 9FFF	2K	Reserved
0000 A000	0000 A7FF	2K	PP2 Data RAM 2
0000 A800	0000 AFFF	2K	Reserved
0000 B000	0000 B7FF	2K	PP3 Data RAM 2
0000 B800	00FF FFFF	16M†	Reserved
0100 0000	0100 07FF	2K	PP0 Parameter RAM
0100 0800	0100 0FFF	2K	Reserved
0100 1000	0100 17FF	2K	PP1 Parameter RAM
0100 1800	0100 1FFF	2K	Reserved
0100 2000	0100 27FF	2K	PP2 Parameter RAM
0100 2800	0100 2FFF	2K	Reserved
0100 3000	0100 37FF	2K	PP3 Parameter RAM
0100 3800	0100 FFFF	50K	Reserved

† Block sizes have been rounded to the nearest unit size.

Figure 3–1. TMS320C80 Memory Map (Continued)

Starting Address (hex)	Ending Address (hex)	Bank size (bytes)	Memory or Device
0101 0000	0101 07FF	2K	MP Parameter RAM
0101 0800	0180 17FF	8M†	Reserved
0180 1800	0180 1FFF	2K	PP0 Instruction Cache
0180 2000	0180 37FF	6K	Reserved
0180 3800	0180 3FFF	2K	PP1 Instruction Cache
0180 4000	0180 57FF	6K	Reserved
0180 5800	0180 5FFF	2K	PP2 Instruction Cache
0180 6000	0180 77FF	6K	Reserved
0180 7800	0180 7FFF	2K	PP3 Instruction Cache
0180 8000	0180 FFFF	32K	Reserved
0181 0000	0181 07FF	2K	MP Data Cache 0
0181 0800	0181 0FFF	2K	MP Data Cache 1
0181 1000	0181 7FFF	28K	Reserved
0181 8000	0181 87FF	2K	MP Instruction Cache 0
0181 8800	0181 8FFF	2K	MP Instruction Cache 1
0181 9000	0181 FFFF	28K	Reserved
0182 0000	0182 01FF	512	Memory-Mapped TC Registers
0182 0200	0182 03FF	512	Memory-Mapped VC Registers
0182 0400	01FF FFFF	8M†	Reserved
0200 0000	FFFF FFFF	4G†	External Memory

† Block sizes have been rounded to the nearest unit size.

Figure 3–2. TMS320C82 Memory Map

Starting Address (hex)	Ending Address (hex)	Bank size (bytes)	Memory or Device
0000 0000	0000 0FFF	4K	PP0 Data RAM 0
0000 1000	0000 1FFF	4K	PP1 Data RAM 0
0000 2000	0000 7FFF	24K	Reserved
0000 8000	0000 8FFF	4K	PP0 Data RAM 1
0000 9000	0000 9FFF	4K	PP1 Data RAM 1
0000 A000	00FF FFFF	16M†	Reserved
0100 0000	0100 0FFF	4K	PP0 Parameter RAM
0100 1000	0100 1FFF	4K	PP1 Parameter RAM
0100 2000	0100 FFFF	56K	Reserved
0101 0000	0101 0FFF	4K	MP Parameter RAM
0101 1000	0180 0FFF	8M†	Reserved
0180 1000	0180 1FFF	4K	PP0 Instruction Cache
0180 2000	0180 2FFF	4K	Reserved
0180 3000	0180 3FFF	4K	PP1 Instruction Cache
0180 4000	0180 FFFF	48K	Reserved
0181 0000	0181 0FFF	4K	MP Data Cache
0181 1000	0181 7FFF	28K	Reserved

† Block sizes have been rounded to the nearest unit size.

Figure 3–2. TMS320C82 Memory Map (Continued)

Starting Address (hex)	Ending Address (hex)	Bank size (bytes)	Memory or Device
0181 8000	0181 8FFF	4K	MP Instruction Cache
0181 9000	0181 FFFF	28K	Reserved
0182 0000	0182 01FF	512	Memory-Mapped TC Registers
0182 0200	01FF FFFF	8M†	Reserved
0200 0000	FFFF FFFF	4G†	External Memory

† Block sizes have been rounded to the nearest unit size.

3.2 On-Chip Memory Organization

Each of the on-chip static RAMs is dedicated to one of four uses:

- **Data caches** are read/write memory for frequently used data (MP only). In the ’C80 and in the ’C82, the size of the MP’s data cache is 4 KB. The cache memory is managed by the MP’s data-cache controllers. The PP’s manage their on-chip data storage explicitly in software instead of through an automatic hardware cache.
- **Instruction caches** store frequently-used code blocks to accelerate program execution. Each ’C80 PP has one 2 KB instruction cache and each ’C82 PP has one 4 KB instruction cache. In the ’C80 and in the ’C82, the size of the MP’s instruction cache is 4 KB.
- **Data RAMs** are standard read/write memory with no caching or special features. Data RAMs are the main areas in which the PPs store the data they are processing. Any transfer of data to or from the data RAMs is done explicitly by the processors, either by a direct load (read) from or store (write) to memory, or by a request to the TC to transfer the data through packet transfers. In the ’C80, the PPs each have three 2 KB data RAMs. In the ’C82, they each have two 4 KB data RAMs.
- **Parameter RAMs** are similar to data RAMs, except that part of each RAM is dedicated to specific hardware functions. Certain locations in the lower addresses are reserved for interrupt vectors, TC use, etc. For a PP, the default location for the stack is also in its parameter RAM. Typically, the parameter RAMs also contain frequently used data and structures. The size of the MP’s parameter RAM is 2 KB in the ’C80 and 4 KB in the ’C82. The size of each PP’s parameter RAM is 2 KB in the ’C80 and 4 KB in the ’C82.

The memory map allocations for the data RAMs, parameter RAMs, and cache RAMs are shown in Figure 3–1 (’C80) and Figure 3–2 (’C82).

The data RAMs and the PP parameter RAMs are collectively referred to as **shared RAMs** because they are accessible to all processors through the crossbar.

3.2.1 MP Data Caches

The MP's data cache is mapped into directly into the address space, as shown in Figure 3–1 and Figure 3–2. In the 'C80 and in the 'C82, the MP's data cache occupies 4096 bytes and is four-way set associative:

- Each set has four 256-byte blocks.
- Each block contains four 64-byte subblocks.
- Each subblock holds sixteen 32-bit data words.
- Each block's address begins on a 256-byte boundary.
- Each subblock's address begins on a 64-byte boundary inside the block's address space.
- The cache uses the write-back protocol to maintain cache coherency when data is modified.
- The cache does not use bus snooping or bus watching.
- The cache uses the least recently used (LRU) replacement algorithm.

3.2.2 MP Instruction Cache

The MP's instruction cache is mapped directly into the address space (see the memory map in Figure 3–1 and Figure 3–2). The 'C80 MP has two 2 KB instruction caches and the 'C82 MP has one 4 KB instruction cache. In both cases, the MP's instruction cache occupies 4096 bytes and is four-way set associative:

- Each set has four 256-byte blocks.
- Each block contains four 64-byte subblocks.
- Each subblock holds sixteen 32-bit instructions.
- Each block's address begins on a 256-byte boundary.
- Each subblock's address begins on a 64-byte boundary inside the block's address space.
- The cache uses the LRU replacement algorithm.

3.2.3 PP Instruction Caches

Each PP instruction cache is mapped directly into the address space (see Figure 3–1 and Figure 3–2). In the ’C80 each PP instruction cache occupies 2048 bytes; in the ’C82 each instruction cache occupies 4096 bytes.

- Each cache is divided into four blocks.
- Each block contains four subblocks.
- Each subblock holds sixteen 64-bit instructions.
- Each block’s address begins on a 512-byte (’C80) or a 1024-byte (’C82) boundary.
- Each subblock’s address begins on a 128-byte (’C80) or a 256-byte (’C82) boundary inside the block’s address space.
- The cache uses the LRU replacement algorithm.

3.2.4 Data and Parameter RAM Organization

Each of the on-chip RAMs in the ’C80 consists of 256 (’C80) or 512 (’C82) rows of 64-bit wide single-ported static RAM. Each RAM can service either one read (load) or one write (store) request per machine cycle. The processors read data from and write data to memory as bytes, halfwords (two bytes), words (four bytes), or doublewords (eight bytes). Accesses are aligned to the data size; in a given address, halfword accesses ignore the least significant bit (LSB), word accesses ignore the two LSBs, and doubleword accesses ignore the three LSBs. Additionally, bytes, halfwords, and words are addressed differently in big-endian and little-endian modes (see Section 3.5 for more information about endian modes).

3.3 Accessing Memories and Registers

The processors of the TMS320C8x read data and produce results through accesses to memories and registers. Although most accesses occur on-chip, accesses to external memory and external devices are necessary for the chip to perform its tasks.

3.3.1 On-Chip RAM Accesses

Memory load or store operations at addresses less than 0x0200 0000 access on-chip locations. During each clock cycle each processor can access a RAM, assuming that contention does not occur. The accesses can switch between processors on a cycle-by-cycle basis. Switching between processors is facilitated by the crossbar (See Chapter 4 for more information about the crossbar.) An access to on-chip RAM banks occurs in one cycle if there is no contention.

MP loads and stores can access the following:

- any PP's data RAMs
- the MP's parameter RAM (stores are possible only in the MP supervisor mode)
- external memory via the MP's data cache

3.3.2 Other On-Chip Accesses

Control registers are used to monitor and control the status of the MP, PPs, and other on-chip mechanisms. The following on-chip registers can be accessed only by the MP:

- **Transfer controller registers.** The control registers for the transfer controller are memory-mapped into the 'C8x's address space so that MP loads or stores can access individual registers. Section 1.3, *Transfer Controller Registers*, in the *TMS320C80 Transfer Controller User's Guide* describes the TC's control register.
- **Video controller registers.** The control registers for the video controller ('C80 only) are memory-mapped into the 'C80's address space so that MP loads or stores can access individual registers. Chapter 3, *Frame Timer Registers*, in the *TMS320C80 Video Controller User's Guide* describes the VC's control registers.

- **Master processor control registers.** The MP control registers are memory-mapped into the ’C8x’s address space. Table 2–1, *MP Control Register Numbers*, in the *TMS320C80 Master Processor User’s Guide* describe how the MP uses branch, read, swap, or write control-register instructions (brcr, rdcr, swcr, or wrcr respectively) to access the control registers. For a complete description of the control registers, see Chapter 3, *Master Processor Control Registers*, in the *TMS320C80 Master Processor User’s Guide*.

3.3.3 Accessing External Memory

The MP’s off-chip data and instructions reside in external memory beginning at address 0x0200 0000. Information in this area is cached by the MP’s instruction and data caches and by the PPs’ instruction caches. External memory is typically accessed directly by testing and diagnostic software, by packet transfers, and by accesses to peripherals. Figure 3–1 shows the addresses of the cache RAMs.

A PP or the MP can access single locations in off-chip memory directly using the direct external memory access (DEA) mechanism (described in Section 3.4, *Direct External Memory Access*). In this way, the MP can bypass the data cache mechanism and access peripheral device registers without cache interference.

3.4 Direct External Memory Access (DEA)

The MP and PPs operate primarily on data that has been brought on-chip by the TC to service an MP cache request or by a request for a packet transfer. However, there are a few cases (such as accessing a single word in external memory) when it is desirable to specify an external memory address directly in an MP or PP instruction. The MP or a PP can access off-chip memory directly using the direct external access (DEA) mechanism. DEAs are used primarily to access memory-mapped peripherals in the ’C8x’s external memory.

3.4.1 Master Processor DEAs

DEAs allow values to be manipulated without loading or flushing the data cache. This is appropriate for memory-mapped peripherals and for handshake locations used in communications with other processors.

DEA versions of the MP’s load and store instructions (dld and dst) bypass the data cache for addresses greater than or equal to 0x0200 0000. These instructions are implemented by sending a modified cache request to the TC. The modified request reads or writes directly to or from external memory.

A DEA requires more time than an on-chip load or store because it carries nearly all of the overhead of a cache subblock miss for each access. However, a DEA can improve performance—the alternative would be to flush an entire cache subblock.

Note:

You must maintain cache coherency if you use DEAs that alter external memory contents that are also resident in the cache. See Section 11.10, *Clean Data Cache Using the dcachec Instruction*, in the *TMS320C80 Master Processor User’s Guide*.

3.4.2 Parallel Processor DEAs

The PPs' DEA instructions are specified exactly like crossbar memory accesses to the shared RAMs, except that the address does not point to an on-chip shared RAM. When a PP attempts an access to an address that is not in the shared RAMs, a DEA request is sent to the TC.

If the address is greater than 0x01FF FFFF, the request is to off-chip memory. This off-chip memory access is serviced by the TC when the requesting PP's turn in the cache/DEA round-robin is reached (see Section 4.4 for more information about crossbar access decisions).

Functionally, a DEA is exactly like a memory access to the shared RAMs. However, unlike accesses to shared RAMs, which require a single cycle (with no contention), a DEA access requires a minimum of 11 cycles for a load and 8 cycles for a store. Also, if the TC's service queue already contains other cache service or DEA requests, a DEA access may require additional cycles.

For a load, the PP's pipeline is stalled until the data read from off-chip memory has been written to the destination register. For a store, the PP's pipeline is stalled only until the TC verifies that the memory access can complete. Thus, the minimum pipeline stall for a DEA store is less than that for a DEA load.

Note:

For accessing more than a few bytes of data, such as rows or blocks of image data, packet transfers are more efficient than DEAs. Packet transfers are programmed data transfers that are submitted to the TC and serviced in parallel with normal PP operation. Chapter 12, *Packet Transfers*, in the *TMS320C80 Parallel Processor User's Guide* discusses packet transfers in further detail.

If the address is in on-chip memory but is not accessible by the PP (for example, any on-chip memory address that is not in the shared RAMs), a fault occurs and the PP remains stalled until it is reset by either a software reset command issued by the MP or by a hardware reset.

3.5 Endian Mode

The ’C8x can operate on data in either little-endian or big-endian mode. During power-up reset, a hardware signal automatically sets the endian mode and also sets the MP’s CONFIG[E] bit to the correct value. The endian mode can not be changed while the processor is operating.

The processor divides memory into four types of data elements: 8-bit bytes, 16-bit halfwords, 32-bit words, and 64-bit doublewords. Big-endian mode orders data elements from left to right, with the most significant element on the right, as shown in Figure 3–3. Little-endian mode orders data elements from right to left, with the most significant element on the left, as shown in Figure 3–4.

Single-precision floating-point elements are 32-bit words, and are, therefore, treated as 32-bit data elements. Note that endian mode affects only the ordering of elements; bits are ordered from right to left, regardless of endian mode.

Notes:

Use caution when you use different sizes for data loads and stores in the same 64-bit location, since the endian mode affects how the data is loaded or stored. For example, pay attention to endian mode when using two store word instructions (st) with one load double instruction (ld.d).

Figure 3–3. Data Order in Big-Endian Mode

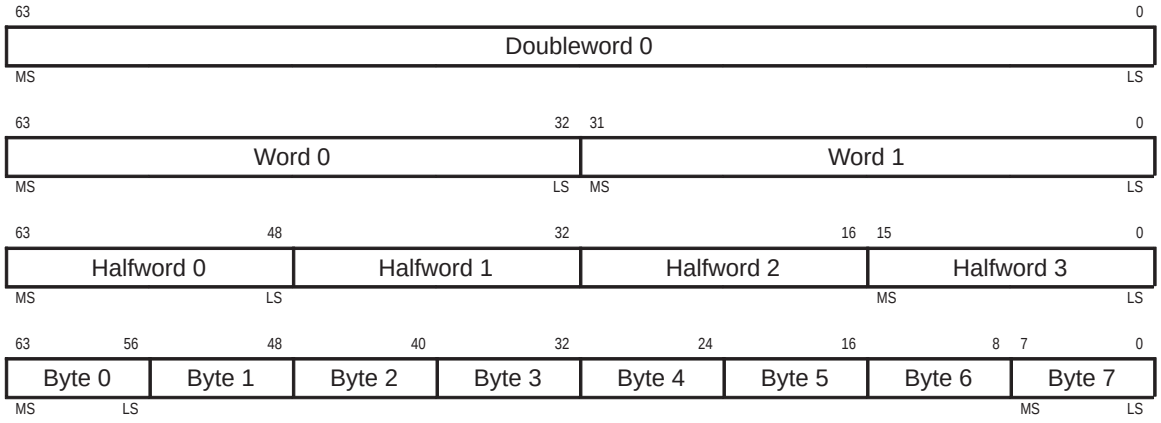
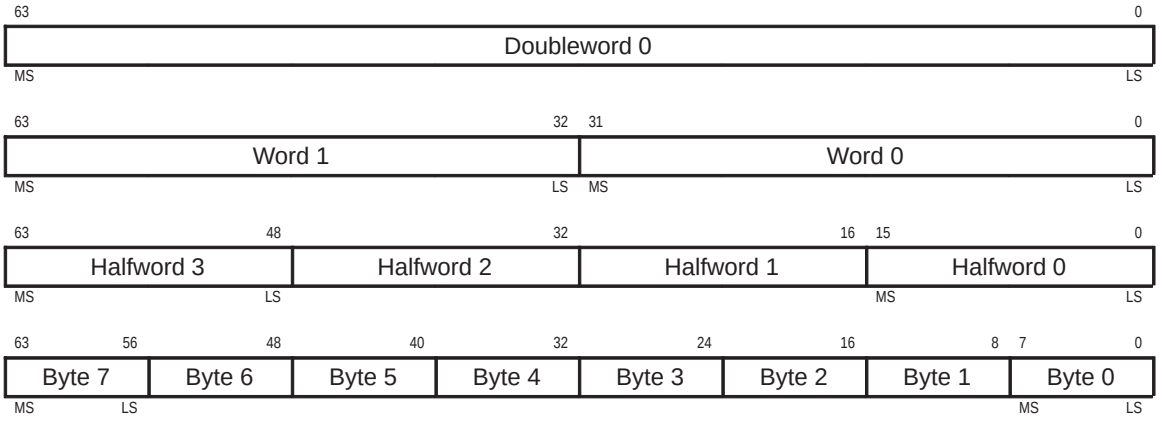


Figure 3–4. Data Order in Little-Endian Mode



Chapter 4

The Crossbar

The crossbar is the communications network of the TMS320C8x. This chapter describes the crossbar and message transfer and prioritization.

Topics

4.1 Overview of the Crossbar

The crossbar is the primary means for enabling the high degree of parallelism achieved by the device. The crossbar facilitates accesses to on-chip memory by the master processor, parallel processors, and transfer controller; it controls which module connects to a RAM for a given access. The crossbar is autoconfiguring: for a processor to access a given RAM, it only needs to specify the address of the word in memory that it needs to access; the crossbar automatically connects the processor to that RAM.

The crossbar connects processors to memories on a cycle-by-cycle basis. It uses two pipeline stages, with each stage completing in one cycle; an access to a RAM can be initiated during every cycle, but a given access actually occurs over two clock periods. Each RAM can be accessed by a different processor during each clock cycle.

In the ’C82, the crossbar allows three instructions fetches and six parallel data accesses per clock cycle to provide high overall data transfer rates:

- 1.0 GB/s transferring instructions
- 1.6 GB/s transferring data

In the ’C80, the crossbar allows five instructions fetches and ten parallel data accesses per clock cycle to provide overall transfer rates even higher higher than those of the ’C82:

- 1.8 GB/s transferring instructions
- 2.4 GB/s transferring data

Crossbar-shared memory reduces restrictions on where data must reside by maintaining nearly 1000 data and address lines between processors and memories. Furthermore, it is cost-effective to use because all of the memory connections are integrated on one chip. The crossbar’s flexibility improves efficiency in execution speed and ease of programming.

In addition to managing memory accesses, the crossbar is also used to send commands between processors. Interprocessor commands can perform several functions:

- Send message interrupts
- Halt the MP or PPs or unhalt the PPs
- Flush caches
- Reset the PPs, MP, TC, or VC (’C80 only)
- Send new tasks to the PPs.

4.2 Crossbar Connections

The crossbar does not connect all processors to every RAM. Instead, it has specialized connections that correspond to each individual processor.

The master processor (MP) has two ports to the crossbar:

- The instruction port accesses the MP's instruction cache.
- The data port accesses the MP's data cache, parameter RAMs, and any of the shared data RAMs

Each parallel processor (PP) has three ports to the crossbar:

- The global data port can access any of the shared RAMs.
- The local data port can access only those RAMs that are local to a given PP.
- The instruction port accesses the processor's instruction cache.

Table 4–1 shows accessibility of RAMs through the crossbar.

Crossbar switch connections are determined by the MSBs of each address during each cycle. If more than one access is requested of the same RAM in a single cycle, round-robin prioritization hardware determines which processor is allowed access and which processor is stalled until the next cycle. (see Section 4.4, *Crossbar Access Decisions*, for more information about round-robin prioritization.)

Table 4–1. Accessibility of RAMs From Processor Ports for the TMS320C8x

	Crossbar Port								
RAM	**TC Port**	**MP Data Port**	**MP Instr. Port**	**PPx Instr. Port**	**PPx Global Data Port**	**PP0 Local Data Port**	**PP1 Local Data Port**	**PP2 Local Data Port‡**	**PP3 Local Data Port‡**
MP data caches	yes	yes	no	no	no	no	no	no	no
MP instr. caches	yes	no	yes	no	no	no	no	no	no
MP parameter RAM	yes	yes	no	no	no	no	no	no	no
PPx instr. cache	yes	no	no	yes	no	no	no	no	no
PPn instr. cache†	yes	no	no	no	no	no	no	no	no
PP0 data RAMs	yes	yes	no	no	yes	yes	no	no	no
PP1 data RAMs	yes	yes	no	no	yes	no	yes	no	no
PP2 data RAMs‡	yes	yes	no	no	yes	no	no	yes	no
PP3 data RAMs‡	yes	yes	no	no	yes	no	no	no	yes
PP0 parameter RAM	yes	yes	no	no	yes	yes	no	no	no
PP1 parameter RAM	yes	yes	no	no	yes	no	yes	no	no
PP2 parameter RAM‡	yes	yes	no	no	yes	no	no	yes	no
PP3 parameter RAM‡	yes	yes	no	no	yes	no	no	no	yes

†PPn instruction cache is an instruction cache for a PP other than PPx.
‡PP2 and PP3 and their associated RAMs and ports do not exist in the ’C82.

4.3 Crossbar Operation

The crossbar operates in a pipelined fashion, with each pipeline stage completing in one machine cycle. An access can start on every cycle, but each access occurs over two cycles:

1) At the beginning of the first pipeline stage, the requesting processor or the TC send to the crossbar the address bits that determine which RAM to access. Several processors and/or the TC may request access to the same RAM during a single cycle, but the crossbar logic associated with each RAM determines which unit is granted access to the RAM during the cycle (see Section 4.4, *Crossbar Access Decisions*).

2) At the beginning of the second pipeline stage, the processor/TC that was granted access to a RAM sends the LSBs of the address over the crossbar to the RAM. The LSBs determine which 8-byte doubleword to access in the RAM. The entire access is performed during this stage.

For a read from RAM, the data is returned to the processor/TC before the end of the second stage of the crossbar pipeline. For a write to RAM, the processor/TC sends the bytes of the doubleword to be modified at the same time it sends the LSBs of the address; the write completes at the end of the second stage.

Figure 4–1 and Figure 4–2 summarize the operation of writes and reads through the crossbar.

Figure 4–1. Pipeline Operation for RAM Write Accesses Over the Crossbar

Stage 1	1. One or more processors and/or the TC sends the MSBs of the address over the crossbar to select the actual RAM to access. 2. The crossbar sends the granted signal to one of the requesting processors or to the TC.
Stage 2	3. The processor/TC that is granted access sends the address LSBs and the data to be written over the crossbar. 4. The system completes the write to RAM.

Figure 4–2. Pipeline Operation for RAM Read Accesses Over the Crossbar

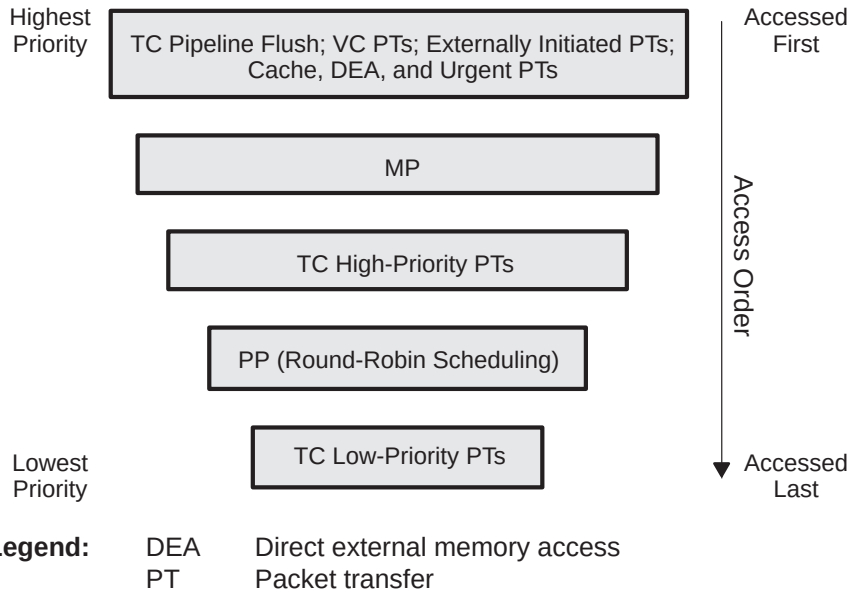
Stage 1	1. One or more processors and/or the TC sends the MSBs of the address over the crossbar to select the actual RAM to access. 2. The crossbar sends the granted signal to one of the requesting processors or to the TC.
Stage 2	3. The processor/TC that is granted access sends the address LSBs over the crossbar. 4. The system returns the contents at the address to the requesting processor or to the TC.

4.4 Crossbar Access Decisions

The crossbar allows only one access to each memory on a given clock cycle. If more than one processor or transfer controller attempt to access the same memory in a single cycle, the crossbar arbitrates and allows only one access to succeed. The processor or controller that is denied access on that cycle stalls until it is granted access. The crossbar uses a combination of two mechanisms to assign RAM accesses to processors:

- **Priority scheduling** takes precedence over round-robin scheduling. Through this mechanism, the processor or controller with the highest priority is granted access to the RAM first. Figure 4–3 shows the priority level of each processor.
- **Round-robin scheduling** is used if two or more processors try with equal priorities attempt to access the same memory and no other processor or controller with higher priority attempts to access that memory. Round-robin scheduling grants access to one processor at a time in a cyclical fashion, preventing one processor from monopolizing accesses to the RAM.

Figure 4–3. Crossbar Access Priority



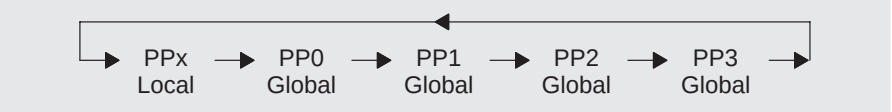
For example, if the MP, PP0, and the TC try to access PP0's parameter RAM, then the MP succeeds on the first cycle, the PP succeeds next, and the TC succeeds last. However, this example assumes that no further requests for the RAM are made and that the TC request was a low-priority packet transfer.

When more than one PP requests access to a single RAM and the PP accesses have the highest priority, a round-robin scheduling scheme allows accesses to cycle equally among PPs. Round-robin accesses are based on a “token” that is passed between PP crossbar ports in a fixed cyclical order, as shown in Figure 4–4. The token is given to the port that is granted access to the RAM. At the next access, the port holding the token has the lowest priority, and the next port in the ring has highest priority.

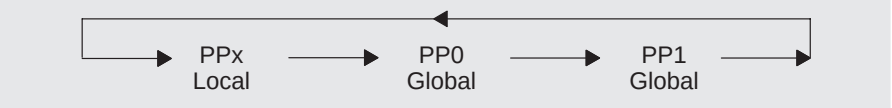
You can disable round-robin scheduling by using the MP’s CONFIG register. If round-robin scheduling is turned off, the token is always held by PP3’s global port in the ’C80 or by PP1’s global port in the ’C82. In this condition, the local port of the adjacent PP always has highest priority, PP0’s global port has the next highest priority, and PP3’s global port (’C80) or PP1’s global port (’C82) has the lowest priority. With round-robin scheduling turned off, one PP can monopolize the access to a RAM to the exclusion of other PPs. However, by turning round-robin scheduling off, you have the advantage of knowing the access order; with round-robin scheduling enabled, access order may vary significantly between runs of a program.

Figure 4–4. Round-Robin Token Passing Order

(a) Token Passing Order in the TMS320C80



(b) Token Passing Order in the TMS320C82



Note: PPx local is the local port of the PP to which the RAM is local.

4.5 Resolving Crossbar Contention

Contention occurs when two or more simultaneous access attempts are made to the same RAM. It can occur between different processors, between the global and local ports of the same PP, and between the TC and one or more of the processors. Contention lowers performance by stalling the pipeline of one or more processors and/or the transfer controller until each processor's corresponding memory access can complete.

The TMS320C8x supports automatic resolution of contention. When contention occurs, instruction execution pauses for each contending processor until access is granted. The crossbar grants access to contending requests individually, according to the prioritization described in Section 4.4.

Intelligent allocation of data can minimize contention. For example, in certain applications in which the parallel processors are tightly coupled, the order in which data accesses are performed becomes a significant factor. For a software protocol example for passing data between processors via shared memory, see Section 5.2, *Circular Command Queue*, in the *TMS320C80 Multitasking Executive User's Guide*.

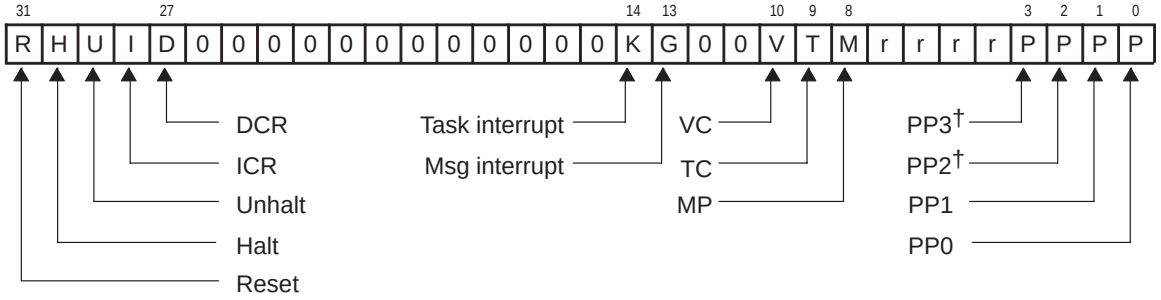
The 'C8x provides several hardware features that assist software management of shared RAMs. One of these features allows each processor to send interrupts to other processors to signal when it is finished with a RAM (see subsection 9.1.3, *The intflg Register*, in the *TMS320C80 Parallel Processor User's Guide*). These interrupts can be managed either by interrupt service routines or by polling of interrupt flags.

4.6 Interprocessor Communication

Interprocessor communication is used to signaling other processors and to allow the MP basic control over other processors and controllers. Interprocessor communication takes the form of 32-bit command words that can be sent by the MP or the PPs. The MP sends command words with the cmnd instruction. The PP sends command words with the cmnd=register subinstruction.

A command word is sent through the crossbar to one or more processors. A command word contains two sets of bits: one set of bits determines which processors receive the command and the other set determine the command itself. Figure 4–5 illustrates the format of the 32-bit command word. Bits 0–3 and 8–10 select the processors to receive the command. Bits 13, 14, and 27–31 determine the commands sent. When appropriate, multiple commands can be sent in a single instruction.

Figure 4–5. Command Word for Interprocessor Communication (’C8x)



† Bits 2 and 3 are reserved in the TMS320C82, since that unit does not have PP2 or PP3.

Legend:

r	reserved
DCR	data-cache reset DTAG registers and DLRU register
ICR	instruction-cache reset ITAG registers and ILRU register

Notes:

- Instruction or data cache resets, message or task interrupts, halts or resets require one additional pipeline stage to take effect.
- Setting both the halt and unhalt commands is an example of multiple commands that should not be sent in the same command word.

Not all processors support all commands. Unsupported commands, if used, are ignored by the selected processors. For example, the MP does not support an unhalt of the MP. The commands supported by the processors are as follows:

- The MP can issue a reset, a halt, an instruction-cache reset (ICR), a data-cache reset (DCR), or a message (msg) to itself.
- The MP can issue a reset to the VC or TC.
- The MP can issue a reset, halt, unhalt, ICR, msg interrupt, or task interrupt to any PP.
- A PP can issue a halt, ICR, or msg interrupt to itself.
- A PP can issue a msg interrupt to the MP or to another PP.

Notes:

- Interrupt bits should be enabled so that the target processors (the MP or PPs) can service a command request (see Figure 3–2, *MP Interrupt Registers—IE and INTPEN*, and Table 9–1, *Maskable Interrupt Priorities and Vector Addresses*, in the *TMS320C80 Master Processor User's Guide*, for the MP).
- An instruction cache reset (ICR) resets all instruction tag registers and the ILRU register for the selected processor(s). A data cache reset (DCR) resets all data tag registers and the DLRU register for the master processor only. An ICR or a DCR also resets cache-resident flags to **not present** (for the MP data cache, dirty flags are reset to 0 to indicate **not modified**). This does **not** perform any write-back of modified data subblocks to external memory.

4.6.1 Processor/Controller Select Bits

The following are individual descriptions of the processor select bits associated with the command word.

- **Resetting the VC (V bit, ’C80 only)**

  The MP uses the V bit in conjunction with the R bit (described on page SL:4-13) to reset the VC.

- **Resetting the TC (T bit)**

  The MP uses the T bit in conjunction with the R bit (described on page SL:4-13) to reset the TC.

- **Detecting a message on the MP (M bit)**

  The MP designator bit, when written in conjunction with a 1 in the message interrupt (G) bit, indicates to the MP that it should set the message interrupt flag corresponding to the PP or the MP that wrote the bit.

  The MP can send any command to itself (except the task and unhalt commands). The PPs can send message interrupt commands to the MP or any PP.

- **Detecting a message on the PPs (P bits)**

  When issued by a PP, the P bits indicate which PPs are to receive a message and which will have their message interrupt flags set. When a PP issues a halt or instruction-cache reset to itself, it must set its own designator bit, otherwise the command will not be executed.

  When issued by the MP, these bits indicate which PPs should respond to the operation in the rest of the command word. Additionally, the MP can send a data-cache reset to itself.

4.6.2 Command Select Bits

The following are individual descriptions of the command select bits associated with the command word.

❑ **Resetting a processor or controller (R bit)**

The MP can issue a software reset to any PP, TC, VC ('C80 only), or itself. The V, T, M, and P bits (described on page SL:4-12) determine which processors/controllers respond.

- A TC reset clears all of the TC's status bits and stops all transmissions.
- A VC reset clears all operations and resets VC on-chip register values ('C80 only).
- The MP reset is discussed in Section 9.5, *Reset*, in the *TMS320C80 Master Processor User's Guide*.
- A PP reset halts the target PPs.

❑ **Halting a PP (H bit)**

The MP or any PP can send a halt by setting the H bit. The M and P bits (described on page SL:4-12) determine which processors respond.

- A PP can issue a halt command to itself, or the MP can issue a halt command to any PP. A PP halt stalls that PP's pipeline until a future operation causes that PP to resume operation. During a PP halt, interrupts remain pending but are not processed until the PP is unhalted. When the PP receives an unhalt command, the PP resumes execution as though nothing had happened.
- Only the MP can send a halt to the MP; the MP will halt only if it originally powered up in a halted state. The MP halt provides an external host with the ability to perform a soft reset operation (external interrupt x3).

❑ **Unhalting a PP (U bit)**

An unhalt command clears the PP's halt latch. Only the MP can issue an unhalt command to the PPs. The P bits (described on page SL:4-12) determine which PPs will respond. Unhalt commands include the following:

- Unhalt after reset—The MP issues an unhalt command to start a PP following a hardware or software reset. After a reset-induced halt and subsequent unhalt, the PP begins code execution at the address given by the PP's task interrupt vector.
- Unhalt after halt—When an unhalt command follows a halt command, the PP resumes code execution as though nothing had happened.

The PP can issue a halt to itself but not to other PPs or the MP. However, an MP unhalt command has precedence over the PP halt command if they are simultaneous. Unhalt has precedence if the MP submits a command word with both the halt bit and the unhalt bit set. The MP software reset command will always cause a PP halt, even if the halt and unhalt bits are both set.

- **Resetting the instruction cache (I bit)**

  A PP can issue an instruction cache reset (ICR) to itself, or the MP can issue an ICR command to itself or to any PP. An ICR clears all of the tag address and present bits in the ITAG registers. It also resets the ILRU stack register. This command can be used with a software reset to regain control of the MP or a PP if it has corrupted its instruction cache.

- **Resetting the data cache (D bit)**

  The MP can only issue a data cache reset (DCR) command to itself. A DCR clears all of the tag address, present, and dirty bits in the DTAG registers. A DCR also resets the DLRU stack register. This command can be used with a software reset to regain control of the MP if it has corrupted its data cache.

Note:

The DCR command does not perform the write-back protocol of data cache to external memory. If cache coherency is mandatory, use the dcache instructions shown in Section 11.10, *Clean Data Cache Using the dcache Instruction,* in the *TMS320C80 Master Processor User's Guide*.

- **Setting the task interrupt flags (K bit)**

  The MP uses the K bit to signal the designated PPs to set their task interrupt flags. If the task interrupt flag is set, the PP takes the task interrupt. Task interrupts are typically used to make the PP switch tasks under MP control. The P bits (described on page SL:4-12) determine which processors are affected.

- **Setting the message interrupt flags (G bit)**

  The MP and the PPs can set the G bit in any of the PPs or the MP. This bit is used in conjunction with the M and/or P bits to indicate which processors should set their message interrupt flags.

  Message interrupts provide a method of on-chip interprocessor communication. For example, if the PP is performing a task for the MP, the PP can send a message interrupt to the MP when it has completed the task.

Appendix A

Glossary

A

address unit: Hardware on the PP that computes a 32-bit address during each cycle. Each PP has two address units: a global address unit and a local address unit.

ALU: *Arithmetic logic unit*. Hardware that provides the logic for arithmetic and Boolean operations.

assembler: A software utility that creates a machine-language program from a source file. There are two assemblers associated with the TMS320C8x: a mnemonic-based RISC-type assembler for the MP and an algebraic assembler for the PP.

B

big endian: An addressing protocol in which bytes are numbered from left to right within a word. More significant bytes in a word have lower numbered addresses. Endian ordering is specific to hardware and is determined at reset. See also *little endian.*

C

C compiler: A program that translates C source statements into assembly language source statements or object code.

cache: A fast memory into which frequently used data or instructions from slower memory are copied for fast access. Fast access is facilitated by the cache's high speed and its on-chip proximity to the CPU.

contention: A situation where two or more simultaneous access attempts for the same RAM are made. Contention is resolved automatically in hardware by arbitration, though a delay can occur.

control register: A mechanism for controlling and monitoring the operation of a device.

crossbar: A generally configurable, high-speed bus switching network for a multiprocessor system, permitting any of several processors to connect to any of several memory modules.

D

data cache: The MP's two SRAM banks that hold cached data needed by the MP. Data RAMs for the PPs are not cached.

data RAM: On-chip RAM that is available for the general-purpose storage of data by the MP or PPs on the TMS320C8x.

data unit: The PP's data manipulation hardware unit that includes the ALU, the multiplier, the mf expander, and the barrel rotator.

DEA: *Direct external access.* A method of accessing off-chip (external) memory without having to issue a packet transfer request to the TC.

debugger: A window-oriented software interface that helps you to debug TMS320C8x programs running on an TMS320C8x emulator or simulator.

direct external access: See *DEA.*

dirty flag: A storage bit associated with each subblock of MP data-cache memory that indicates whether the subblock contains modified data that needs to be written back to main memory.

double-precision floating-point: A floating-point number with 64 bits plus an additional hidden bit.

doubleword: A 64-bit value.

DRAM: *Dynamic random access memory.* Memory typically used for external memory; it must be refreshed periodically to maintain valid data.

E

executive: See *multitasking executive.*

external address: See *off-chip address.*

F

floating-point unit: See *FPU.*

FPU: *floating-point unit.* The MP's IEEE-754-standard hardware that consists of a full double-precision floating-point add unit and a double-precision floating-point multiply unit with a single precision core.

frame timers: In the VC, timers that provide video timing control. The frame timers indicate to the serial-register-transfer (SRT) controller when an SRT is necessary ('C80 only).

H

halfword: A 16-bit value.

I

instruction cache: An on-chip SRAM that contains current instructions being executed by one of the TMS320C8x processors. Cache misses are handled by the transfer controller.

internal address: See *on-chip address.*

interprocessor command: A message sent via the crossbar to the other on-chip processors.

J

JPEG standard: *Joint Photographic Experts Group standard*. A standard used for compressed still-picture data.

K

kernel: A set of low-level software primitives within the TMS320C8x multitasking executive that implement multitasking, the passing of messages and signals, and the monitoring of multiple events.

L

linker: A software tool that combines object files to form an object module that can be allocated into system memory and executed by the device.

little endian: An addressing protocol in which bytes are numbered from right to left within a word. More significant bytes in a word have higher numbered addresses. Endian ordering is specific to hardware and is determined at reset. See also *big endian*

LSB: *Least significant bit.* The bit having the smallest effect on the value of a binary numeral, usually the rightmost bit. The TMS320C8x numbers the bits in a word from 0 to 31, where bit 0 is the LSB.

M

master processor: See *MP.*

memory map: A map of target system memory space that is partitioned into functional blocks.

MIMD: *Multiple instruction stream, multiple data stream.* A parallel processing structure composed of multiple independent processors.

MP: *Master processor*. A general-purpose RISC processor that coordinates the activity of the other processors on the TMS320C8x. The MP includes an IEEE-754 floating-point hardware unit.

MPEG standard: *Moving Picture Experts Group standard.* A proposed standard for compressed video data.

MSB: *Most significant bit*. The bit having the greatest effect on the value of a binary numeral. It is the leftmost bit. The TMS320C8x numbers the bits in a word from 0 to 31, where bit 31 is the MSB.

multitasking executive: The portion of a multitasking software system that is responsible for executing application tasks, providing communications among tasks, and managing shared resources. See *executive*

O

off-chip address: An address external to the TMS320C8x chip. Addresses from 0x0200 0000 to 0xFFFF FFFF are off-chip addresses. See also *on-chip address*

on-chip address: An address internal to the TMS320C8x chip. Addresses from 0x0000 0000 to 0x1FFF FFFF are on-chip addresses. See also *off-chip address*

P

packet: A collection of patches of data.

packet transfer: See *PT.*

packet transfer request: An I/O request submitted to the TC that is issued when a block of data is to be moved via packet transfer. Packet transfer requests can be submitted by the MP, the PPs, the VC, or an external device.

parallel processor: See *PP.*

parameter RAM: A general-purpose RAM that is associated with a specific processor, part of which is dedicated to packet transfer information and interrupt vectors.

patch: A group of lines of equal length whose starting addresses are an equal distance apart.

PP: *Parallel processor*. The TMS320C8x's advanced digital signal processor that is used for video compression/decompression (P $\times$ 64 or MPEG), still-image compression/decompression (JPEG), 2-D and 3-D graphic functions such as line draw, trapezoid fill, antialiasing, and a variety of high-speed integer operations on image data. An TMS320C8x single-chip multiprocessor device may contain from one to eight PPs, depending on the device version.

program flow control unit: A unit that manages the opcode fetches from the PP's instruction cache.

PT: *Packet transfer*. A transfer of data blocks between two areas of memory. The TMS320C8x supports packet transfers of one, two, or three dimensions. The TC performs packet transfers that are requested by processors or external devices.

R

refresh: A method of restoring the charge capacitance to a memory device (such as a DRAM or VRAM) or of restoring memory contents.

RISC: *Reduced instruction set computer.* A computer whose instruction set and related decode mechanism are much simpler than those of microprogrammed complex instruction set computers. The result is a higher instruction throughput and a faster real-time interrupt response from a smaller, cost-effective chip.

S

semaphore: A type of shared data object that contains an integer count that is used to synchronize tasks to external events and to each other and to coordinate the sharing of resources among tasks.

shared RAMs: Memory that can be shared by TMS320C8x's processors. This consists of the PP data RAMs and the PP parameter RAMs.

single-precision floating-point: 32-bit floating-point number.

SRAM banks: *Static random access memory banks*. These include parameter and data RAM and instruction and data caches.

T

TC: *Transfer controller*. The TMS320C8x's on-chip DMA controller that services caches and transfers one-, two-, and three-dimensional data blocks between memories.

TMS320C8x: A single-chip multiprocessor device that accelerates applications such as video compression and decompression, image processing, and graphics manipulation. The device contains a master processor and from one to eight parallel processors, depending on the device version. For example, the TMS320C80 device contains four PPs.

transfer controller: See *TC.*

V

VC: *Video controller*. The portion of the TMS320C80 responsible for controlling the video interface.

video controller: See *VC.*

W

word: A sequence of 32 adjacent bits that constitutes a register or memory value. The PP supports 32-bit words. The MP also supports doublewords of 64 bits for loads and stores.

X

XPT: *Externally initiated packet transfer.* A packet transfer initiated by an external device through the ’C8x’s $\overline{\text{XPT}}$ inputs.

Index

A

B

C

D

E

F

G

H

I

K

L

M

O

P

R

S

T

U

V

X