TMS320 DSP Development Support Reference Guide

Literature Number: SPRU011F
May 1998

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.

Preface

Read This First

About This Manual

The *TMS320 DSP Development Support Reference Guide* details the extensive development support available from TI for the TMS320 family of digital signal processors. As a reference manual, it provides helpful and essential information to assist you in selecting the proper TI tools for design and development of TMS320 applications.

Hundreds of third-party suppliers provide development support tools and application hardware/software that supplement the TI DSP products and tools.

How to Use This Manual

The chapters and appendices that make up this book provide detailed reference information on the TMS320 family of DSPs. The information is arranged in the following manner.

Chapter 1, *Introduction*, outlines the advantages and applications of DSP technology. This chapter also gives an overview of TI DSPs and lists the development support products for the TMS320 family.

Chapter 2, *Selection Guide*, presents an overview of the TI DSP offerings and a listing of the TMS320 digital signal processors now available. Also included in this chapter are tables detailing part information, typical TI DSP applications, and performance benchmarks for the TMS320 DSPs.

Chapters 3 through 10 describe the TMS320 family of digital signal processors according to generation. Included in each chapter are the key features of the devices.

Chapter 11, *Mixed Signal Products*, describes products that provide interfaces for DSP solutions: analog-to-digital converters, digital-to-analog converters, and analog interface circuits.

Chapter 12, *Customizable DSPs (cDSP)*, provides the attributes, benefits, and key features of these joint customer- and TI-designed DSPs that enable custom solutions for specific applications.

Chapter 13, *Code Generation Tools*, provides an overview of software development. A discussion of software development products includes information on the TMS320C2x/C2xx/C5x, the TMS320C54x, TMS320C3x/C4x, and TMS320C8x C compilers and the TMS320 macro assembler/linker.

Chapter 14, *System Integration and Debugging Tools*, gives the reader an overview of the integration and debugging process. This chapter also discusses system integration and debugging products such as the TMS320 debugger's interface (C/assembly source debugger) and the TMS320 software simulators and emulators. Chapter 15 also covers system integration and evaluation tools, TMS320C2x/C3x/C5x DSP starters kits (DSKs), TMS320 XDS upgrade packages, and the parallel processing development system (PPDS) for the TMS320C40.

Chapter 15, *TMS320 Technical Support*, provides an overview of the technical literature and technical assistance. The chapter's technical literature overview covers application reports, data sheets, the TMS320 newsletter (*Details on Signal Processing*), product bulletins, technical articles, user's guides, and textbooks. The overview of the technical assistance covers the TMS320 Hotline, FAX capabilities, and the TMS320 Bulletin Board Service (BBS).

Chapter 16, *TMS320 Third-Party Support* explains how to get listings of third-party algorithims currently available and information on how to license third-party software. This chapter also how to obtain lists of the third-party companies and consultants who support the TMS320 DSP family.

Chapter 17, *TMS320 Seminars and Workshops*, covers seminars and 3-day (or longer) workshops offered by the TI Technical Training Organization (TTO). The chapter discusses design assistance services offered by the TI worldwide Customer Design Centers and lists their offices and addresses.

Chapter 18, *TMS320 University Program,* presents an overview of TMS320 code-generation, system-integration, and debugging tools available to universities. Additionally, it lists textbooks on DSP theories and applications using the TMS320 devices and discusses how to establish a DSP lab/research environment.

Appendix A covers *Factory Repair and Exchange Instructions*, while Appendix B presents the *Program License Agreements*. Appendix C discusses the *ROM Codes*, and Appendix D covers *TMS320 PROM Programming*.

Notational Conventions

This document uses the following conventions.

- Program listings, program examples, and interactive displays are shown in a `special typeface` similar to a typewriter's. Examples use a **`bold version`** of the special typeface for emphasis; interactive displays use a **`bold version`** of the special typeface to distinguish commands that you enter from items that the system displays (such as prompts, command output, error messages, etc.).

  Here is a sample program listing:

  ```
  0011  0005  0001         .field    1, 2
  0012  0005  0003         .field    3, 4
  0013  0005  0006         .field    6, 3
  0014  0006               .even
  ```

  Here is an example of a system prompt and a command that you might enter:

  ```
  C:  csr –a /user/ti/simuboard/utilities
  ```

Related Documentation From Texas Instruments

Texas Instruments provides extensive documentation to support the TMS320 family devices and development tools. The *TMS320 DSP Product Overview* (SPRZ094) and *TMS320 Digital Signal Processing Solutions Selection Guide* (SSDV004) are prime sources of information. See Chapter 16 of this book, *TMS320 Technical Support*, for complete lists of related materials.

Trademarks

AT and XT are trademarks of International Business Machines Corporation.

AURIS is a trademark of CSELT.

cDSP is a trademark of Texas Instruments Incorporated.

CGS is a trademark of Alta Group of Cadence Design Systems.

Code Composer is a trademark of Texas Instruments Incorporated.

DAQ-200 is a trademark of Sonitech International, Incorporated.

DAS is a trademark of Tektronix.

DEC is a trademark of Digital Equipment Corporation.

Dolby is a trademark of Dolby Laboratories Licensing Corporation.

DSP On-Line Lab is a trademark of Texas Instruments Incorporated.

Elf is a trademark of Atlanta Signal Processors, Incorporated.

ELOQUENS is a trademark of CSELT.

EPIC is a trademark of Texas Instruments Incorporated.

FLEXUS is a trademark of CSELT.

Helios is a trademark of Perihelion Software, Limited.

Hotline On-Line is a trademark of Texas Instruments Incorporated.

IBM is a trademark of International Business Machines Corporation.

Intel, i286, i386, and i486 are trademarks of Intel Corporation.

MAC II, Macintosh, and MPW are trademarks of Apple Computer Corporation.

Motorola-S is a trademark of Motorola, Incorporated.

MS-Windows is a registered trademark of Microsoft Corporation.

Microsoft is a registered trademark of Microsoft Corporation.

MS-DOS is a registered trademark of Microsoft Corporation.

NEC is a trademark of NEC Corporation.

OpenWindows is a trademark of Sun Microsystems, Inc.

OS/2 is a trademark of International Business Machines Corporation.

PAL® is a registered trademark of Advanced Micro Devices, Incorporated.

PC-AT, PC-DOS, and PC-XT are trademarks of International Business Machines Corporation.

PCMCIA is a trademark of Personal Computer Memory Card International Association.

Pentium is a trademark of Intel Corporation.

QuiteQuiet is a trademark of Lucent Technologies Incorporated.

SCbus is a trademark of Dialogic Corporation.

Solaris is a trademark of Sun Microsystems, Inc.

Sony is a trademark of Sony Corporation.

SPARC is a trademark of SPARC International, Inc., but licensed exclusively to Sun Microsystems, Inc.

SPARCstation is trademark of SPARC International, Inc., but licensed exclusively to Sun Microsystems, Inc.

Sun, Sun-3, and Sun4 are trademarks of Sun Microsystems, Inc.

SunOS is a trademark of Sun Microsystems, Inc.

SunView is a trademark of Sun Microsystems, Inc.

Tektronix is a trademark of Tektronix, Inc.

TI is a trademark of Texas Instruments Incorporated.

TImeline Technology is a trademark of Texas Instruments Incorporated.

Unison is a trademark of Multiprocessor Toolsmiths, Inc.

UNIX is a registered trademark of Unix System Laboratories, Inc. in the United States and other countries, licensed exclusively through X/Open Company Limited.

VAX and VMS are trademarks of Digital Equipment Corporation.

VME and VMEbus are registered trademarks of Motorola, Inc.

VPro and VProPRL are trademarks of Voice Processing Corp.

Virtuoso is a trademark of Eonic Systems, Inc.

Visual Basic and Visual C/C++ are trademarks of Microsoft Corporation.

Windows and WindowsNT are registered trademarks of Microsoft Corporation.

XDS, XDS510, XDS 510WS, XDS511, and XDS522 are trademarks of Texas Instruments Incorporated.

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/pub/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
Technical Documentation Services, MS 702
P.O. Box 1443
Houston, Texas 77251-1443

Email: dsph@ti.com

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

Contents

TI offers a worldwide network of over hundreds of third parties and consultants who support the TMS320 DSP family. This chapter includes a summary of third-party support.

Describes the design workshops that can help you shorten your design cycle.

Texas Instruments believes it is important to train future engineers and encourages universities to do advanced research in the area of digital signal processing. This chapter describes the TI university program.

Figures

Tables

Examples

Chapter 1

Introduction

Texas Instruments (TI) has been the worldwide digital signal processor (DSP) market leader since 1982, with the introduction of the TMS32010 DSP. Since that first TMS320 DSP was introduced in 1982, Texas Instruments has been dedicated to the advancement of digital signal processing technology and its applications. TI recognizes that fast time to market, increased productivity, and design ease are of primary importance in the development of DSP-based applications. TI offers an innovative, comprehensive program of development support for TMS320 DSPs to facilitate the design process from system concept to production.

Topic **Page**

1.1 Why Use TI DSPs?

Since the introduction of the first TMS320 digital signal processor, the TMS320 family has established itself as the industry standard. The following list of characteristics highlights the advantages of designing with DSPs over other microprocessors:

❑ High-speed number crunching
❑ Real-time performance, simulation, and emulation
❑ Flexibility
❑ Reliability
❑ Increased system performance
❑ Reduced system cost

The advantages of the TMS320 family of DSPs are as follow:

❑ Broad portfolio

❑ Market leaders in solutions

❑ Manufacturing strength and commitment

❑ Mixed-signal support devices

❑ Wide variety of packaging options

❑ Better support from concept to completion

❑ Low-cost starter kits and evaluation modules

❑ Cycle-accurate simulators

❑ Optimizing high-level-language compilers

❑ Debuggers

❑ Real-time scan-based emulators

❑ Application software library

❑ Technical hotline and bulletin board service

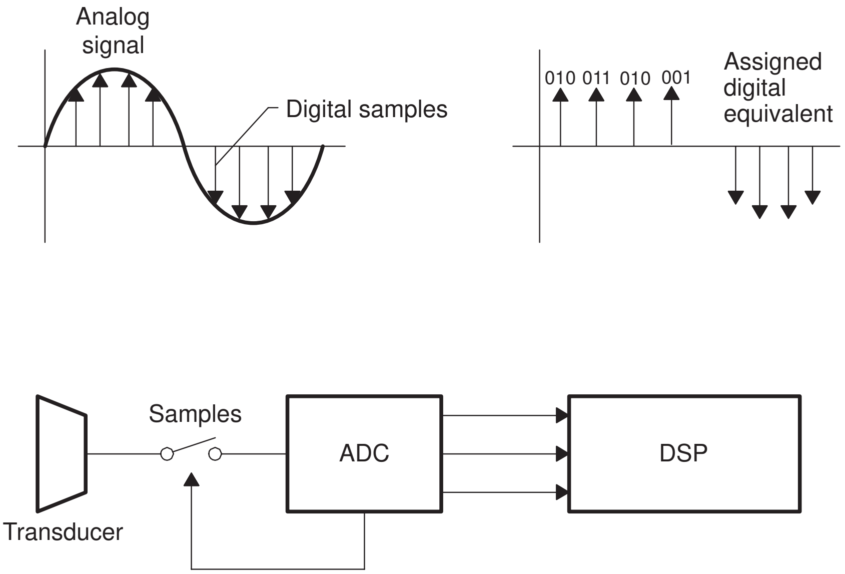
❑ Third-party support

❑ DSP Solutions web site, including TI & ME Internet Information Service, DSP On-Line Lab™, and Hotline On-Line™

1.2 From the Real World: DSP Technology

The natural environment that we wish to interpret is analog. Signals vary continually with time and may feasibly take any value. The digital domain is fixed between two values: high and low. Conversion from the analog domain to the digital domain (and back) is essential to sense signals, manipulate them, and reintroduce them to the environment. The advantages gained from digital manipulation justify the conversion. Special devices that perform this conversion are manufactured to interface with DSPs. These devices are called analog-to-digital (A/D) and digital-to-analog (D/A) converters. A binary code is assigned to a periodic sample of an analog signal that is in proportion to its magnitude; this gives a digital representation (see Figure 1–1).

Figure 1–1. A/D-D/A Conversion



Assigning a binary code to an analog signal introduces some error, as there is a limit to the accuracy to which an analog value can be defined. This error is called quantization.

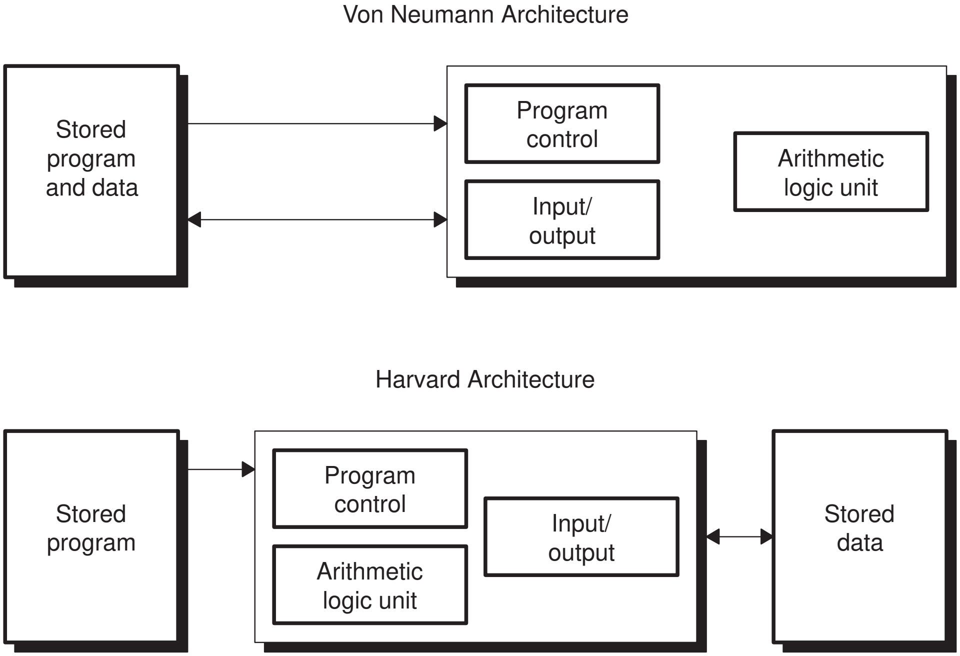
Texas Instruments provides a wide selection of different analog-to-digital converters (ADCs), digital-to-analog converters (DACs), and codecs (coders/decoders). A variety of performance levels and prices ensures that there is a device to suit your application.

1.2.1 DSP Architecture

All DSPs consist of several fundamental modules: a digital signal processing core to perform mathematical operations, memory to store data and program instructions, and possibly a mixed-signal product to converse between the analog and digital worlds.

As a stored-program machine, the processor must be told what to do every clock cycle. Typically, a DSP fetches an instruction and some data from memory, operates on these, and then returns the manipulated data to storage. The way this is conducted is not the same for all processors. Two different architectures can be identified: Von Neumann and Harvard (see Figure 1–2). The DSP application, in addition to the memory and peripheral configuration, usually governs the type of architecture employed.

Figure 1–2. Architectures of Digital Processors



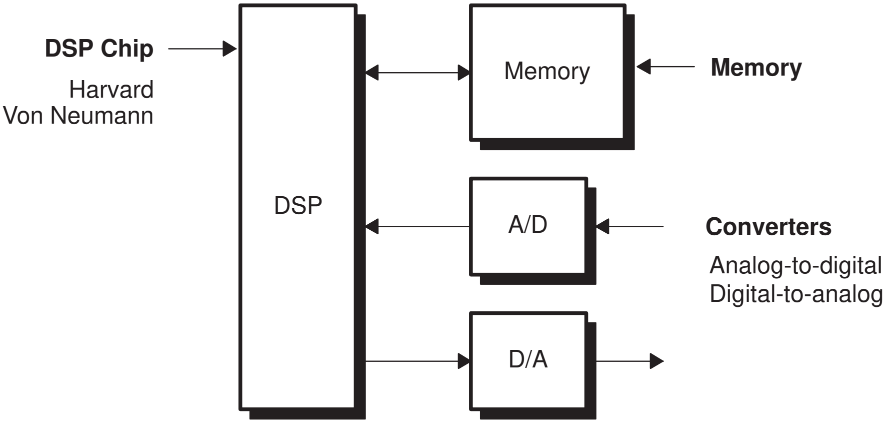
Von Neumann Architecture

Harvard Architecture

Von Neumann architecture has set the standard for computer development over the past 40 years. Essentially, the architecture is very simple. Both program and data can reside in the same memory-mapped space. This architecture forms a basis for more general-purpose processing needs, as seen in the x86 range. The disadvantage with this architecture is that there is only one bus that shares both data and memory addresses. Therefore, only data space or program space can be accessed in one cycle at any one time.

Where fast data manipulation is vital, accessing both program and data memory in a single cycle is advantageous. Harvard architecture separates the program and data memory spaces. Having two buses to serve each address space ensures that data and program access occurs in parallel, increasing processing speed. Unfortunately, processing power comes with a cost penalty. Two memory spaces require twice as many addresses, and therefore, twice as many data pins. An elegant solution has been found that is a compromise between price and performance. Modified Harvard architecture has only one external bus (thus reducing pin count), yet has both program and data internal buses. Many of the DSPs supplied by Texas Instruments support modified Harvard architecture to reduce cost for the customer while maintaining speed.

Figure 1–3. Typical DSP System



Some of the common features in a typical DSP system (see Figure 1–3) are as follows:

❑ A DSP to perform mathematical operations
❑ Memory (both on-chip and external) to store data and program instructions
❑ Converters to convert signals from analog to digital and from digital to analog

The DSP system may require external memory. In practice, the program to be run is normally held in external memory and then downloaded onto the DSP when the system is started.

1.2.2 The Need for Speed

The main concern for real-time algorithms is the amount of processing that can be done before a new sample arrives. DSP-type algorithms are generically of the form involving a multiply and add operation:

$$A = BC + D$$

The addition function is quite simple for conventional computers and can be performed in a single clock cycle. The same is true of subtraction. Most computers subtract by negating one number and then adding it to the other. Multiply functions take much longer, especially when considering numbers such as π. A general-purpose processor may take several hundreds of clock cycles to implement such a calculation. A machine is needed that can perform a multiply and an add in just one clock cycle. This requires an architecture molded to the specific application. DSPs have hardwired units within the processors for completing a multiply and add within one single clock cycle. Because such multiply and accumulate (MAC) instructions are the fundamental building blocks for many DSP applications, efficient execution is imperative.

Pipelining is an additional method of speeding up the instruction throughput of a processor, rather than speeding up the actual time to execute a single instruction. The simplest analogy is that of a car production line. It might take 10 hours to assemble a complete car, but because the construction of the car is broken down into many subsections, a car might be finished every 10 minutes. Computer instructions can be broken down into stages, such as fetching the instruction, decoding the instruction, fetching any data, executing the instruction, and storing the result. In a way similar to that of a production line, instructions are executed more quickly.

1.2.3 Example Architecture

Programmable DSPs can be categorized into two distinct groups according to their math type: floating point and fixed point. Each has a different architecture that benefits some applications and reduces the effectiveness of others.

❑ Fixed-point DSPs represent a number in a fixed range with a finite number of bits of precision. For example, a 16-bit processor will give a $\pm 2^{15}$ range. The earliest DSPs were based on this technology and, for the majority of applications today, the industry chooses 16-bit fixed-point processors. The price advantage gained from fixed-point 16-bit DSPs is significant.

❑ Floating-point DSPs express numbers between +1.0 and –1.0 using a mantissa. In addition, the representation also contains a scaling function called the exponent. This method of representation gives a greater dynamic range and therefore reduces the chance of overflow. Floating-point algorithms are well-suited to optimizing high-level-language (HLL) compilers. The reduced quantization error introduced in a 32-bit floating-point processor makes it ideal for audio applications.

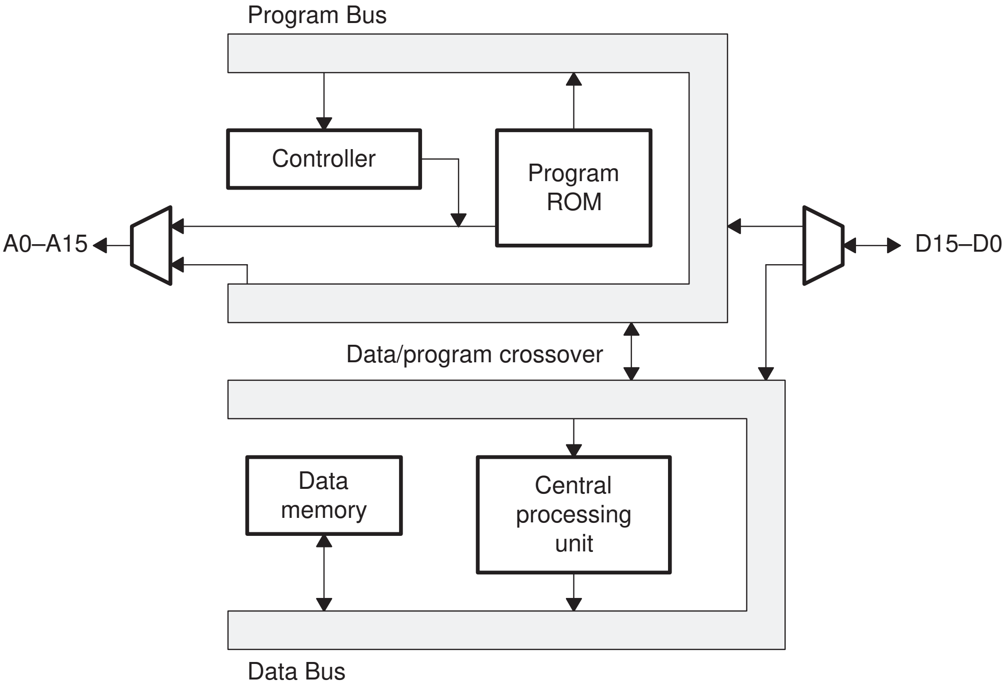
Multiprocessor systems such as the TMS320C80 introduce parallel processing to DSP on one-chip applications. Four fixed-point processors, working independently yet in a coordinated fashion, can process data much more quickly.

Figure 1–4, a simplified block diagram of a fixed-point DSP, shows the separation of data and program buses and the modification giving a crossover between them. Externally, the device could be mistaken for a von Neumann architecture processor, since it only has one set of data and program buses visible. Internally, data and program buses are separated with a crossover. The advantage gained on cost far outweighs the performance penalty.

The architecture is built around two major buses: the program bus and the data bus. The program bus carries the instruction code and immediate operands from program memory. The data bus interconnects various elements, such as the central arithmetic logic unit (CALU) and the auxiliary register file, to the data memory. Together, the program and data buses can carry data from on-chip data memory and internal or external program memory to the multiplier in a single cycle for multiply and accumulate operations.

The device has a high degree of parallelism; that is, while the data is being operated on by the CALU, arithmetic operations may also be executed in the auxiliary register arithmetic unit (ARAU). Such parallelism results in a powerful set of arithmetic, logic, and bit-manipulation operations that may all be performed in a single machine cycle.

Figure 1–4. Example Architecture



The internal hardware of the processor executes functions that other processors typically implement in software or microcode. For example, the device contains hardware for single-cycle 16 x 16-bit multiplication, data shifting, and address manipulation. This hardware-intensive approach provides computing power previously unavailable on a single chip.

1.3 Complete DSP Solutions

Texas Instruments not only designs, manufactures, and markets high-performance DSP semiconductors but also provides customer added value. An established network of customer support is already in place within Texas Instruments for your convenience. Design and training workshops, online help, bulletin boards, and comprehensive documentation are only a few of the ways that we help our customers find a complete DSP solution. We actively encourage the publication of papers in both industry and academic institutions and have substantial third-party support to help you in your projects. In our experience, a partnership with Texas Instruments enables you to extract the most from our products while bringing new and innovative designs to the marketplace.

The commitment of Texas Instruments to the customer is also seen in our range of peripheral products. While the heart of many systems may be the DSP, several devices are frequently required to support the solution. Texas Instruments provides chipsets to complement the most common application solutions, consisting of specifically designed peripherals such as codecs and memories. Such peripherals are designed for our processors, making interfacing simple. As with all Texas Instruments products, these chipsets are backed by our full support and quality assurance.

1.4 Typical Applications

The TMS320 family's unique versatility and real-time performance offer flexible design approaches in a variety of applications. In addition, TMS320 devices can simultaneously provide the multiple functions often required in those complex applications. Table 1–1 lists typical TMS320 family applications.

Table 1–1. Typical TMS320 Family Applications

General-Purpose DSP
Digital filtering
Convolution
Correlation
Hilbert transforms
Fast Fourier transforms (FFTs)
Adaptive filtering
Windowing
Waveform generation
Discrete cosine transforms
Hartley transforms

Instrumentation
Spectrum analysis
Function generation
Pattern matching
Seismic processing
Transient analysis
Digital filtering
Phase-locked loops

Control
Disk control
Servo control
Robot control
Laser printer control
Engine control
Motor control

Automotive
Engine control
Vibration analysis
Antilock brakes
Antiskid brakes
Adaptive ride control
Global positioning navigation
Voice commands
Digital radio
Cellular telephones
Active suspension
Noise suppression
Electronic power steering
4-wheel steering
Air bag control
System diagnosis
Radar detectors
Intelligent cruise control

Telecommunications
Hand-free speaker phones/ echo cancellations
ADPCM transcoders
Digital PBXs
Line repeaters
Channel multiplexing
1200- to 56,000 kilobit modems
Adaptive equalizers
DTMF encoding/decoding
Data encryption
Low-speed transcoders/ vocoders
ISDN basic/primary rate interfaces
FAX
Cellular telephones
Cordless telephones
Digital speech interpolation (DSI)
Packet switching and protocol
Videoconferencing/video compression/multimedia
Spread spectrum communications
Answering machines
Cable modems
Network switching
Modems

Consumer
Radar detectors
Power tools
Digital audio/TV
Music synthesizer
Educational toys
Answering machines
Multimedia
Digital cameras
Digital videodisk players
White goods (dishwashers, washing machines, etc.)
Karaoke
Feature phones
Arcade games
Set top boxes

Graphics/Imaging
3-D rotation
Robot vision
Image transmission/ compression
Pattern recognition
Image enhancement
Homomorphic processing
Workstations
Animation/digital map

Voice/Speech
Voice mail
Speech vocoding
Speech recognition
Speaker verification
Speech enhancement
Speech synthesis
Text-to-speech

Industrial
Robotics
Numeric control
Security access
Power line monitors
Active noise cancellation
Electronic meters

Computers
Laser printers/copiers
Scanner/bar-code scanner
Optical character recognition (OCR)
Neural networks
High-speed array processors
Imaging
Videoconferencing
Modems
Networking controller

Military
Secure communications
Radar processing
Sonar processing
Image processing
Navigation
Missile guidance
Radio frequency modems

1.5 Telecommunications Applications

Telecommunications is one of the fastest growing areas within the electronics market. Decentralization has necessitated more remote business control and collaboration between physically separated business entities. The information age has also generated a thirst for more data, creating the need for fast and effective communications.

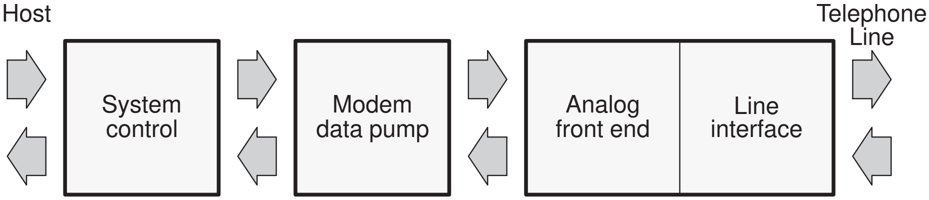
The telecommunications network is under great pressure as more data is being transmitted, and there is always a call for higher data-transmission rates. The high processing power achieved using DSPs provides a natural solution for the demands put on the telecommunications industry. Specific architectural advantages inherent in DSP devices reduce cycle times and operations per function. Consequently, implementing real-time telecommunication solutions relies heavily on DSP products.

1.5.1 Modems

The mathematically intensive calculations inherent in modem design dictate using the most powerful and sophisticated DSP performance levels. Typically, when modem data rates double for a given bandwidth, DSP performance must increase fourfold. Typical modem tasks are shown in Figure 1–5.

Modem evolution is dominated by the processing power of DSPs to the extent that virtually all modems available today use a DSP or a chipset with an embedded DSP equivalent. Application-specific DSPs and customizable DSP (cDSP) devices are also being increasingly used in modems for notebook and portable computers, where size, weight, and power consumption are key concerns.

Figure 1–5. Modem Tasks



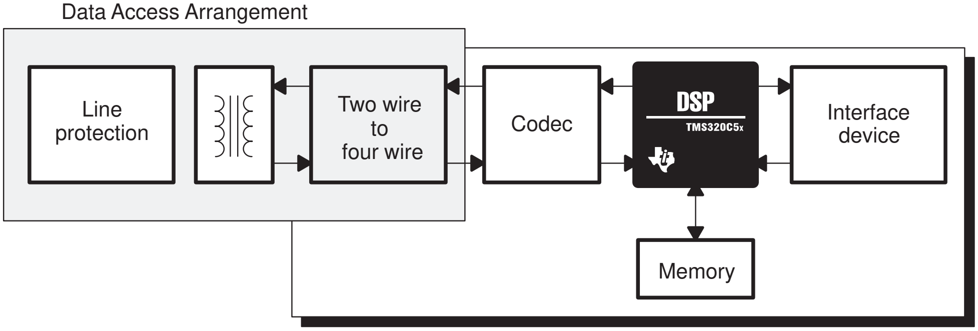
”AT” commands
Data compression
Error correction
Modem configuration
Protocol handling
Pulse dialing
On/Off-hook control
T.30 FAX protocol

Modulation/demodulation
Scrambler/descrambler
Decoder/encoder
Carrier generation
Fixed/adaptive equalization
Carrier/timing recovery
Echo cancellation
Filtering
Gain control
DTMF

A/D conversion
D/A conversion
2/4 wire hybrid line interface (DAA)

Originally, modem architectures included a microcontroller to govern the host interface and overall system control functions. Multiple DSPs were also required in parallel to manage baud rates in excess of 19.2 kbps. Today, DSPs from TI can implement a high-speed data modem on a single microchip. A TMS320C54x DSP with approximately 80 million instructions per second (MIPS) can perform both control and data pump functions at 33.6 baud (based on V.34 + standard). A 100-MIPS capacity allows further capabilities, such as voice/data or speakerphone, to be included in the design. Texas Instruments provides a complete modem chipset comprising core processor, codec, and memory to implement the most appropriate solution. Figure 1–6 shows the block diagram of a fax/modem design, which includes the dedicated mixed-signal products from TI.

Figure 1–6. TI Fax/Modem System Solutions



TI Digital Signal Processors
TMS320C54x
TMS320C5x
TMS320C32
TMS320C2xx

Coder/Decoders
Analog interface circuits
–TLC320AC01, –TLC320AC02
–TLC320AD55, –TLC320AD56

Data Access Arrangement
Low distortion, low power, supply of amps
–TLE2662, TL08X, –TLE2064,
–TLC2272, TLE2144

Interface Device
Internal UARTS
–TL16C550B/C, –TL16C554,
–TL16C750, –TL16PN550,
–TL16C552A
Stand Alone: RS232 driver/receivers
–SN75C188/9, –MC1488/1489
PCMCIA: UART + PCMCIA LOGIC
–TL16PC564A

Additional Circuitry
–Voltage supervisors (TL77XX)
–Voltage regulators (TL750LXX, 79LXX)
–Virtual grounds (TLE2425)
–Voltage converters (LT1054)

DSPs in modems are not only increasing processing power, they are also saving energy through low-power modes and providing circuit flexibility through high speed, smaller form factors, and greater functionality.

1.5.2 ISDN

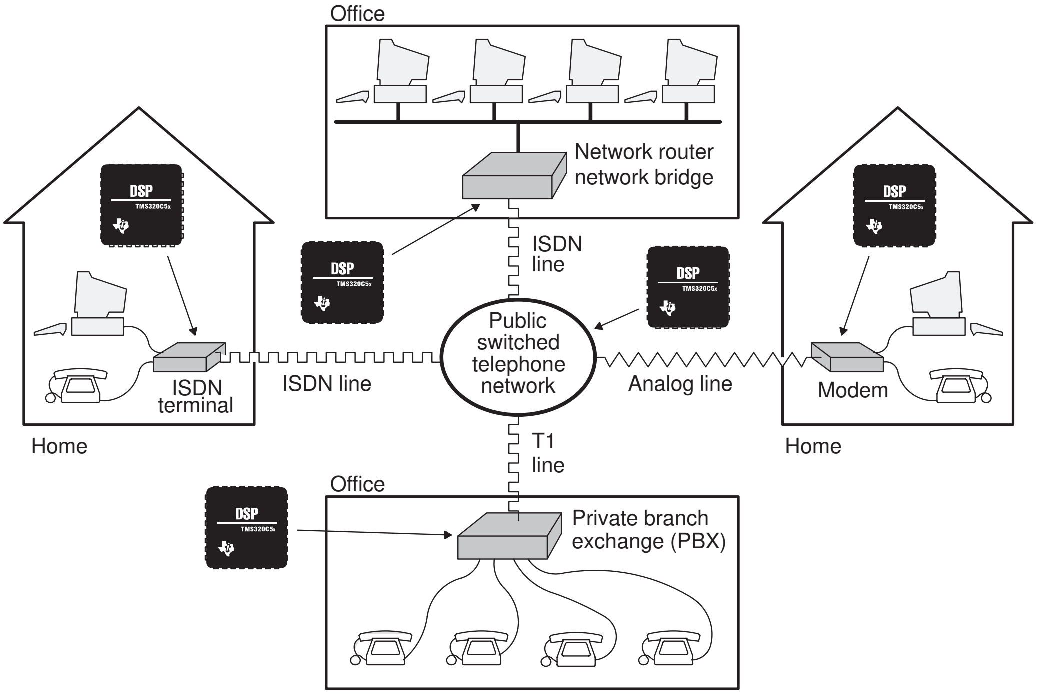
The Integrated services digital network (ISDN) offers a dramatic increase in the speed of data and document transfer at a significantly reduced cost. The speed achieved through ISDN empowers a new breed of applications:

- □ Interactive publishing
- □ Telecommuting
- □ Inexpensive videoconferencing
- □ LAN-to-LAN connectivity
- □ Teleradiology
- □ Remote health care
- □ Teleteaching
- □ Remote broadcasting
- □ Collaborative CAD/CAM engineering

ISDN brings the digital network to the individual user. The same twisted-pair copper telephone line that could traditionally only support one voice or one digital conversation can now carry as many as three separate conversations simultaneously through the same line.

Texas Instruments DSPs provide the necessary processing performance to ensure that the secure connection between multiple devices in the ISDN is maintained. TI DSPs support all the equipment incorporated in the ISDN interface device and in the public switched telephone network (PSTN), as shown in Figure 1–7.

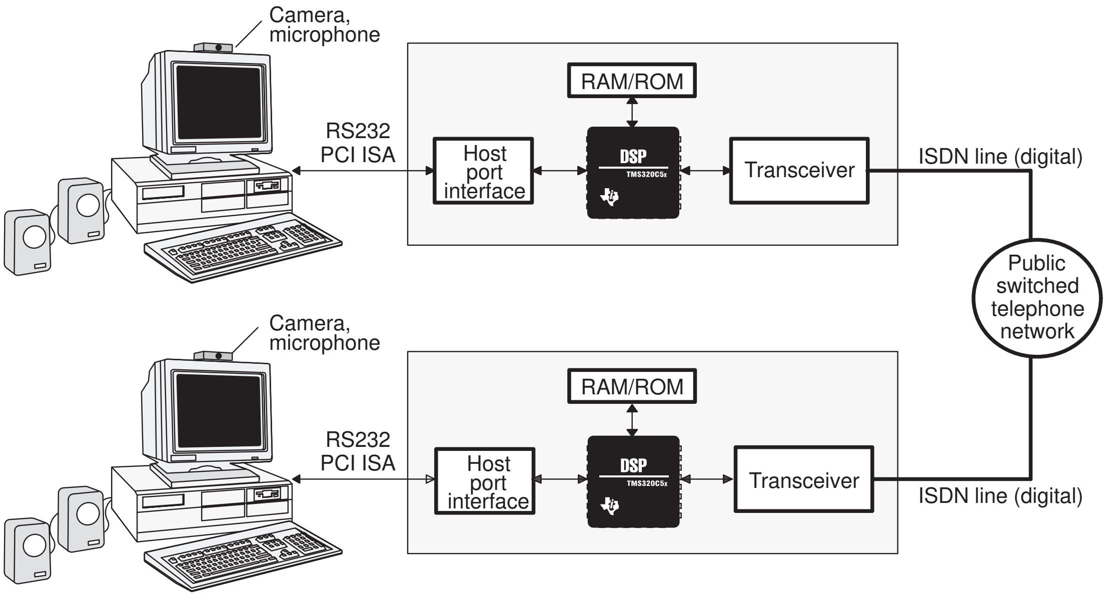
Figure 1–7. TI DSPs for ISDN Infrastructure



1.5.3 ISDN Videoconferencing

The high transmission rates associated with videoconferencing require significant bandwidth. Traditionally, such bandwidth requirements dictated dedicated broad-band lines that often had to be leased. It is now possible, through ISDN, to use videoconferencing without the added expense of leased lines (see Figure 1–8).

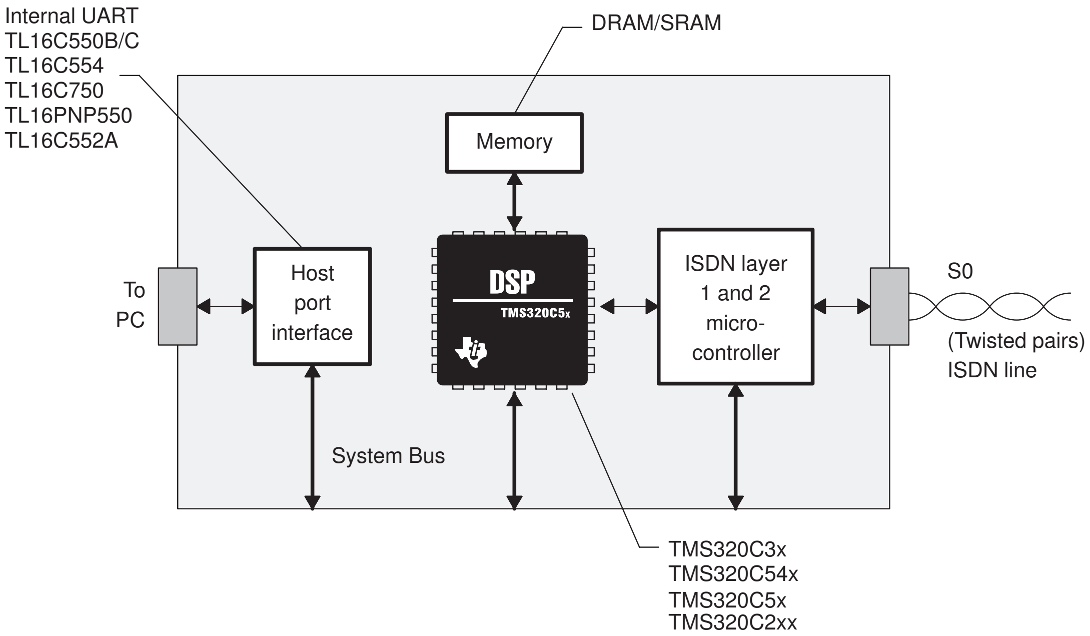
Figure 1–8. ISDN Terminal Videoconferencing



1.5.4 Dual-Mode ISDN Modems

A dual-mode modem (see Figure 1–9) communicates with either an ISDN terminal or an analog modem transceiver. In addition to executing algorithms such as acoustic echo cancellation and ISDN supervisory code, the modem must automatically switch between analog and digital mode. When communicating with an analog modem, the dual-mode modem switches to an analog mode. In this mode, the DSP performs modem modulation and demodulation. However, if the signal is sent through an ISDN, it must be transmitted in digital format, since the ISDN line is digital only.

Figure 1–9. Dual Mode ISDN Modem



TI DSPs can also be used for high-end ISDN telephone sets with features including hands-free function with echo cancellation, high-quality speech compression (G.722), and digital answering machine functions.

1.6 Audio Applications

Today, the consumer is demanding increasing functionality and multiple capabilities from electronic products. Computerization has led to digital systems rapidly replacing analog ones, giving better quality, flexibility, and performance. Where processing power limits what is achieved by conventional central processing units, DSPs provides a natural solution to satisfy the demands of the market. In most situations, lack of processing power results in poor quality, slow response time, and single tasking. A dedicated DSP has sufficient real-time processing power to cope with the real-time data manipulation that audio solutions require.

1.6.1 Multimedia

Multimedia is being driven by the conversion of traditional analog video signals to digital signals. Movies and video segments are already available on CDs for games, business presentations, and educational purposes. Digital video data will soon be transmitted over cable TV channels, telephone lines, and cellular channels and through satellites. The massive amounts of data generated by digitizing video require high-performance DSPs that reduce storage space and transmission bandwidth for cost-effective delivery of digital video.

Standard multipurpose CPUs work well at directing a variety of host functions; however, they are not well-suited for the multitasking and computing-intensive processing of real-time multimedia. Optimized for processing power, DSPs provide a cost-effective means of off-loading the digital signal processing from host CPUs. Telephone answering, speech and audio processing, sound effects, and music are rapidly becoming standard functions in workstations and PCs. DSPs are designed specifically to execute numeric-intensive tasks like these, where processing information must correspond to real-time events and several tasks are often performed simultaneously.

Texas Instruments offers programmable 16-bit fixed-point and 32-bit floating-point DSPs ranging from 20 MIPS to 1600 MIPS, as well as a family of application-specific MPEG audio and video decoders. Table 1–2 summarizes the compatibility of the TMS320 family with multimedia applications.

Table 1–2. TMS320 Digital Signal Processors for Multimedia

Application	**’C3x/’C4x**	**’C5x**	**’C54x**	**’C6x**	**’C8x**	**Mixed Signal**
Modems	X	X	X	X	X	X
Audio	X	X	X	X	X	X
Speech	X	X	X	X	X	X
Digital Tapeless Answering Device (DTAD)		X	X		X	X
Graphics/Imaging	X			X	X	
Video					X	X
Videoconference					X	X

1.6.1.1 DSPs for High-Speed Communications

For the cost-sensitive PC market, TI’s 16-bit TMS320C54x DSPs offer an optimum combination of price and performance. Fully programmable processors deliver up to 100 MIPS performance. Prices for the ’C54x family of devices start at less than $10. This processing power provides the freedom to cost-effectively implement very complicated applications, such as transmitting real-time voice and data simultaneously for telegaming and collaborative computing. With the ’C54x DSPs’ variety of speed, memory, and peripheral configurations, you will certainly find one to meet your needs.

The flexible 32-bit TMS320C32 is designed to bring the benefits of floating-point processing to the desktop. The ’C32 is driving the development of affordable speech and audio subsystems for videoconferencing, integrated voice mail, and video games. With 60 million floating-point operations per second (MFLOPS), the ’C32 also has the processing power to execute real-time three-dimensional graphics. High-performance DSPs like the ’C32 will continue to drive the evolution of a user-friendly multimedia environment.

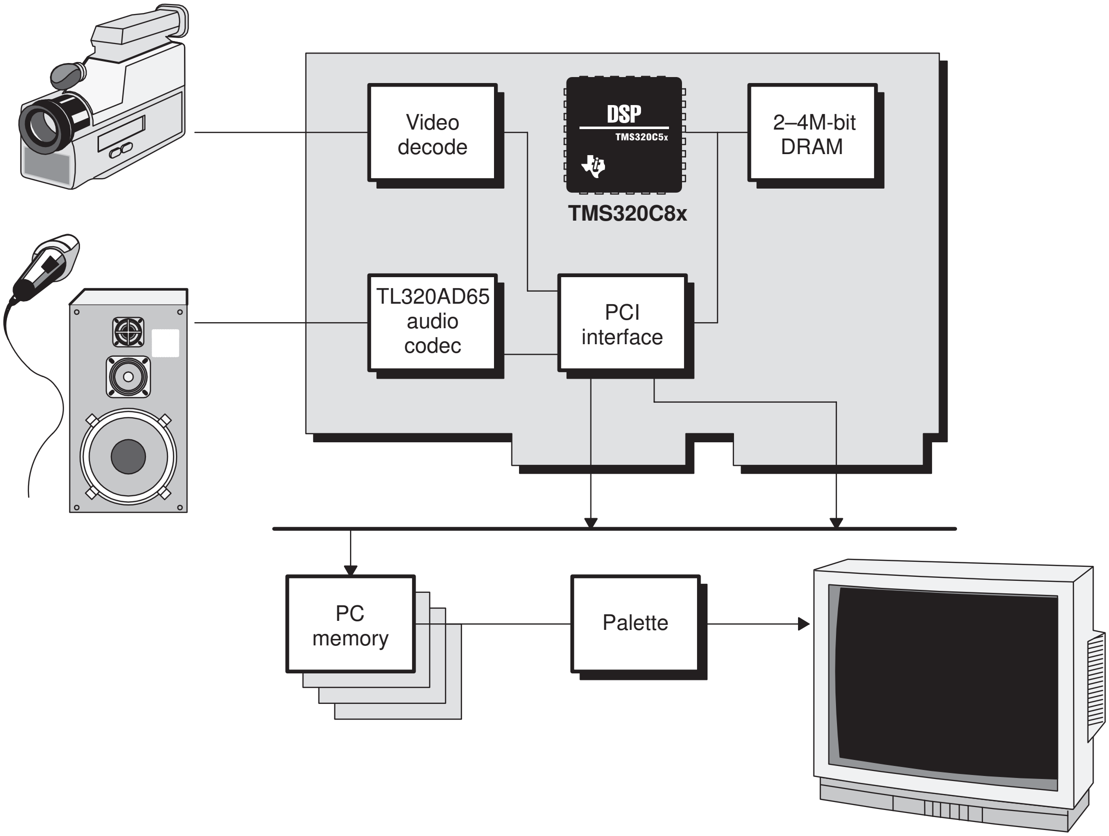
Some features of the ’C32 are:

- ❑ Voiceover data standards
- ❑ V.34 modem/V.17 fax
- ❑ Music synthesis
- ❑ Acoustic echo cancellation
- ❑ Full-duplex speaker phone
- ❑ Speech recognition
- ❑ Vocoders
- ❑ Voice mail
- ❑ Digital telephone answering functions
- ❑ Moving picture expert group (MPEG) audio
- ❑ Joint photographic experts group (JPEG)

The real-time multitasking capability of the ’C80 enables the parallel execution of video and audio encoding/decoding, video scaling, color space conversion, acoustic echo cancellation, filtering, error correction, multiplexing, and bit-stream protocol handling (see Figure 1–10). The ’C80 is the only single-chip DSP solution available today that offers these capabilities. Videoconferencing, video CD (MPEG-1 or MPEG-2), and collaborative computing are emerging technologies that are supported by the real-time manipulation of multiple data streams offered by the ’C8x. With its full programmability, the ’C80 provides the flexibility to respond to evolving standards and to create products with many features.

The ’C8x generation supports all industry-standard algorithms, such as H.320, MPEG, and JPEG, while offering designers the opportunity to blend their own proprietary algorithms to clearly differentiate their products. To simplify development of ’C8x-based video systems, TI now offers a library of popular functions and standard algorithms, including a complete H.320 software library. Additional standards libraries such as H.324 and imaging/graphics libraries are planned for the future.

Figure 1–10. Multimedia Opportunities



TI's family of TMS320AVxxx application-specific DSPs delivers the required combination of high performance and cost sensitivity. The 'AV110 and 'AV120 provide MPEG audio decoding for applications such as digital broadcast satellite systems, karaoke systems, Video CD players, and PC multimedia cards.

The new 'AV411 is a flexible, cost-effective Digital NTSC/PAL encoder that is designed for use in digital set-top boxes and DVD players. The support of multiple input and output data formats, on-chip as well as external sync signal generation, and 16-color on-screen overlay with closed-caption capability provides a video solution for getting from the digital world back to the television format.

1.6.1.2 MSLP 1394: The Standard for Multimedia

Multimedia data on the information superhighway must be processed in an ongoing, real-time manner by the entire system, not just by the DSP or host processor. A proposed IEEE standard (1394) that bridges computer and consumer applications is the emerging protocol for high-speed serial buses in multimedia systems. Other high-speed protocols exist, but only IEEE 1394 provides asynchronous data transfer (guaranteed bandwidth) to address the challenges of manipulating multiple data mediums on a real-time basis.

IEEE 1394 offers other benefits in addition to its guaranteed bandwidth for high-speed data transfer. An obvious advantage for both system designers and consumers is 1394's ability to interface with different types of peripherals. This in effect eliminates the numerous I/O interconnects normally required for peripherals like CD ROM, digital cameras, and printers. With the new standard, separate ports, including the parallel, audio, video, serial power, and SCSI ports, can be consolidated into an IEEE 1394 port — all connected by a single cable.

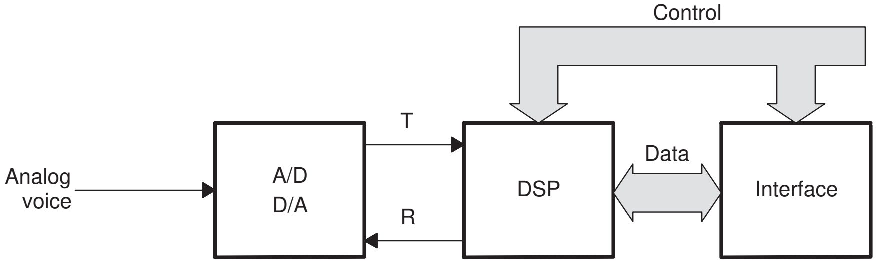
Data on the 1394 serial bus can be moved among a maximum of 63 different peripherals without ever burdening the host processor. This is vital for multimedia applications, where processing power is at a premium. The standard also enables both branching and daisy-chaining of nodes for increased flexibility and true *plug and play* capabilities required for next-generation operating systems.

TI currently offers the industry's first fully-compatible 1394 chipset. One manufacturer, Sony™, has already used the two-chip solution in prototypes for a 1394-based digital camera that will be used for both PC and workstation applications.

1.6.2 Speech

The need for embedded processing systems for speech in general-purpose computing machinery is growing fast. Multimedia speech processing applications include recognition, verification, voice mail, and text-to-speech voice processing, all of which are computationally demanding. The solution is to convert analog voice to a digital equivalent and then compress this digital signal for efficient storage and/or transmission. A general speech-processing system is shown in Figure 1–11.

Figure 1–11. Voice Processing



1.6.2.1 Speech Compression

Representing speech with a minimum number of bits while keeping the highest quality at the lowest cost, all within the application's environmental constraints, is the challenge of speech compression. Contemporary single-chip DSPs have made implementing real-time speech compression algorithms relatively inexpensive and easy and therefore commercially viable.

In general, high-quality speech compression at low bit rates is accomplished using very complex and computing-intensive coding algorithms. For example, a real-time implementation of a low-rate algorithm may require up to 40 MIPS of DSP processing power. Among the different types of speech coders, two families can be distinguished: waveform coders and parametric coders.

Waveform coders use direct quantization, that is, a binary representation of the speech samples themselves. They often operate in the time domain by quantizing the derivative of amplitude. Parametric coders, however, are based on a mathematical representation of speech model and spectral parameters. Most of the existing standards result from modifications and enhancements of a few different algorithmic approaches including adaptive differential pulse code modulation (ADPCM) and code excited linear prediction (CELP).

1.6.2.2 Speech Recognition

Automatic speech recognition lets computer users replace their keyboards with verbal instructions. Until recently, use of such systems, which carry out the user's spoken commands, have been restricted by the limitations inherent in voice recognition technology, as well as by the problems associated with language dialects and noisy environments. However, many of these limitations are being overcome through the use of powerful DSP technology.

Speech recognition is primarily used for telecommunications applications, such as hands-free dialing for cellular telephones and caller identification. Demand is also growing in industrial applications, including visual inspection processes, inventory control, and hands-free operations. Hands-free operations enhance safety in areas where dangerous machinery, toxic chemicals, or high temperatures pose a potential threat to workers. In the event of an emergency, the ability to issue commands to the equipment by voice allows the user to quickly shut down a process without having to search for a shut-down switch. In the multimedia world, speech recognition systems are beginning to allow users to navigate through their PC software applications by voice.

Today, high-performance DSPs are proving to be the enablers for industrial speech-recognition controls. DSPs are 10 to 50 times more powerful than other computer CPUs in terms of their ability to handle tasks as computing-intensive as speech recognition. This level of performance ensures that they can process data in real time. The bandwidth of information is reduced by digitally compressing communications signals. DSPs can process more information to meet the requirements of sophisticated speech recognition systems. For example, in systems requiring a high degree of security, designers can use DSPs in voice recognition systems to compare the user's voice pattern to a prerecorded voice print and grant access only if the two match. In addition, designers can use DSPs with noise-canceling and echo-canceling software to ensure higher degrees of reliability in noisy manufacturing environments.

1.6.3 Answering Machines

An existing Texas Instruments solution for a solid-state answering machine application — the digital tapeless answering machine or DTAD — is based on the chipset containing the MSP58C80 and the MSP58C20. This high-performance mixed-signal processor chipset has incorporated CMOS technology to give low power consumption. With its internal phase-locked loop (PLL), it is capable of running at an internal clock speed of 65.536 MHz by referencing an external oscillator of 4.096 MHz. Designing a DTAD system is simplified by using the glueless logic interface supported by the MSP58C80. The total system component count, and thus the cost, is minimal with a range of peripherals and advanced built-in DSP algorithms incorporated in the MSP58C80. Further cost saving in speech storage is provided by an internal 4.8-kbps MCELP vocoder, which can achieve close to 15 minutes of recording time using just 4M-bits of audio random-access memory (ARAM).

Texas Instruments DTAD chipset can be driven by an external host MCU. A powerful command set is included as a software interface to the MSP58C80 for implementation of a DTAD system. Programming using this command set is made very simple with the random access feature in the message management commands. The command set also supports direct control over the peripherals in the MSP58C80. Intelligent and flexible data transfer protocol is incorporated in the command set to minimize communication error.

The advantages of a DTAD are as follows:

- ❑ Digital recording with selectable compression rate: MCELP 4.8 kbps and 7.2 kbps
- ❑ Recording time up to 20 minutes per 4 Mbits at 4.8 kbps with silence compression
- ❑ Directs parallel MCU interface (eight data plus four control lines)
- ❑ Supports direct external ROM interface for customized DTS, voice menu, or add-on DSP functions
- ❑ On-chip ARAM refresh with direct 4Mx1, 1Mx4, 16Mx1, or 4Mx4 bits ARAM interface
- ❑ 4.096-MHz external crystal with internal PLL
- ❑ Real-time clock function
- ❑ Two 8-bit general-purpose user I/O ports (MSP58C024)
- ❑ Three 8-bit ADC inputs with user-defined thresholds for event monitoring
- ❑ User-programmable single/dual tone generator with adjustable tone level
- ❑ Call progress tone detect for U.S., Germany, France, and United Kingdom
- ❑ Reliable DTMF detect
- ❑ Ring detect with programmable bandwidth
- ❑ Internal day-time stamp with voice clock (English version — male voice)
- ❑ Ease of use and powerful MCU command set
- ❑ Supports up to eight mailboxes
- ❑ Random access of out-going message (OGM) record, playback, delete, and message information retrieval
- ❑ Allows a total of 128 messages in any combination of OGMs, incoming messages (ICMs), or memos
- ❑ Two power-down mode options
- ❑ +5 V single voltage supply
- ❑ 100-pin QFP package with MSP58C80, 20-pin wide body small outline IC (SOIC) with MSP58C20

A typical DTAD is based on the MSP58C80 and the MSP58C20. The MSP58C80 supports a direct interface for an external host MCU and for x1 or x4 ARAM. Its on-chip peripherals also allow direct control for external analog switching, supervision of power supply circuitry, and ring signal detection, so that a solid-state answering machine can be implemented with a minimal component count. The MSP58C20 is a delta-sigma converter that provides analog-to-digital conversion for the MSP58C80. With the MCU version of MSP58C80, you would be shielded from the complexity of the MSP58C80 and be able to impart more flexibility to your products by using separate MCU code.

System architecture cost for stand-alone answering machines can be optimized by suppressing the MCU. All the functions are then taken over by the DSP.

1.7 Control Applications

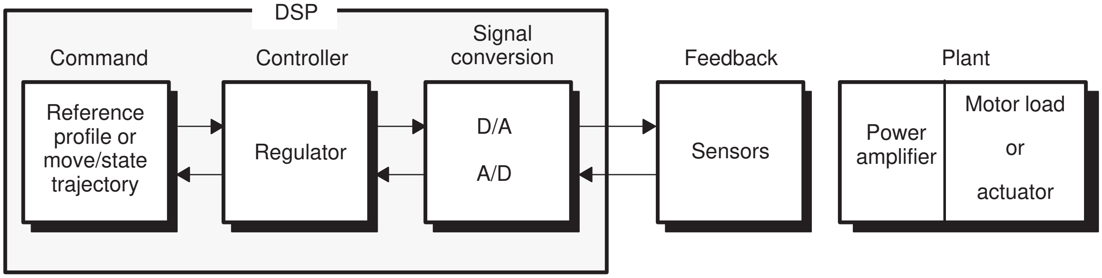
Digital signal processors are probably the most truly embedded of all controllers. Their presence is of little significance to the user, yet their effects are far-reaching. In a world of automation, DSP is becoming increasingly important in applications such as networking, signal reproduction, and power generation. The embedded nature of DSP and the high level of computational power enables control of the most complicated system.

1.7.1 Motor Control

Today's generations of automated systems must be designed with environmental consciousness in mind, with more efficient power conservation and a more robust control function implementation. Such systems are now possible through the use of digital motor control technologies (see Figure 1–12).

To address this growing market, TI offers a broad line of semiconductor technology oriented to motor control applications, as well as an extensive selection of development tools and third-party support. Optimized specifically for digital motor and motion control, TI offers the TMS320C24x DSP controllers, as well as other TMS320 digital signal processors. The TMS370 family of microcontrollers and the industry's only family of integrated power switching transistors completes TI's spectrum of motor control solutions.

Figure 1–12. Digital Motor Control



- Command generation: spline, polynomial, lookup table
- Controller: PID, LQR, Kalman filter, self-tuning regulator, model reference adaptive control, notch filter, fuzzy logic, gain scheduling
- Other DSP Tasks:

  – Communication control for brushless multiphase Motor
  – PWM signal generator for 1- to 3- phase motor
  – Vector control of induction motors
  – System modeling and diagnostics

Motor control systems have traditionally been implemented using analog and passive components such as operational amplifiers, resistors, capacitors, and voltage regulators. Alternatively, control may be performed digitally by converting a discrete sample of the analog input signal to a digital equivalent. The input signal is not processed continuously but is sampled at discrete intervals. A sampling interval of at least six to ten times the bandwidth of the system is usually implemented, placing significant performance demands on the system processor.

Due to the high performance and low cost of digital signal processors and microcontrollers, digital motor control systems are replacing analog controllers in today’s designs.

Advantages of DSP-based motor control include:

- ❑ Real-time generation of smooth reference and move profiles
- ❑ Integration of memory (lookup tables) or multiple processors into a single DSP
- ❑ Use of advanced algorithms, resulting in fewer sensors and lower system cost
- ❑ Vector control of brushless and induction motors
- ❑ Control of power switching inverters and the generation of high-resolution pulse-width-modulation (PWM) outputs
- ❑ Control of multivariable and complex systems using modern intelligent methods such as neural networks and fuzzy logic

1.7.2 Laser Printers and Copiers

The TMS320C6x generation of DSPs can turn the office copier into an intelligent document management station. Desktop scanners already have the ability to read a document into digital form, while laser printers can convert a digital file into a paper copy. Linking these two machines through a microprocessor, as some new small-business-oriented products do today, produces a rudimentary digital copier. Unfortunately, these machines are restricted to making simple copies and usually reproduce images with artifacts introduced by the scanning process.

Digital copiers enhance the reproduction process by applying digital signal processing techniques. By compressing images and storing them on disk, the digital copier increases the reliability and simplifies the process of reproducing multiple collated copies. The use of digital processing for improving on analog and electromechanical methods has existed for some time. However, more revolutionary products will emerge as advanced image/signal processing is applied to do more than simply copy documents.

DSP solutions are the key to the transformation of copiers into document processing workstations. Already, high-performance DSP devices, such as the TMS320C62xx and TMS320C3x, offer the processing power to take copying machines to a higher level.

A single processor that can handle both image printing and recognition reduces the overall system cost. When scanning a document for faxing, the ’C8x could analyze the document and correct it for rotation, also carrying out the process know as grid stretching in order to send a document. This would eliminate most of the “jaggies” commonly associated with faxed material. Similarly, image processing could clean up a document sent by a machine that did not have “jaggy removal.”

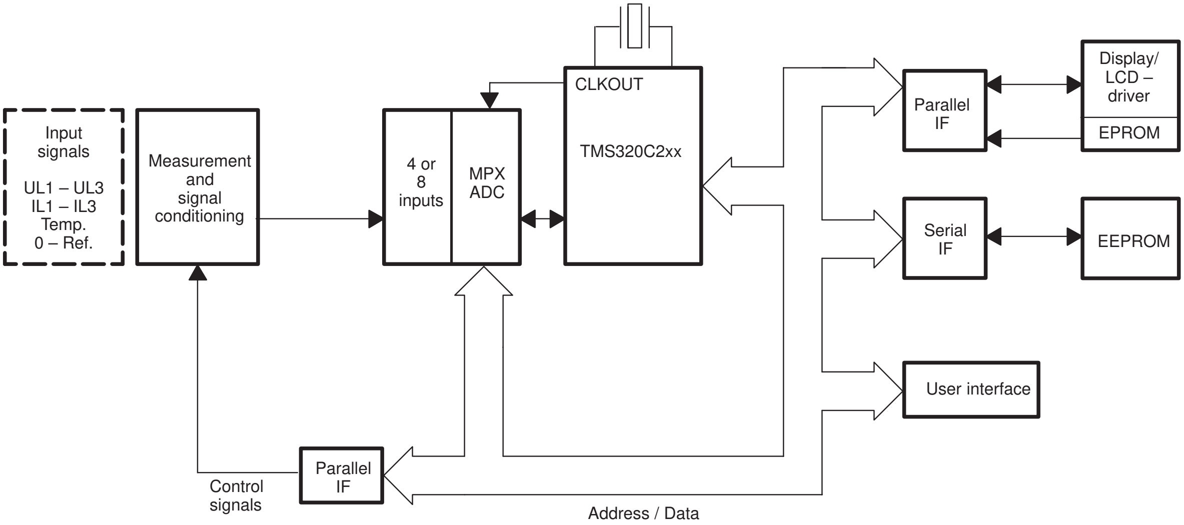
The move to digital processing on high-end, high-volume copiers and color copiers is already prolific. However, much of the digital processing in these products is dedicated to specific copying tasks. These copiers are using digital processing to improve their reliability, ease of use, and image quality.

1.7.3 High-End Metering With DSPs

Programmability, low power consumption, and the advantage of reduced maintenance and operating costs of electronic meters are the driving forces behind the replacement of the Ferrari’s wheel meters. DSPs are a key feature of such systems and are suited to high-feature (multitariff) 1-phase or 3-phase meters. These applications require 16- x 16-bit multiply operations in real time at a high sampling rate to measure the harmonic content of the current. Traditional solutions using low-cost controllers would require one microcontroller for each phase, whereas a single DSP can handle all three phases on its own at relatively low clock frequencies.

A DSP solution based on standard devices, such as the ’C2xx, would be as shown in Figure 1–13. The low-power mode and the 16-bit architecture of the ’C2xx family have enabled DSPs to be suitable for high-end metering applications.

Figure 1–13. TMS320C2xx-Based Electricity Meter



With special scanning and computation algorithms, the power consumption of a 3-phase-system load can be measured with minimum error. System calibration, self-checking, and setup menus for system parameter modification are implemented in software.

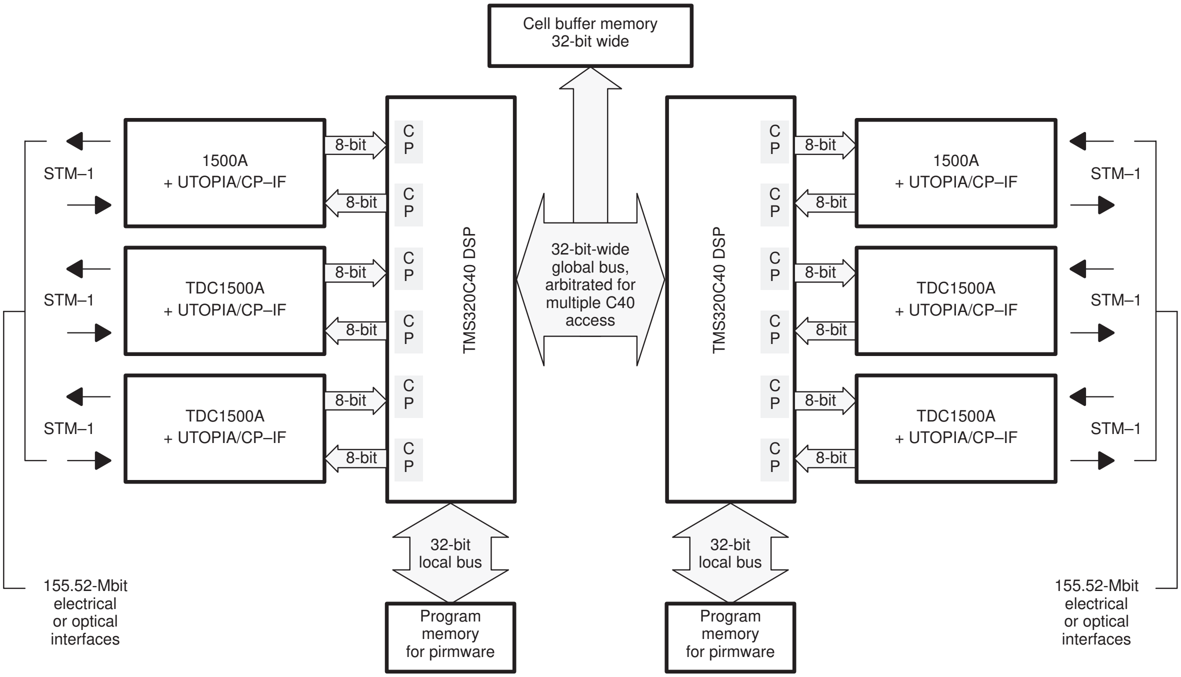
In high-volume applications, a customizable DSP (cDSP) could integrate the interface functions on-chip. These functions include LCD drivers, a remote meter-reading interface, analog-to-digital converters, a smart card interface, and an external memory interface. Such a single-chip solution would result in lower standby currents and improved battery life, thereby reducing maintenance costs.

1.7.4 Networking Controllers

Emerging applications such as videomail, virtual reality, and interactive television require flexible transmission bandwidths. In addition, they must have their own worldwide standards and maintain compatibility with existing systems and networks.

ISDN and multimedia networks must therefore have the capability of transmitting a combination of signals of varying bandwidths simultaneously on the same transmission line. High-performance networks already exist with the capacity to transmit image, voice, and data for these applications cost-effectively and in real time. asynchronous transfer mode (ATM) is becoming the worldwide standard for high-speed data communication. Texas Instruments offers solutions based on standard products like the ’C40 for ATM switching (see Figure 1–14).

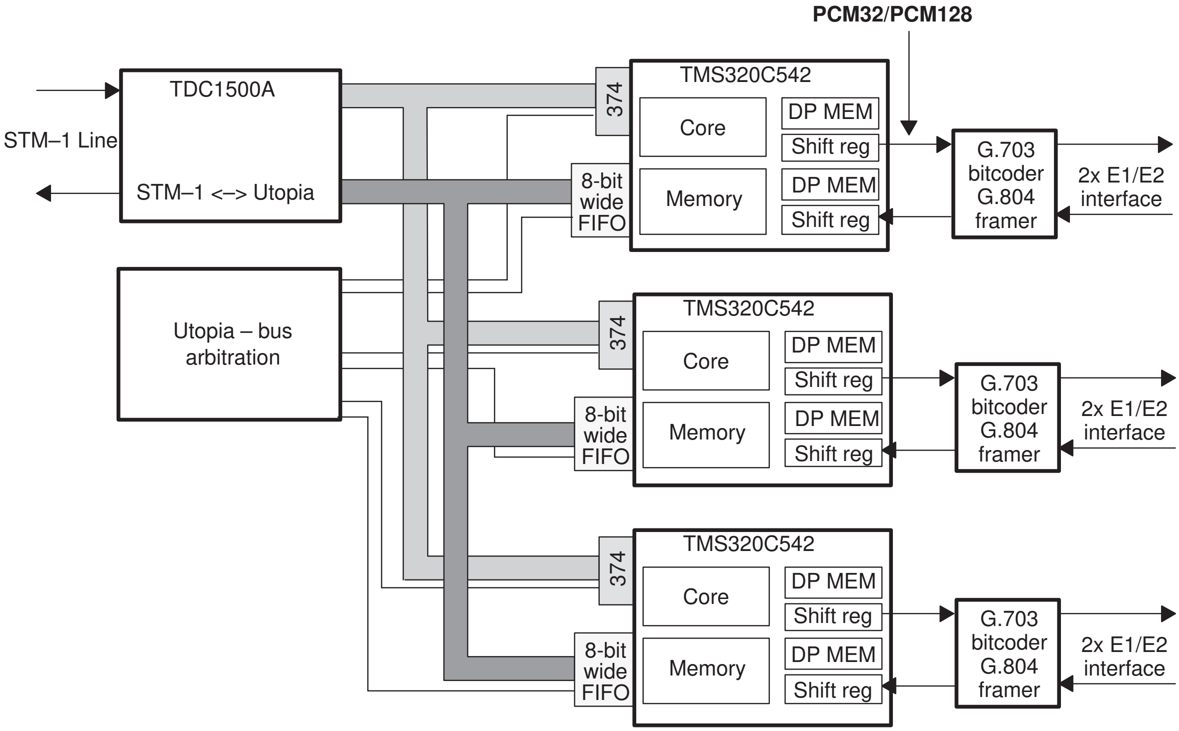
Figure 1–14. ATM Switch Based on a TMS320C40



The data flow to and from the TDCs is controlled via internal direct memory access (DMA) channels. Data bandwidths of 60M bytes/s per ’C40 are achievable with the system. For data exchange to other lines, the ’C40’s global bus can be shared by several other processors. Access arbitration, internal data routing, and signaling are all performed with software. The TNETA1500 single-chip line interface to STM-1/STC3c scrambles and descrambles the signal and facilitates framing, idle cell insertion, and extraction. It also generates status messages on dedicated error lines for fast reaction. This configuration allows a fast time to market and can be used as a cheap emulation tool.

DSPs can also be used to connect narrow-band ISDN networks with ATM networks, as shown in Figure 1–15. The ’C542 offers 10K words of on-chip RAM, used for cell buffering. Internal logic interfaces directly to PCM32/PCM128. OAM flow, control flow, and feedback control are performed by software. Connection, rewiring, and software updates are done by OAM cell booting. This configuration allows flexible algorithms like multiple SRTS handling in those areas where specifications are not fully defined.

Figure 1–15. N-ISDN to ATM Switch Based on a TMS320C542



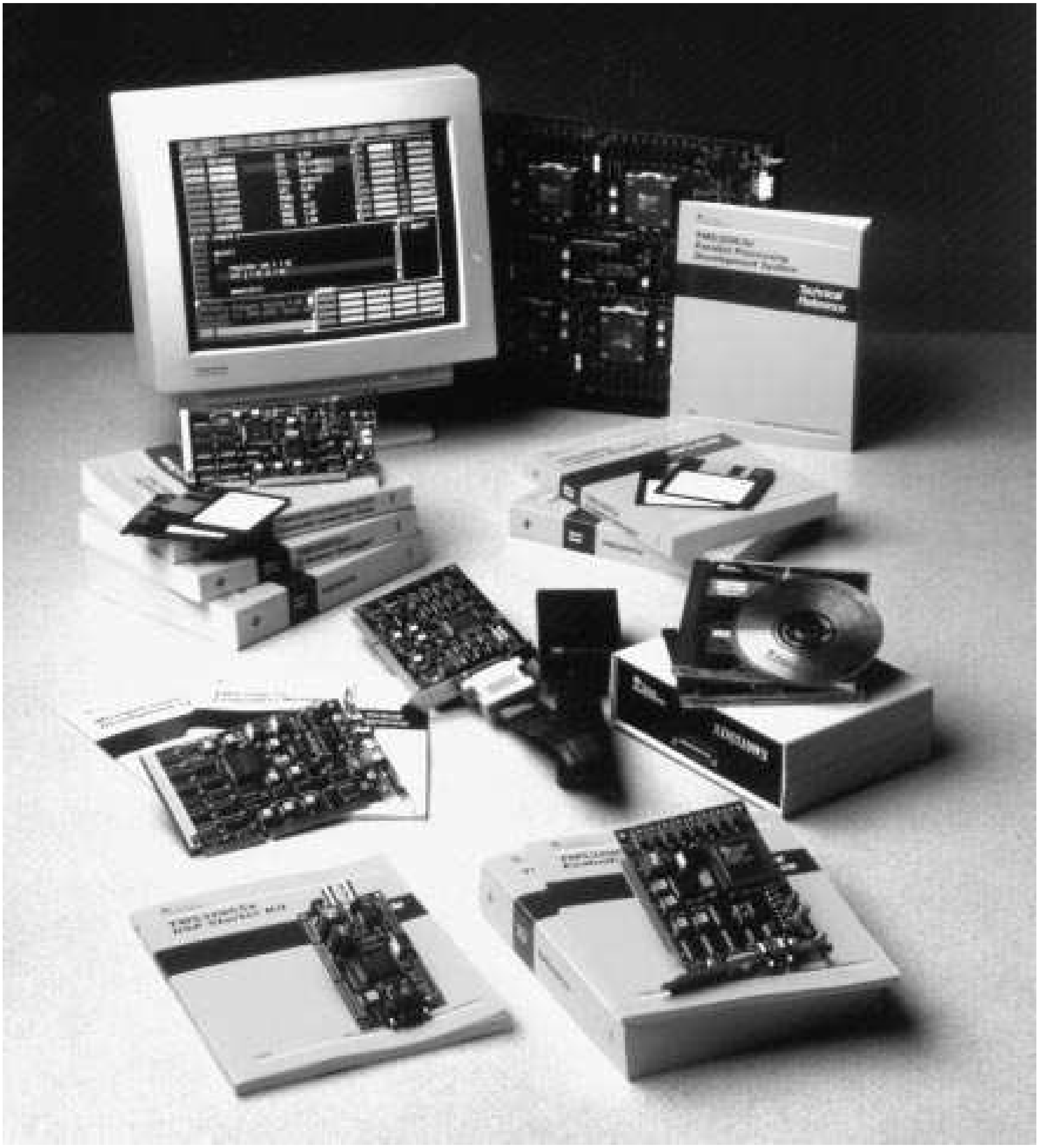
1.8 TI DSP Support Overview

The TMS320 support program includes leading-edge hardware and software development systems from TI and TMS320 third parties: optimizing C compilers, a user-friendly programming interface consisting of C/assembly language source debuggers with code-profiling capabilities, low-cost evaluation tools, simulators, real-time emulators, real-time operating systems, and application software. More than ever, TMS320 DSP users enjoy a development environment that is comparable to the environment available for general-purpose microprocessor systems. Figure 1–16 shows the wide range of development tools available.

Various other support services are also available through the technical hotline, World Wide Web and FTP sites, Bulletin Board Service, field technical staff, and the Technical Training Organization. A library of textbooks and over 2500 pages of application notes provide extensive information about the TMS320 DSP products.

The TMS320 third-party product offerings and consultant services can be found in Chapter 16 of this book.

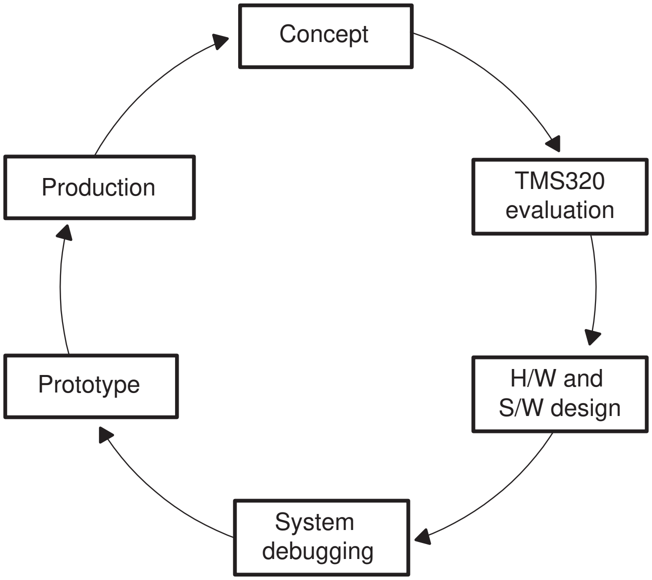
Figure 1–16. TMS320 Family Development Support



1.9 TMS320 Development Support Products

Texas Instruments supports designers in complete application development, from concept through production. TI offers an extensive line of development support products to assist you in all aspects of TMS320 design and development. These products range from development and application software to complete hardware integration and debugging systems. Figure 1–17 shows a typical application flow. Products to support this flow are described in the list that follows the figure.

Figure 1–17. Typical TMS320 Application Development Flow



❑ **Concept.** Application development for a new or upgraded design often begins with research. TMS320 developers have numerous resources for conceptual designs, including system benchmarks, application notes, algorithms, user's guides, Internet sites, the TI field technical staff, and the DSP hotline.

❑ **TMS320 system evaluation.** Support tools for design evaluation include DSP starter kits (DSKs), evaluation modules (EVMs), simulators, assembler/linkers, and compilers from TI and from third parties. Using these tools, a developer can benchmark code and determine single or multiple DSP system configurations. TI's extensive documentation provides the necessary information on specifications and capabilities.

❑ **Hardware and software designs.** You can design these modules in parallel by using a TMS320 simulator, assembler/linker, compiler, or EVM for software development and by using TMS320 behavioral models and emulators for hardware development. The hotline, Internet sites, and field technical staff offer technical support during this phase; technical documentation and third-party tools are also available.

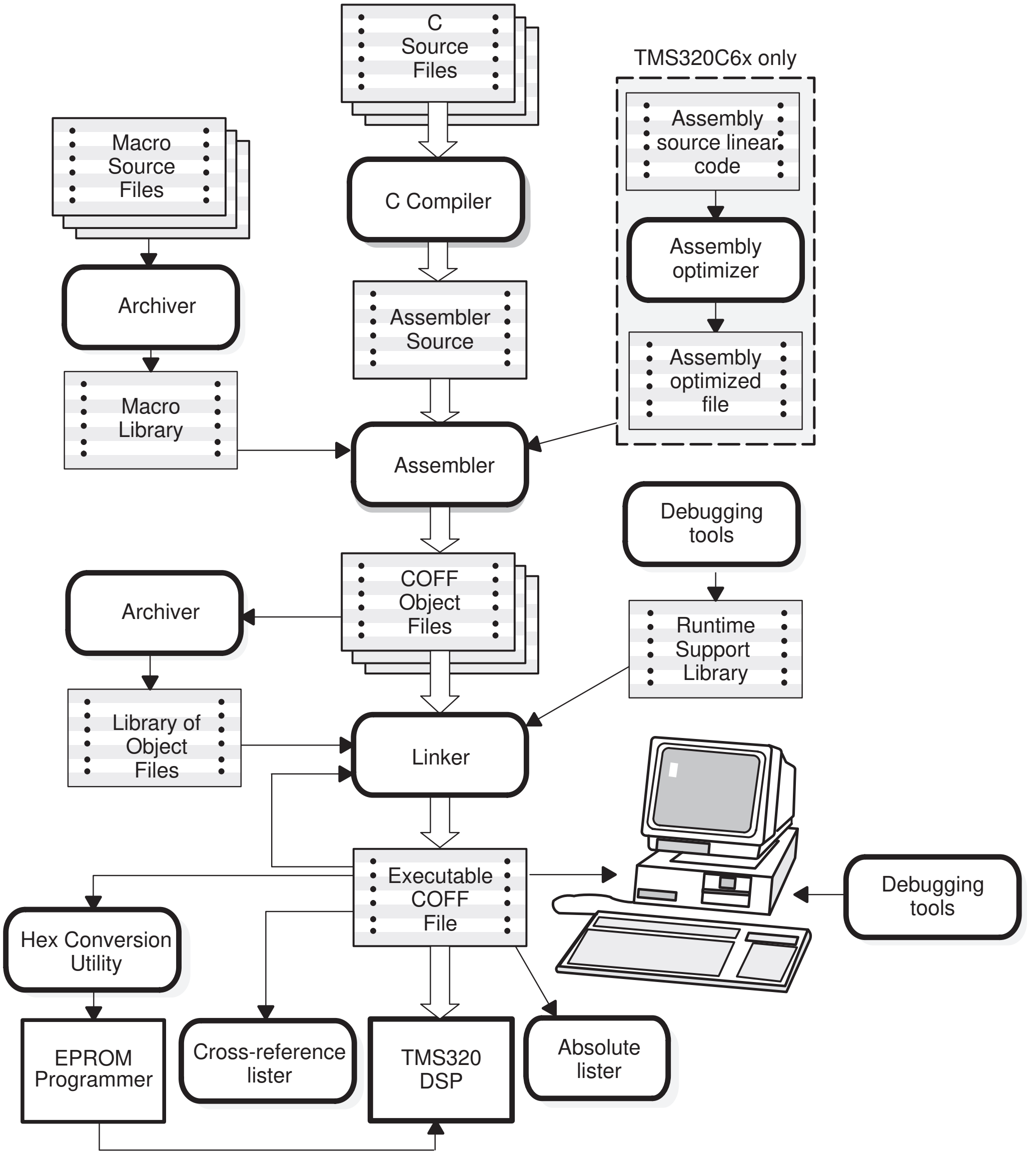
❑ **System Debugging.** Typically, the next phase is integration of the software and hardware modules and debugging of the entire DSP system. You can use emulators and the new source-level debugger at this stage; technical assistance is available from the hotline and/or field technical staff.

❑ **Prototype.** When you complete your system prototype, you can submit and/or release your device's ROM code to TI through the BBS. One-time-programmable and Flash DSPs provide for early prototype development and smooth the transition to the production phase.

❑ **Production.** Once system production begins, you can design a system upgrade. TMS320 family compatibility, a well-defined product migration path, and high-level-language compilers facilitate this phase of the system development cycle.

Figure 1–18 shows development product integration. The appropriate TMS320 support product is indicated for each stage of development. Table 2–11 on page 2-24 provides a matrix of the features of the TMS320 simulation/emulation development tools, comparing capabilities such as development purpose and software and hardware features.

Figure 1–18. TMS320 Development Product Integration



Chapter 2

Selection Guide

Choosing the right DSP for your needs is an important process, and one that can be confusing given the wide range of choices. This chapter is intended to help with the selection process.

Topic **Page**

2.1 TMS320 Family Overview

The TMS320 family consists of 16-bit fixed-point, 32-bit floating-point, and 64-bit multiprocessor single-chip DSPs. These processors have the operational flexibility of high-speed controllers and the numerical capability of array processors. Combining these two qualities, the TMS320 processors are inexpensive alternatives to custom-fabricated VLSI and multichip bit-slice processors. The following characteristics make this family the ideal choice for a wide range of processing applications:

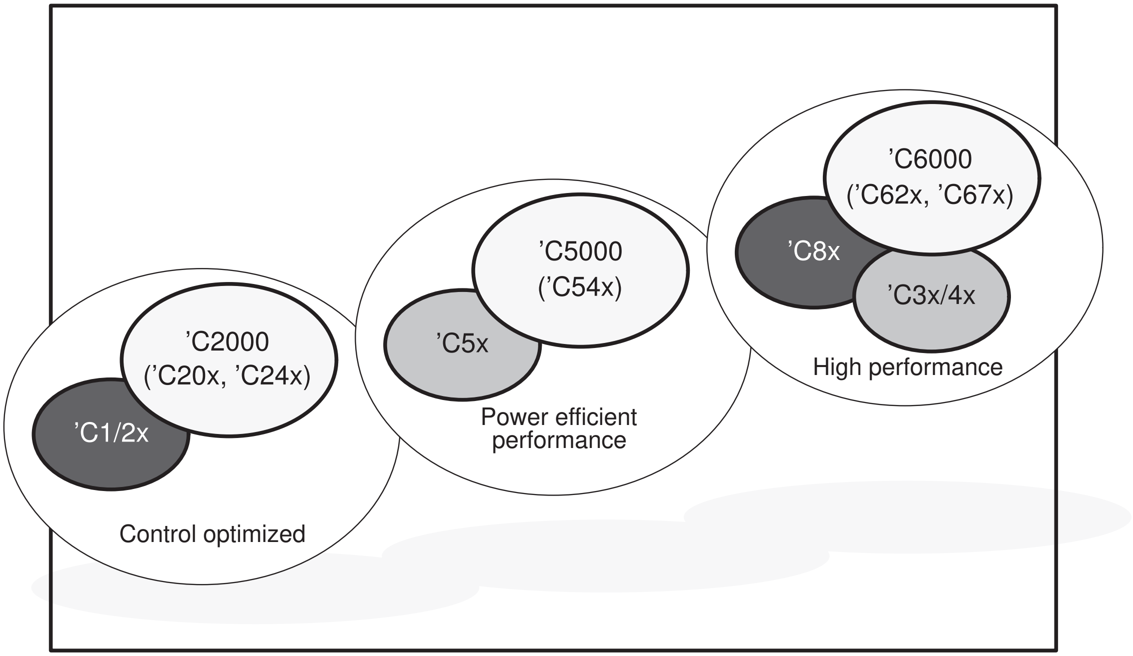
❑ Flexible instruction set
❑ Inherent operational flexibility
❑ High-speed performance
❑ Innovative, parallel architecture
❑ Cost-effectiveness

In 1982, Texas Instruments introduced the TMS32010—the first fixed-point DSP in the TMS320 family. The TMS32010 became the model for future TMS320 generations.

Today, the TMS320 family (see Figure 2–1) consists of these generations: the ’C1x, /C2x, ’C20x, ’C24x, ’C5x, ’C54x, and ’C62x 16-bit fixed-point generations; the ’C3x ’C4x, and ’C67x 32-bit floating-point generations; and the ’C8x 64-bit multiprocessor generation. These generations are all complemented by mixed-signal products such as data converters.

Each generation of TMS320 devices has a similar structure, a CPU combined with a variety of on-chip memory and peripheral configurations. New combinations of on-chip memory and peripheral options are used to create spin-off devices that 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 board space is saved.

Figure 2–1. The TMS320 Family Road Map



2.2 TMS320 Device Naming Conventions

This section explains the nomenclature used in the TMS320 device numbering system.

To classify the stages in the product development cycle, Texas Instruments assigns prefix designators in the part number nomenclature. In addition to the prefix, the device nomenclature includes a two-part suffix that follows the device's family name. This suffix indicates the package type (for example, FN) and temperature range (for example, L). The speed (in MHz or MIPS, depending on the device), if included, follows the temperature suffix (for example, –40). Figure 2–2 provides a legend for reading the complete nomenclature of any TMS320 family member. TMX and TMP devices are shipped with the following disclaimer:

Developmental product intended for internal evaluation purposes

Note:

Texas Instruments recommends that prototype devices (TMX or TMP) not be used in production systems, because their expected end-use failure rate is undefined but is predicted to be greater than the failure rate of standard, qualified production devlces.

TMS devices and SMJ devices have been properly tested, and the quality and reliability of the devices have been successfully demonstrated. TI's standard warranty applies.

Figure 2–2. TMS320 Device Nomenclature

TMS 320 C 25 FN L –40

Prefix

SMJ	=	MIL-PRF-38535 (QML)
SMQ	=	MIL-PRF-38535 (QML) plastic
TMX	=	experimental device
TMP	=	prototype device
TMS	=	qualified device

Device Family

320	=	TMS320 family

Technology

No letter	=	NMOS
AV	=	audio/video encoders or decoders
BC	=	CMOS with ROM bootloader
C	=	CMOS
E	=	CMOS EPROM
F	=	CMOS with flash memory
LBC	=	low-voltage CMOS with ROM bootloader
LC	=	low-voltage CMOS
LF	=	low-voltage CMOS with flash memory
P	=	CMOS one-time-programmable ROM
VC	=	very low voltage

Device

1x-generation microprocessors/microcomputers:
10, 14, 15, 16, 17

2x-generation microprocessors: 25, 26

2xx-generation microprocessors: 203, 206, 209,
240, 241, 242, 243

3x-generation microprocessors: 30, 31, 32

4x-generation microprocessors: 40, 44

5x-generation microprocessors:
50, 51, 52, 53, 56, 57

54x-generation microprocessors:
541, 542, 543, 545, 546, 548, 549,

6x-generation microprocessors
6201, 6701

8x-generation microprocessors: 80, 82

AVxxx-generation microprocessors:
110, 120, 411

Speed
(in MHz or MIPS)

Temperature Range

A	=	–40° to 85°C
H	=	0° to 85°C
L	=	0° to 70°C
M	=	–55° to 125°C
S	=	–55° to 100°C

Package Type*

FD	=	LCCC
FJ	=	JLCC
FN	=	PLCC
FZ	=	CERQUAD
GB	=	CPGA
GE	=	CPGA
GF	=	CPGA
GFA	=	CPGA
GFW	=	PBGA
GGU	=	BGA (ball grid array)
HFG	=	CQFP
HFH	=	CQFP
J	=	CDIP
JD	=	CDIP SB
KGD	=	known good die
N	=	plastic DIP
PCM	=	PQFP
PDB	=	PQFP
PE	=	PQFP
PG	=	PQFP
PGE	=	TQFP
PH	=	PQFP
PN	=	TQFP
PJ	=	PQFP
PPM	=	PQFP
PQ	=	PQFP
PZ	=	TQFP
TA	=	TAB (encapsulated)
TAB	=	TAB (encapsulated)
TB	=	TAB (bare die)
TBB	=	TAB (bare die)

Note: Questions on availability should be directed to the nearest TI Sales Office, Authorized TI Distributor, or TI Semiconductor Product Information Center (PIC).

2.3 TMS320 DSP Overview

The powerful instruction sets, inherent flexibility, and innovative architectural designs have made the TMS320 digital signal processor family the ideal solution for many automotive, computer, consumer, industrial, military, and telecommunication applications. An overview of the TMS320 family (see Table 2–1) consists of the following generations of devices. Further expansion of this family is planned, creating even higher-performing spin-offs and new generations. Presently, the key members (by generation) of the TMS320 family are:

- TMS320C2xx generation
  - TMS320C203 — a fixed-point, 16-bit DSP with 544 words of on-chip RAM and two serial ports that runs up to 80 MHz. Pin compatible with the ’F206
  - TMS320LC203 — a low-power version of the ’C203
  - TMS320C206 — 16-bit, fixed-point DSP with 4.5K words of on-chip RAM, 32K words with ROM
  - TMS320F206 — a 16-bit, fixed-point DSP with 4.5K words of on-chip RAM, 32K words of on-chip flash memory, and two serial ports that runs up to 80 MHz. Pin compatible with the ’C203 and ’C205
  - TMS320LC206 — low-power version of ’C206
  - TMS320C209 — a 16-bit, fixed-point DSP with 4.5K words of on-chip RAM and 4K words of ROM that runs up to 57 MHz
  - TMS320C240 — a 16-bit, fixed-point DSP with an optimized event manager, dual on-chip 10-bit analog-to-digital converters, SPI and SCI ports, 16K words of ROM, 28 bidirectional I/O pins, and a watchdog timer
  - TMS320F240 — a 16-bit, fixed-point DSP with an optimized event manager, dual on-chip 10-bit analog-to-digital converters, SPI and SCI ports, 16K words of flash memory, 28 bidirectional I/O pins, and a watchdog timer
  - TMS320C241 — a 16-bit, fixed-point DSP with 544 words of on-chip RAM, 8K words of on-chip ROM, event manager, 10-bit analog-to-digital converter, SCI and SPI ports, Controller Area Network (CAN) module
  - TMS320F241 — a 16-bit, fixed-point DSP with 544 words of on-chip RAM, 8K words of flash EEPROM, event manager, 10-bit analog-to-digital converter, SCI and SPI ports, Controller Area Network (CAN) module

- TMS320C242 — a 16-bit fixed-point DSP with 544 words of on-chip RAM, 4K words of on-chip RAM, event manager module, 10-bit analog-to-digital converter
- TMS320F243 — a 16-bit fixed-point DSP with 544 words of on-chip RAM, 8K words of flash EEPROM, event manager, 10-bit analog-to-digital converter

Table 2–1. TMS320 Device Overview

’C2xx	’C3x	’C4x	’C5x	’C54x	’C6x	’C8x	AVxxx
C203	C30	C40	C50	C541	C6201	C80	AV110
LC203	C31	C44	LC50	LC541	C6701	C82	AV120
C206	LC31		C51	C542			
F206	C32		LC51	LC542			
C209	LC32		C52	LC543			
C240			LC52	LC545			
F240			C53	LC546			
C241			LC53	LC548			
F241			C53S	LC549			
C242			LC53S	VC549			
F243			LC56				
			LC57				
			BC57S				

❑ TMS320C3x generation

- TMS320C30 — a 33-million floating-point operations per second (MFLOPS) (33-MHz), 32-bit, floating-point DSP with two memory expansion buses, two serial ports, on-chip ROM, 2K words of on-chip RAM, one channel DMA, and CMOS technology
- TMS320C31 —similar to the ’C30 with one memory expansion bus, no on-chip ROM, 2K words of on-chip RAM, one serial port, a bootloader, and one channel of DMA
- TMS320LC31 — a low-power version of the ’C31

  - TMS320C32 — the lowest-cost version of the 'C3x with a flexible memory interface, 512 words of on-chip RAM, low-power modes, a bootloader, and two channels of DMA with configurable priorities
  - TMS320LC32 — a low-power version of the 'C32

- ❑ TMS320C4x generation
  - TMS320C40 — a high-performance, 330-MOPS, 384 Mbytes/s, 32-bit floating-point, multiport, parallel-processing digital signal processor
  - TMS320C44 — a lower-cost version of the 'C40 with four communications ports and a smaller address reach

- ❑ TMS320C5x generation
  - TMS320C50 — a complete system on a single chip. With 2K 16-bit words boot ROM and 10K 16-bit words on-chip RAM, an entire DSP can be integrated into a 132-pin plastic quad flat pack (PQFP).
  - TMS320LC50 — a low-voltage version of the 'C50; 3.3-V power supply
  - TMS320C51 — in the 'C51, the 'C50's 10K-word block of RAM is replaced by 8K x 16-word ROM. This provides a considerable advantage in cost and performance for users who require large amounts of on-chip program space.
  - TMS320LC51 — a low-voltage version of the 'C51; 3.3-V power supply
  - TMS320C52 — a superb combination of both low cost and high performance. Traditionally, devices in the same price range have offered 10 million instructions per second (MIPS) performance.
  - TMS320LC52 — a low-voltage version of the 'C52; 3.3-V power supply
  - TMS320C53 — provides even greater integration of on-chip ROM than the 'C51. With 16K 16-word on-chip ROM and 4K 16-word on-chip RAM, an entire DSP system can be integrated into the 132-pin PQFP.
  - TMS320LC53 — a low-voltage version of the 'C53; 3.3-V power supply
  - TMS320C53S — similar to the 'C53 specification with the exception of two standard serial ports, the 100-pin thin quad flat pack (TQFP), and preprogrammed ROM.
  - TMS320LC53S — a low-voltage version of the 'C53S; 3.3-V power supply
  - TMS320LC56 — a 3.3-V device with 32K 16-word on-chip ROM and 7K 16-word on-chip RAM, integrated into a 100-pin TQFP. The communication ports consist of a standard serial port and a buffered serial port (BSP). Available at 35- or 25-ns cycle times

  - TMS320LC57—similar to the 'LC56 with an additional parallel host port interface (HPI) and a 128-pin TQFP.
  - TMS320BC57S—a 5-V device with 2K 16-word boot ROM and 7K 16-word on-chip RAM. The communication ports are similar to the 'LC57's and consist of a standard serial port, a host port interface, and a buffered serial port.

- ❑ TMS320C54x generation
  - TMS320C541 — available with an instruction cycle time of 25 ns for 5K words of RAM and 28K-words of ROM and a 5-V power supply
  - TMS320LC541 — low-voltage versions of the 'C541, available with instruction cycle times of 25 or 20 ns, and 3.3-V power supplies
  - TMS320C542 — available with an instruction cycle time of 25 ns for 10K-words of RAM and 2K-words of ROM and a 5-V power supply
  - TMS320LC542 — low-voltage versions of the 'C542 available with instruction cycle times of 25 or 20 ns, and 3.3-V power supplies
  - TMS320LC543 — available with instruction cycle times of 25 or 20 ns, 3.3-V power supplies, and the same memory mix as the 'C542 but with fewer peripherals
  - TMS320LC545 — available with instruction cycle times of 25 or 20 ns, 3.3-V power supplies, and 6K-words of RAM and 48K-words of ROM
  - TMS320LC546 — available with instruction cycle times of 25 and 20 ns, 3.3-V power supplies, and the same memory mix as the 'C545 but with fewer peripherals
  - TMS320LC548 — available with instruction cycle times of 15 or 12.5 ns, 3.3-V power supplies, and 32K-word RAM and 2K-word ROM
  - TMS320LC549/VC549 — available with instruction cycle times of 15, 12.5, or 10 ns, 3.3- or 2.5-V power supplies, and 32K-word RAM and 16K-word ROM

- ❑ TMS320C6x generation
  - TMS320C6201 — available with an instruction cycle time of 5ns. Based on the VelociTI VLIW 'C6200 fixed-point CPU core. Executes up to 8 instructions per clock cycle and achieves performance of 1600 MIPS @200MHz. Contains 1 M-bit on-chip static RAM (SRAM) and 32-bit external memory interface (EMIF). 3.3v I/O, 2.5v CPU core.
  - TMS320C6701 — floating point version of the TMS320C6201 available with an instruction cycle time of 6ns. Based on the VelociTI VLIW

'C6700 floating-point CPU core. Executes up to 8 instructions per clock cycle and achieves performance of 1 billion floating-point operations per second (1 GFLOPS) single precision (420 million floating-point operations per second double precision). Contains 1 M-bit on-chip static RAM (SRAM) and 32-bit external memory interface (EMIF). 3.3V I/O, 2.5V CPU core.

- TMS320C8x generation
  - TMS320C80 — a high-performance 2-billion-operations-per-second (BOPS), 400 Mbytes/s multiprocessor device with up to 400 Mbytes of on-chip memory. The 'C80 offers 50K bytes of static RAM (SRAM) and can handle up to 64-bit instruction words.
  - TMS320C82 — a cost-effective version of the 'C80 without the video controller, and with only two parallel processors, and with some added features

Table 2–2 through Table 2–9 give detailed operating characteristics of the commercial devices. Table 2–10 lists similar information about each military DSP chip.

Table 2–2. TMS320C2xx Commercial Devices[†]

				Memory (Words)						Peripherals				
				On-Chip			Off-Chip							
Device Name[‡]	Freq (MHz)	Cycle Time (ns)	MIPS	RAM	ROM	Flash	Data	Prog	I/O	Serial Port	Timers	Package Type	Typ Diss (mW)	Temp Range (°C)
’C203PZ	40	50	20	544	–	–	64K	64K	64K × 16	2	1	100 TQFP (p)[§]	190	0/70
’C203PZ–57	57	35	28.5	544	–	–	64K	64K	64K × 16	2	1	100 TQFP (p)[§]	270	0/70
’C203PZ–80	80	25	40	544	–	–	64K	64K	64K × 16	2	1	100 TQFP (p)[§]	380	0/70
’LC203PZ	40	50	20	544	–	–	64K	64K	64K × 16	2	1	100 TQFP	73	0/70
’LC203PZA	40	50	20	544	–	–	64K	64K	64K × 16	2	1	100 TQFP	73	–40/85
’C206[¶]	80	25	40	4.5K	32K	–	64K	64K	64K × 16	2	1	100 TQFP	400	0/70
’LC206	80	25	40	4.5K	32K	–	64K	64K	64K × 16	2	1	100 TQFP	400	0/70
’F206PZ	40	50	20	4.5K	–	32K	64K	64K	64K × 16	2	1	100 TQFP (p)[§]	190	0/70
’C209PN	40	50	20	4.5K	4K	–	64K	64K	64K × 16	–	1	80 TQFP (p)[§]	190	0/70
’C209PN–57	57	35	28.5	4.5K	4K	–	64K	64K	64K × 16	–	1	80 TQFP (p)[§]	270	0/70
’C240	40	50	20	544	16K	–	64K	64K	64K × 16	2	4	132 PQFP	400	–40/85
’F240	40	50	20	544	–	16K	64K	64K	64K × 16	2	4	132 PQFP	400	–40/85
’C241	40	50	20	544	8K	–	64K	64K	64K × 16	2	4	68 PLCC 64 PQFP	400	–40/85
’F241	40	50	20	544	–	8K	64K	64K	64K × 16	2	4	68 PLCC 64 PQFP	400	–40/85
’C242	40	50	20	544	4K	–	64K	64K	64K × 16	2	2	68 PLCC 64 PQFP	400	–40/85
’F243	40	50	20	544	–	8K	64K	64K	64K × 16	2	2	144 TQFP	400	–40/85

[†] Industrial temperature, military devices, and automotive temperature qualification devices are also available.
[‡] See Figure 2–2 for explanation of device nomenclature.
[§] Plastic package
[¶] The core operates as 3.3V; I/O operates at 5V.

Table 2–3. TMS320C3x Commercial Devices

					Memory (Words)				Peripherals					
					On-Chip			Off-Chip						
Device Name†	Freq (MHz)	Cycle Time (ns)	MIPS	MOPS	RAM	ROM	Cache	Parallel	Serial Port	Timer	DMA Channels	Package Type	Typ Diss (mW)	Temp Range (°C)
’C30GEL	33	60	16.667	183.337	2K	4K	64	16M x 32‡	2	2	1§	181 CPGA	1000	0/85
’C30GEL–40	40	50	20	220	2K	4K	64	16M x 32‡	2	2	1§	181 CPGA	1250	0/85
’C30GEL–50	50	60	25	275	2K	4K	64	16M x 32‡	2	2	1§	181 CPGA	1500	0/85
’C31PQL–40	40	50	20	220	2K	BL	64	16M x 32	1	2	1§	132 PQFP	900	0/85
’LC31PQ–40	40	50	20	220	2K	BL	64	16M x 32	1	2	1§	132 PQFP	700	0/85
’C31PQL–50	50	40	25	275	2K	BL	64	16M x 32	1	2	1§	132 PQFP	1100	0/85
’C31PQL–60	60	33	30	330	2K	BL	64	16M x 32	1	2	1§	132 PQFP	1300	0/85
’C31PQA–40	40	50	20	220	2K	BL	64	16M x 32	1	2	1§	132 PQFP	900	–40/125
’C32PCM–40	40	50	20	220	512	BL	64	16M x 8/16/32	1	2	2¶	144 PQFP	900	0/85

† See Figure 2–2 for explanation of device nomenclature.

‡ Two external parallel buses

§ Fixed priority

¶ Configurable priority

Table 2–3. TMS320C3x Commercial Devices (Continued)

					Memory (Words)				Peripherals					
					On-Chip			Off-Chip						
Device Name[†]	Freq (MHz)	Cycle Time (ns)	MIPS	MOPS	RAM	ROM	Cache	Parallel	Serial Port	Timer	DMA Channels	Package Type	Typ Diss (mW)	Temp Range (°C)
’C32PCMA–40	40	50	20	220	512	BL	64	16M x 8/16/32	1	2	2[¶]	144 PQFP	900	–40/125
’C32PCM–50	50	40	25	275	512	BL	64	16M x 8/16/32	1	2	2[¶]	144 PQFP	1100	0/85
’C32PCMA–50	50	40	25	275	512	BL	64	16M x 8/16/32	1	2	2[¶]	144 PQFP	1100	–40/125
’C32PCM–60	60	33	30	330	512	BL	64	16M×8/16/32	1	2	2	144 PQFP	1300	0/85

[†] See Figure 2–2 for explanation of device nomenclature.
[‡] Two external parallel buses
[§] Fixed priority
[¶] Configurable priority

Table 2–4. TMS320C4x Commercial Devices

					Memory (Words)				Peripherals					
					On-Chip			Off-Chip						
Device Name[†]	FREQ (MHZ)	Cycle Time (ns)	MIPS	MOPS	RAM	ROM	Cache	Parallel	Serial Port	Timer	DMA Channels	Package Type	Typ Diss (mW)	Temp Range (°C)
’C40GFL–50	50	40	25	275	2K	BL	128	4G x 32	–	2	6 (12)	325 CPGA	1500	0/85
’C40GFL–60	60	33	30	330	2K	BL	128	4G x 32	–	2	6 (12)	325 CPGA	1800	0/85
’C44PDB–50	50	40	25	275	2K	BL	128	32M x 32	–	2	6 (12)	304 PQFP	1800	0/85
’C44PDB–60	60	33	30	330	2K	BL	128	32M x 32	–	2	6 (12)	304 PQFP	1800	0/85
’C44GFW–50	50	40	25	275	2K	BL	128	32M x 32	–	2	6(12)	388 BGA		0/85
’C44GFWA–50	50	40	25	275	2K	BL	128	32M x 32	–	2	6(12)	388 BGA		–40/125
’C44GFW–60	60	33	30	330	2K	BL	128	32M x 32	–	2	6(12)	388 BGA		0/70

[†] See Figure 2–2 for explanation of device nomenclature.

Table 2–5. TMS320C5x Commercial Devices

				Memory (Words)						Peripherals					
				On-Chip			Off-Chip								
Device Name[†]	Freq (MHz)	Cycle Time (ns)	MIPS	RAM	ROM	OTP	Data	Prog	I/O	Serial Port	Timers	HPI	Package Type	Typ Diss (mW)	Temp Range (°C)
’C50PQ	40	50	20	10K	BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	525	0/70
’C50PQ-57	57	35	28.57	10K	BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	590	0/70
’C50PQ-80	80	25	40	10K	BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	825	0/70
’C50PQA	40	50	20	10K	BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	525	–40/85
’C50PQA-57	57	35	28.57	10K	BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	590	–40/85
’LC50PQ	40	50	20	10K	BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	155	0/70
’BC51PQ-57	57	35	28.57	2K	8K	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	590	0/70
’BC51PQ-80	80	25	40	2K	8K	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	825	0/70
’BC51PQ-100	100	20	50	2K	8K	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	975	0/70
’BC51PQA-57	57	35	28.57	2K	8K	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	590	–40/85
’BC51PZ–57	57	35	28.57	2K	8K	–	64K	64K	64K × 16	2[‡]	1	–	100 PQFP	650	0/70
’BC51PZ–80	80	25	40	2K	8K	–	64K	64K	64K × 16	2[‡]	1	–	100 TQFP[§]	750	0/70
’BC51PQ–57	57	35	28.57	2K	8K/BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	590	0/70
’BC51PQ–80	80	25	40	2K	8K/BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	825	0/70
’BC51PQ–100	100	20	50	2K	8K/BL	–	64K	64K	64K × 16	2[‡]	1	–	132 PQFP	975	0/70

[†] See Figure 2–2 for explanation of device nomenclature.

[‡] One TDM serial port; one standard serial port

Table 2–5. TMS320C5x Commercial Devices (Continued)

				Memory (Words)						Peripherals					
		Cycle		On-Chip			Off-Chip							Typ	Temp
	Freq	Time								Serial			Package	Diss	Range
Device Name†	(MHz)	(ns)	MIPS	RAM	ROM	OTP	Data	Prog	I/O	Port	Timers	HPI	Type	(mW)	(°C)
’BC51PQA–57	57	35	28.57	2K	8K/ BL	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	590	–40/85
’LBC51PQ–57	57	35	28.57	2K	8K/ BL	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	190	0/70
’C51PZ–80	80	25	40	2K	8K	–	64K	64K	64K × 16	21	1	–	100 TQFP§	825	0/70
’BC51PZ–57	57	35	28.57	2K	8K	–	64K	64K	64K × 16	21	1	–	100 TQFP§	590	0/70
’BC51PZ–80	80	25	40	2k	8K	–	64K	64K	64K × 16	21	1	–	100 TQFP§	825	0/70
’BC51PZ–100	100	20	50	2K	8K/ BL	–	64K	64K	64K × 16	21	1	–	100 TQFP§	975	0/70
’LBC51PZ–57	57	35	28.57	2K	8K/ BL	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	190	0/70
’C52PJ–80	80	25	40	1K	4K	–	64K	64K	64K × 16	1¶	1	–	100 PQFP	825	0/70
’BC52PJ–57	57	35	28.57	1K	4K/ BL	–	64K	64K	64K × 16	1¶	1	–	100 PQFP	590	0/70
’BC52PJ–80	80	25	40	1K	4K/ BL	–	64K	64K	64K × 16	1¶	1	–	100 PQFP	835	0/70
’BC52PJA–57	57	35	28.57	1K	4K/ BL	–	64K	64k	64K × 16	1¶	1	–	100 PQFP	590	–40/85
’LBC52PJ–57	57	35	28.57	1K	4K/ BL	–	64k	64k	64K × 16	1¶	1	–	100 PQFP	195	0/70

† See Figure 2–2 for explanation of device nomenclature.
‡ One TDM serial port; one standard serial port
§ Plastic package
¶ One standard serial port
# Two standard serial ports

Table 2–5. TMS320C5x Commercial Devices (Continued)

				Memory (Words)						Peripherals					
				On-Chip			Off-Chip								
Device Name†	**Freq (MHz)**	**Cycle Time (ns)**	**MIPS**	**RAM**	**ROM**	**OTP**	**Data**	**Prog**	**I/O**	**Serial Port**	**Timers**	**HPI**	**Package Type**	**Typ Diss (mW)**	**Temp Range (°C)**
’BC52PZ–57	57	35	28.57	1K	4K/BL	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	590	0/70
’BC52PZ–80	80	25	40	1K	4K/BL	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	825	0/70
’BC52PZ–100	100	20	50	1K	4K/BL	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	975	0/70
’BC52PZA–57	57	35	28.57	1K	4K/BL	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	590	–40/85
’LC52PZ	40	50	20	1K	4K	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	155	0/70
’LC52PZ–57	57	35	28.57	1K	4K	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	190	0/70
’LC52PZA	40	50	20	1K	4K	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	155	–40/85
’LBC52PZ–57	57	35	28.57	1K	4K/BL	–	64K	64K	64K × 16	1¶	1	–	100 TQFP§	190	0/70
’C53PQA	40	50	20	4K	16K	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	525	–40/85
’BC53PQ–57	57	35	28.57	4K	16K/BL	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	590	0/70
’BC53PQ–80	80	25	40	4K	16K/BL	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	825	0/70
’BC53PQA	40	50	20	4K	16K/BL	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	525	–40/85
’LBC53PQ	40	50	20	4K	16K/BL	–	64K	64K	64K × 16	2‡	1	–	132 PQFP	155	0/70

† See Figure 2–2 for explanation of device nomenclature.
‡ One TDM serial port; one standard serial port
§ Plastic package
¶ One standard serial port
# Two standard serial ports
|| One buffered serial port; one standard serial port

Table 2–5. TMS320C5x Commercial Devices (Continued)

				Memory (Words)						Peripherals					
				On-Chip			Off-Chip								
Device Name[†]	Freq (MHz)	Cycle Time (ns)	MIPS	RAM	ROM	OTP	Data	Prog	I/O	Serial Port	Timers	HPI	Package Type	Typ Diss (mW)	Temp Range (°C)
’BC53SPZ–57	57	35	28.57	4K	16K/ BL	–	64K	64K	64K × 16	2[#]	1	–	100 TQFP	590	0/70
’BC53SPZ–80	80	25	40	4K	16K/ BL	–	64K	64K	64K × 16	2[#]	1	–	100 TQFP	825	0/70
’LBC56PZ–57	57	35	28.57	7K	32K	–	64K	64K	64K × 16	2[‖]	1	–	100 TQFP	190	0/70
’LBC56PZ–80	80	25	40	7K	32K	–	64K	64K	64K × 16	2[‖]	1	–	100 TQFP	305	0/70
’LBC57PBK–57	57	35	28.57	7K	32K	–	64K	64K	64K × 16	2[‖]	1	1	128 TQFP	190	0/70
’LBC57PBK–80	80	25	40	7K	32K	–	64K	64K	64K × 16	2[‖]	1	1	128 TQFP	305	0/70
’BC57SPGE–57	57	35	28.57	7K	BL	–	64K	64K	64K × 16	2[‖]	1	1	144 TQFP	590	0/70
’BC57SPGE–80	80	25	40	7K	BL	–	64K	64K	64K × 16	2[‖]	1	1	144 TQFP	825	0/70

[†] See Figure 2–2 for explanation of device nomenclature.
[‡] One TDM serial port; one standard serial port
[§] Plastic package
[¶] One standard serial port
[#] Two standard serial ports
[‖] One buffered serial port; one standard serial port

Table 2–6. TMS320C54x Commercial Devices

				Memory (Words)					Peripherals					
				On-Chip		Off-Chip								
Device Name[†]	Freq (MHz)	Cycle Time (ns)	MIPS	RAM	ROM	Data	Prog	I/O	Serial Port	HPI	Timers	Package Type	Typ Diss (mW)	Temp Range (°C)
’C541#-40	80	25	40	5K	28K	64K	64K	64K × 16	2	–	1	100 TQFP[‡]	350	–40/85
’C542#-40	80	25	40	10K	2K	64K	64K	64K × 16	2	1	1	128/144 TQFP[‡]	350	–40/85
’LC541#-40	80	25	40	5K	28K	64K	64K	64K × 16	2	–	1	100 TQFP[‡]	80	–40/85
’LC541#-50	100	25	40	5K	28K	64K	64K	64K × 16	2	–	1	100 TQFP[‡]	100	–40/85
’LC541#-66	133	15	40	5K	28K	64K	64K	64K × 16	2	–	1	100 TQFP[‡]	131	–40/85
’LC542#-40	80	25	40	10K	2K	64K	64K	64K × 16	2[§]	1	1	128/144 TQFP[‡]	150	–40/85
’LC542#-50	100	20	50	10K	2K	64K	64K	64K × 16	2[§]	1	1	128/144 TQFP[‡]	175	–40/85
’LC543#-40	80	25	40	10K	2K	64K	64K	64K × 16	2[§]	–	1	100 TQFP[‡]	150	–40/85
’LC543#-50	100	20	50	10K	2K	64K	64K	64K × 16	2[§]	–	1	100 TQFP[‡]	175	–40/85
’LC545#-40	80	25	40	6K	48K	64K	64K	64K × 16	2[¶]	1	1	128 TQFP[‡]	80	–40/85
’LC545#-50	100	20	50	6K	48K	64K	64K	64K × 16	2[¶]	1	1	128 TQFP[‡]	100	–40/85
’LC546#-40	80	25	40	6K	48K	64K	64K	64K × 16	2[¶]	–	1	100 TQFP[‡]	80	–40/85
’LC546#-50	100	20	50	6K	48K	64K	64K	64K × 16	2[¶]	–	1	100 TQFP[‡]	100	–40/85
’LC548#-66	133	15	66	32K	2K	64K	8M	64K × 16	3[#]	1	1	144 TQFP[‡]	131	–40/85
’LC548#-80	160	12.5	80	32K	2K	64K	8M	64K × 16	3[#]	1	1	144 TQFP[‡]	158	–40/85
’LC549#-80	160	12.5/15	80	32K	16K	64K	8M	64K × 16	3#	1	1	144 TQFP/BGA	158	–40/85
’VC549#-100	200	10	100	32K	16K	64K	8M	64K × 16	3#	1	1	144 TQFP/BGA	113	–40/85

[†] See Figure 2–2 for explanation of device nomenclature.

[‡] Plastic package

[§] One buffered serial port; one TDM serial port

[¶] One buffered serial port; one standard serial port

[#] Two BSPs; one TDM serial port

Table 2–7. TMS320C6x Commercial Devices

					Memory (Words)		Peripherals					
					On-Chip	Off-Chip						
Device Name†	Freq (MHz)	Cycle Time (ns)	MIPS	Data type	RAM	Parallel	Serial Port	Timers	DMA channels	Package Type	Typ Diss (mW)	Temp Range (°C)
’C6201	200	5	1600	fixed	128K‡	8/16/32	2§	2	4	352 BGA		0/85
’C6701	167	6	1336	floating	128K‡	8/16/32	2§	2	4	352 BGA		0/85

† See Figure 2–2 for explanation of device nomenclature.
‡ 64Kbytes program/cache; 64Kbytes data
§ Multichannel buffered serial ports

Table 2–8. TMS320C8x Commercial Devices

				Memory (Words)					Peripherals					
				On-Chip		Off-Chip								
Device Name†	Freq (MHz)	Cycle Time (ns)	BOPS	SRAM	ROM	Cache	Prog	I/O	Serial Port	HPI	Timers	Package Type	Typ Diss (mW)	Temp Range (°C)
’C80GF–60	60	16	2.5	50K			4G x 8	2	1		4	305 PGA	5	0/85
’C80GF–50	50	20	2.5	50K			4G x 8	2	1		4	305 PGA	5	0/85
’C80GGP–50	50	20	2.5	50K			4G x 8	2	1		4	352 BGA	5	0/85
’C80GGP–60	50	20	2.5	50K			4G x 8	2	1		4	352 BGA	5	0/85
’C82GGP	50	20	1.5	44K			4G x 8	–	1		2	352 BGA	3	0/85
’C82GGP	50	20	1.5	44K			4G x 8	–	1		2	352 BGA	3	0/85

Table 2–9. TMS320AVxxx Commercial Devices

Device Name[†]	Function	Input Format	Output Format	Controller Interface	Off-Chip Memory	Package Type
’AV110	MPEG Audio Decoder	MPEG Audio Stream or MPEG System Stream	16- or 18-bit serial PCM	8-bit	Optional 1M DRAM	120-pin PQFP
’AV120	MPEG Audio Decoder	MPEG Audio Stream Serial PCM	16- or 18-bit	None	None	44-pin PLCC
’AV411/410	Digital NTSC Encoder	RGB, YCbCr	RGB, Y/C (S-Video) or composite video	9-bit		100-pin PQFP

[†] See Figure 2–2 for explanation of device nomenclature.

Table 2–10. TMS320 DSP Military Part Numbers

Device Name	Operating Frequency	Package Type	Typical[†] Dissipation	Temp Range, °C
SMJ320C10JDM	20.5 MHz	Ceramic 40-pin DIP	165 mW	–55/125
SMJ320C10JDM25	25.6 MHz	Ceramic 40-pin DIP	200 mW	–55/125
SMJ320C15JDM	20.5 MHz	Ceramic 40-pin DIP	165 mW	–55/125
SMJ320C15JDM25	25.6 MHz	Ceramic 40-pin DIP	200 mW	–55/125
SMJ320C15FJM	20.5 MHz	44-pin JLCC	165 mW	–55/125
SMJ320C15FJM25	25.6 MHz	44-pin JLCC	200 mW	–55/125
SMJ320E14GBM	20.5 MHz	68-pin PGA	325 mW	–55/125
SMJ320E14FJM	20.5 MHz	68-pin JLCC	325 mW	–55/125
SMJ320C25GBM	40 MHz	68-pin PGA	550 mW	–55/125
SMJ320C25GBM50	50 MHz	68-pin PGA	700 mW	–55/125
SMJ320C25FJM	40 MHz	68-pin JLCC	550 mW	–55/125
SMJ320C25FJM50	50 MHz	68-pin JLCC	700 mW	–55/125
SMJ320C25FDM	40 MHz	68-pin LCCC	550 mW	–55/125
SMJ320C26BGBM	40 MHz	68-pin PGA	550 mW	–55/125
SMJ320C26BFDM	40 MHz	68-pin JLCC	550 mW	–55/125
SMJ320C30GBM33	33.3 MHz	181-pin PGA	1100 mW	–55/125
SMJ320C30HFGM33	33.3 MHz	196-pin CQFP	1100 mW	–55/125
SMJ320C30GBM40	40 MHz	181-pin PGA	1250 mW	–55/125
SMJ320C30HFGM40	40 MHz	196-pin CQFP	1250 mW	–55/125
SMJ320C30TAM33	33.3 MHz	203 OLB TAB	1100 mW	–55/125
SMJ320C30TBM33	33.3 MHz	203 OLB TAB	1100 mW	–55/125
SMJ320C30TAM40	40 MHz	203 OLB TAB	1250 mW	–55/125
SMJ320C30TBM40	40 MHz	203 OLB TAB	1250 mW	–55/125
TMP320C30TAL40	40 MHz	203 OLB TAB	1250 mW	0/70
SMJ320C30KGDM33	33.3 MHz	203-pad KGD	1100 mW	–55/125
SMJ320C30KGDM40	40 MHz	203-pad KGD	1250 mW	–55/125

[†] Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

Table 2–10. TMS320 DSP Military Part Numbers (Continued)

Device Name	Operating Frequency	Package Type	Typical[†] Dissipation	Temp Range, °C
TMP320C30KGDL40	40 MHz	203-pad KGD	1250 mW	0/70
SMJ320C31GFAM33	33.3 MHz	141-pin PGA	750 mW	–55/125
SMJ320C31HFGM33	33.3 MHz	132-pin CQFP	750 mW	–55/125
SMJ320C31GFAM40	40 MHz	141-pin PGA	1250 mW	–55/125
SMJ320C31HFGM40	40 MHz	132-pin CQFP	1250 mW	–55/125
SMJ320C31GFAM50	50 MHz	141-pin PGA	1750 mW	–55/125
SMJ320C31HFGM50	50 MHz	132-pin CQFP	1750 mW	–55/125
SMJ320C31TAM33	33.3 MHz	132 OLB TAB	750 mW	–55/125
SMJ320C31TBM33	33.3 MHz	132 OLB TAB	750 mW	–55/125
SMJ320C31TAM40	40 MHz	132 OLB TAB	1250 mW	–55/125
SMJ320C31TBM40	40 MHz	132 OLB TAB	1250 mW	–55/125
SMJ320C31TAM50	50 MHz	132 OLB TAB	1750 mW	–55/125
SMJ320C31TBM50	50 MHz	132 OLB TAB	1750 mW	–55/125
TMP320C31TAL50	50 MHz	132 OLB TAB	1750 mW	0/70
SMJ320C31KGDM33	33.3 MHz	132-pad KGD	750 mW	–55/125
SMJ320C31KGDM40	40 MHz	132-pad KGD	1250 mW	–55/125
SMJ320C31KGDM50	50 MHz	132-pad KGD	1750 mW	–55/125
TMP320C31KGDL50	50 MHz	132-pad KGD	1750 mW	0/70
SMQ320C32PCMM50	50 MHz	144-pin PQFP	1000 mW	–55/125
SMJ320C40GFM33	33.3 MHz	325-pin PGA	1000 mW	–55/125
SMJ320C40HFHM33	33.3 MHz	352-pin CQFP	1000 mW	–55/125
SMJ320C40GFM40	40 MHz	325-pin PGA	1750 mW	–55/125
SMJ320C40HFHM40	40 MHz	352-pin CQFP	1750 mW	–55/125
SMJ320C40GFM50	50 MHz	325-pin PGA	2500 mW	–55/125
SMJ320C40HFHM50	50 MHz	352-pin CQFP	2500 mW	–55/125
SMJ320C40TABM40	40 MHz	325 OLB TAB	1750 mW	–55/125

[†] Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

Table 2–10. TMS320 DSP Military Part Numbers (Continued)

Device Name	Operating Frequency	Package Type	Typical[†] Dissipation	Temp Range, °C
SMJ320C40TBBM40	40 MHz	325 OLB TAB	1750 mW	–55/125
SMJ320C40TABM50	50 MHz	325 OLB TAB	2500 mW	–55/125
SMJ320C40TBBM50	50 MHz	325 OLB TAB	2500 mW	–55/125
TMP320C40TABL50	50 MHz	325 OLB TAB	2500 mW	0/70
TMP320C40TABL60	60 MHz	325 OLB TAB	3250 mW	0/70
SMJ320C40KGDM40	40 MHz	325-pad KGD	1750 mW	–55/125
SMJ320C40KGDM50	50 MHz	325-pad KGD	2500 mW	–55/125
TMP320C40KGDL50	50 MHz	325-pad KGD	2500 mW	0/70
TMP320C40KGDL60	60 MHz	325-pad KGD	3250 mW	0/70
TMP320C50KGDL40	40 MHz	117-pad KGD	525 mW	0/70
TMP320C50KGDL57	57 MHz	117-pad KGD	590 mW	0/70
SMJ320C50KGDM50	50 MHz	117-pad KGD	450 mW	–55/125
SMJ320C50KGDM66	66 MHz	117-pad KGD	600 mW	–55/125
SMJ320C50GFAM50	50 MHz	141-pin PGA	450 mW	–55/125
SMJ320C50GFAM66	66 MHz	141-pin PGA	600 mW	–55/125
SMJ320C50HFGM50	50 MHz	132-pin CQFP	450 mW	–55/125
SMJ320C50HFGM66	66 MHz	132-pin CQFP	600 mW	–55/125
SMQ320C50PQM66	66 MHz	132-pin PQFP	600 mW	–55/125
TMP320BC51KGDL40	40 MHz	117-pad KGD	525 mW	0/70
TMP320BC51KGDL57	57 MHz	117-pad KGD	590 mW	0/70
SMJ320C80GFM40	40 MHz	305-pin CPGA	3 W	–55/125
SMJ320C80GFM50	50 MHz	320-pin CPGA	TBD	–55/125
SMJ320C80HFHM40	40 MHz	305-pin CQFP	3 W	–55/125
SMJ320C80HFHM50	50 MHz	320-pin CQFP	TBD	–55/125

[†] Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

2.4 TMS320 Development and Support Tools Overview

Fast time to market, increased productivity, and ease of design-in are of primary importance when developing a DSP-based system. To meet these needs, TI offers a complete suite of code-generation and debug tools as well as a broad and innovative range of third-party tools (See Chapter 16). TMS320 DSP tool support eases the design process from initial concept through integration and production, enabling customers to take full advantage of rapidly evolving DSP technologies. Table 2–11 shows an overview of the TMS320 development and support tools. Table 2–12 summarizes the features of the simulation and emulation development tools.

Table 2–11. TMS320 Development and Support Tools Overview

’C2xx	**’C3x**	**’C4x**	**’C5x**	**’C54x**	**’C6x**	**’C8x**
	EVM DSK	Parallel Processing Development System (PPDS)	EVM DSK	EVM DSKplus	EVM	
Simulator Software With Debugger	Simulator Software With Debugger	Simulator Software With Debugger	Simulator Software With Debugger	Simulator Software With Debugger	Simulator Software With Debugger	Simulator Software With Debugger, Profiler
C Compiler/ Assembler/ Linker	C Compiler Assembler/ Linker	C Compiler Assembler/ Linker	C Compiler Assembler/ Linker	C Compiler Assembler/ Linker	C Compiler/ Assembler/ Linker/ Assembly Optimizer	C Compiler/ Assembler/ Linker
XDS510/ XDS510WS Hardware	XDS510/ XDS510WS Hardware	XDS510/ XDS510WS Hardware	XDS510/ XDS510WS Hardware	XDS510/ XDS510WS Hardware	XDS510/ XDS510WS Hardware	XDS510/ XDS510WS Hardware
Conversion Cable	Conversion Cable	Conversion Cable	Conversion Cable	Conversion Cable	Conversion Cable	Conversion Cable
Emulator Porting Kit	Emulator Porting Kit	Emulator Porting Kit	Emulator Porting Kit	Emulator Porting Kit	Emulator Porting Kit	Emulator Porting Kit
Code Composer	Code Composer	Code Composer	Code Composer	Code Composer	Code Composer	

† Evaluation module

‡ DSP starter kit

Table 2–12. Features of TMS320 Simulation/Emulation Development Tools

Features	**EVM, SDB or PPDS[†]**	**Simulator**	**XDS510**
TMS320 device supported:	’C30 ’C50 ’C54x ’C6x ’C8x	’C2xx ’C3x ’C4x ’C5x ’C54x ’C6x ’C8x	’C2xx ’C3x ’C4x ’C5x ’C54x ’C6x ’C8x
Development purpose:			
Evaluation/benchmarking	Yes	Yes	Yes
Software design	Yes	Yes	Yes
Hardware design	No	No	Yes
Line-by-line or reverse assembler	Yes	Yes	Yes
Modify/display memory and registers	Yes	Yes	Yes
Single-stepping	Yes	Yes	Yes
Breakpoint on instruction acquisition	Yes	Yes	Yes
Breakpoint on memory access/read/write	No	Yes	Yes
Time-stamping/clock counter	No	Yes	No
Real-time trace samples	No	No	No
Multiuser system	No	Yes	Yes
HLL user interface	Yes	Yes	Yes
Files associated with I/O ports	No	Yes[‡]	No
Full-speed in-circuit emulation:			
From on-board memory	Yes	N/A	No
From target memory	No	N/A	Yes
Amount of external memory (words):			
On-board program/data (’C3x)	16K	N/A	N/A
On-board program/data (’C4x)	384K	N/A	N/A
On-board program/data (’C5x)	64K	N/A	N/A
On-board program/data (’C54x)	64K	N/A	N/A
’C6x		N/A	N/A
Program/data expansion	N/A	N/A	N/A

[†] EVM = evaluation module, SDB = software development board, PPDS = parallel processing development system

[‡] The memory expansion board (included in the ’C2x XDS/22) allows for memory expansion to 64K total words of program and data memory, configurable in 1K-word blocks.

[§] Program/data expansion depends on the user’s target system.

2.4.1 TMS320 Tool Numbering System

To classify the stages in the support tool development cycle, Texas Instruments assigns prefix designators in the part number nomenclature. There are three development support tool prefix designators — TMDX, TMDS, and TMDC — that represent the evolutionary stages of tool development from engineering prototypes (TMDX) through fully qualified production units (TMDS). The development flow is defined as follows:

Support Tool Development Evolutionary Flow

TMDX	Developmental product that has not yet completed TI internal qualification testing
TMDS	Fully qualified development support product
TMDC	Development support product that is unsupported or obsolete

TMX and TMP devices and TMDX development support tools are shipped with the following disclaimer:

Developmental product intended for internal evaluation purposes.

Note:

Texas Instruments recommends that prototype devices (TMX or TMP) not be used in production systems, because their expected end-use failure rate is undefined but is predicted to be greater than the failure rate of standard, qualified production devices.

TMDS development support tools have been properly tested, and the quality and reliability of the devices have been successfully demonstrated. TI's standard warranty applies.

2.4.2 Development and Support Tools

The standard flow for development begins with code written in a high-level language (such as C) or assembly language. This code can be debugged on one of several debug platforms (simulator, software development board, emulator, etc.), all of which feature the same debugger interface. Software and hardware can be developed in parallel by using the TMS320 compiler, assembler/linker, simulator, evaluation module (EVM), or software development board (SDB) for software development, and by using the TMS320 emulators for hardware development. Table 2–13 gives part numbers and host/operating system requirements for the code development and debug support functions.

Table 2–13. TMS320 Code Development Support Tools

Device	Product Description	Part Number	Host	Operating System[†]
’C2xx	Assembler/Linker	TMDS3242850–02	PC	OS/2 DOS Ext
	C Compiler Assembler/Linker	TMDS3242855–02	PC	OS/2 DOS Ext
	C Compiler Assembler/Linker	TMDS3242555–08	SPARC HP9000	N/A
’C3x	Assembler/Linker	TMDS3243850–02	PC	OS/2 DOS Ext Win32
	C Compiler Assembler/Linker	TMDS3243855–02	PC	OS/2 DOS Ext Win32
	C Compiler Assembler/Linker	TMDS3243555–08	SPARC HP9000	N/A

[†] PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System[†]
’C4x	Assembler/Linker	TMDS3243850–02	PC	OS/2 DOS Ext Win32
	C Compiler Assembler/Linker	TMDS3243855–02	PC	OS/2 DOS Ext Win32
	C Compiler Assembler/Linker	TMDS3243555–08	SPARC HP9000	N/A
’C5x	Assembler/Linker	TMDS3242850–02	PC	OS/2 DOS Ext
	C Compiler Assembler/Linker	TMDS3242855–02	PC	OS/2 DOS Ext
	C Compiler Assembler/Linker	TMDS3245555–08	SPARC HP9000	N/A
’C54x	Assembler/Linker	TMDS324L850–02	PC	OS/2 DOS Ext
	C Compiler Assembler/Linker	TMDS324L855–02	PC	OS/2 DOS Ext
	C Compiler Assembler/Linker	TMDS324L555–09	SPARC	N/A
’C6x	C Compiler Assembler/Linker	TMDX3246885–07	PC	Win32
	Compiler Assembler/Linker	TMDX3246555–07	SPARC	

[†] PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System[†]
’C8x	Code Generation Software Toolkit (includes Compiler, Assembler, Linker, and Simulator Debugger)	TMDS3248555–67	SPARC	N/A
	Code Generation Software Toolkit (includes Compiler, Assembler, and Linker)	TMDS3248855–07	PC	Windows NT

[†] PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System[†]
’C2xx	Simulator	TMDX324X851–02	PC	DOS Win16
	Simulator	TMDX324X551–09	SPARC	OpenWin
	XDS510 Debugger	TMDX324012XX	PC	DOS Win16 OS/2
	XDS510WS Debugger	TMDX324062XX	SPARC	OpenWin
	XDS510 Board JTAG Emulator Cable	TMDS00510	PC (ISA)	N/A
	XDS510WS Box JTAG Emulator Cable	TMDS00510WS	SPARC	OpenWin
	JTAG Emulator Cable	TMDS3080002	N/A	N/A
	Emulator Porting Kit[‡]	TMDX324002XX	PC	DOS Win16 OS/2

[†] PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System†
’C3x	Simulator	TMDS3243851–02	PC	DOS Win16
	Simulator	TMDS3243551–09	SPARC	OpenWin
	XDS510 Debugger	TMDS3240130	PC	DOS Win16
	XDS510WS Debugger	TMDS3240630	SPARC	OpenWin
	XDS510 Board MPSD Cable	TMDS00510M	PC (ISA)	N/A
	XDS510WS Controller MPSD Cable Power Supply SCSI Cable	TMDS00510WSM	SPARC	N/A
	MPSD Cable	TMDS3080004	PC SPARC	N/A
	Emulator Porting Kit‡	TMDS3240030	PC	DOS Win16
	’C30 EVM Board Debugger Assembler/Linker	TMDS3260030	PC(ISA)	DOS Win16
	’C31 DSK Assembler/Debugger	TMDS3200031	PC(DB25)	DOS

† PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System†
’C4x	Simulator	TMDS3244851–02	PC	DOS Win16
	Simulator	TMDS3244551–09	SPARC	OpenWin
	XDS510 Debugger	TMDS3240140	PC	OS/2 DOS Win16
	XDS510WS Debugger	TMDS3240640	SPARC	OpenWin
	XDS510 Board JTAG Cable	TMDS00510	PC (ISA)	N/A
	XDS510WS Controller JTAG Cable Power Supply SCSI Cable	TMDS00510WS	SPARC	N/A
	JTAG Cable	TMDS3080002	PC SPARC	N/A
	Emulator Porting Kit‡	TMDX3240040	PC	OS/2 DOS Win16
	C40 PPDS Board	TMDX3261040	PC XDS510 SPARC XDS510WS	N/A

† PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System[†]
’C5x	Simulator	TMDS3245851–02	PC	DOS Win16
	Simulator	TMDS3245551–09	SPARC	OpenWin
	XDS510 Debugger	TMDS3240150	PC	DOS Win16 OS/2
	XDS510WS Debugger	TMDS3240650	SPARC	OpenWin
	XDS510 Board JTAG Emulator Cable	TMDS00510	PC (ISA)	N/A
	XDS510WS Box JTAG Emulator Cable	TMDS00510WS	SPARC	OpenWin
	JTAG Emulator Cable	TMDS3080002	N/A	N/A
	Emulator Porting Kit[‡]	TMDX3240050	PC	DOS Win16 OS/2
	’C50 EVM Board Debugger	TMDS3260050	PC (ISA)	DOS Win16
	’C50 DSK Assembler/Debugger	TMDS3200051	PC (UART)	DOS

[†] PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System†
’C54x	Simulator	TMDS324L851–02	PC	DOS Win16
	Simulator	TMDS324L551–09	SPARC	OpenWin
	XDS510 Debugger	TMDS32401L0	PC	DOS Win16 OS/2
	XDS510WS Debugger	TMDS32406L0	SPARC	OpenWin
	XDS510 Board JTAG Emulator Cable	TMDS00510	PC (ISA)	N/A
	XDS510WS Box JTAG Emulator Cable	TMDS00510WS	SPARC	OpenWin
	JTAG Emulator Cable	TMDS3080002	N/A	N/A
	Emulator Porting Kit‡	TMDX32400L0	PC	DOS Win16 OS/2
	EVM Board Debugger	TMDS3260051	PC (ISA)	DOS Win16
	’C54x DSKplus Assembler/Debugger	TMDS32000L0	PC (UART)	DOS

† PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Table 2–13. TMS320 Code Development Support Tools (Continued)

Device	Product Description	Part Number	Host	Operating System†
’C6x	Simulator	TMDX324551–07	SPARC	N/A
	Simulator	TMDX3246851–07	PC	DOS, OS/2 Win32
	XDS510 Debugger	TMDX3240160–07	PC	DOS, OS/2 Win32
	XDS510 Emulator Hardware JTAG Cable	TMDS00510	PC	DOS, OS/2 Win32
	TMS320C6201 Test and Evaluation Board	TMD326106201		
	TMS320C62x Test and Evaluation Board	TMD326006201	PC	Win32
’C8x	Debugger Software for Emulator	TMDS3240680	SPARC	N/A
	Debugger Software for Emulator	TMDS3240180	PC	Windows NT
	Emulator Card (includes XDS510 Board and JTAG Emulator Cable)	TMDS00510	PC (ISA)	N/A
	Emulator Box (includes XDS510WS Box and JTAG Emulator Cable)	TMDS00510WS	SPARC	N/A
	JTAG Emulator Cable	TMDS3080002	–	–
	Emulator Porting Kit‡	TMDX3240080	PC SPARC	–

† PC = MS-DOS 5.x or higher, OS/2.2x or higher
DOS Ext = DOS/4GW DOS Extender
Win16 = Windows 3.xx
Win32 = Windows NT and Windows 95
SPARC = SunOS 4.1.3 or higher or Solaris 2.x
OpenWin = Open Windows 3.x
HP9000 series 700 = HP-UX 9.x or higher
N/A = Not applicable

Chapter 3

TMS320C20x Devices

The TMS320C1x generation of digital signal processors was the first series of DSPs developed by Texas Instruments. Its first member, the TMS320C10, was introduced in 1982. The ’C2x expanded the ’C1x architecture (16-bit fixed-point Harvard architecture).

Texas Instruments further enhanced the DSP market with the powerful, cost-effective TMS320C20x devices. These DSPs feature a 16-bit fixed-point, Harvard architecture family that offers performance up to 40 MIPS and power dissipation of as little as 1.1 mA/MIPS.

Topic **Page**

3.1 TMS320C20x Introduction

The ’C20x devices are fully static, are offered in 5-V and 3-V versions, have a full CPU shutdown idle mode, and can use on-chip phase-locked loop (PLL) options. These features all contribute to reductions in system power requirements.

The ’C20x devices are source-code compatible with the ’C2x generation and are upwardly compatible with the ’C5x/’C54x generation. The ’C5x/’C54x assembler accepts ’C2xx instructions as well as ’C5x/’C54x instructions. The ’C2xx instruction set incorporates all the ’C2x instructions and adds 11 new ones.

The core CPU of the ’C20x features 544 words of dual-access RAM. The external address reach of the ’C20x is 224K words; 64K words of data, program, and I/O, and 32K words of global memory for multiprocessor operation.

The ’C20x generation is supported by a full suite of development tools that make designing with these devices easy. Texas Instruments offers a C compiler/assembler/linker, simulator, evaluation module, and an emulator/debugger that supports nonintrusive IEEE Standard 1149.1-compliant (JTAG) scan-based emulation. Numerous third-party companies also provide development tools for the ’C20x generation.

Table 3–1. ’C20x Clock Speed and MIPS

Clock Speed (MHz)	Instruction Cycle Time (ns)	MIPS
40	50	20
57	35	28.5
80	25	40

The ’C20x generation also features the first TI DSPs with on-chip flash memory. The ’F206 has 32K words of flash memory and 4.5K words of RAM, enabling designers to eliminate costly off-chip memory.

3.2 TMS320C20x Key Features

Following are the key features of the TMS320C20x generation. When only certain

- Up to 4.5K data/program RAM on chip
- Up to 32K words of flash memory on-chip
- 32-bit ALU/accumulator
- 16 × 16-bit parallel multiplier with a 32-bit product
- Repeat instructions for efficient use of program space and enhanced execution
- 16-bit on-chip timer
- 16-bit barrel shifter
- 8-level hardware stack
- Built-in power-down mode
- Software wait-state generator
- 80- or 100-pin TQFP packages
- Various PLL options for reduced electromagnetic interference (EMI) and system power dissipation
- ’C203, ’LC203, ’C206, ’LC206, and ’F206 are pin-for-pin compatible in a 100-pin TQFP package.

3.3 TMS320C20x Enhanced Synchronous Serial Port

The ’C20x offers a full-duplex framed enhanced synchronous serial port (ESSP) with up to 20-Mbps throughput (at a 25-ns instruction cycle time). The transfer rate is one-half the device clockout rate. This bidirectional synchronous serial port provides direct communication with serial devices such as codecs, serial ADCs, and other serial systems. The serial port can be used for intercommunication between processors in multiprocessing applications.

Both the receive and transmit sides of the serial port have a 4-level-deep buffer or FIFO, allowing the CPU to accept an interrupt at either 1, 2, 3, or 4 levels deep. This capability means less intervention from the CPU, as well as increased flexibility and efficiency with respect to data transfers.

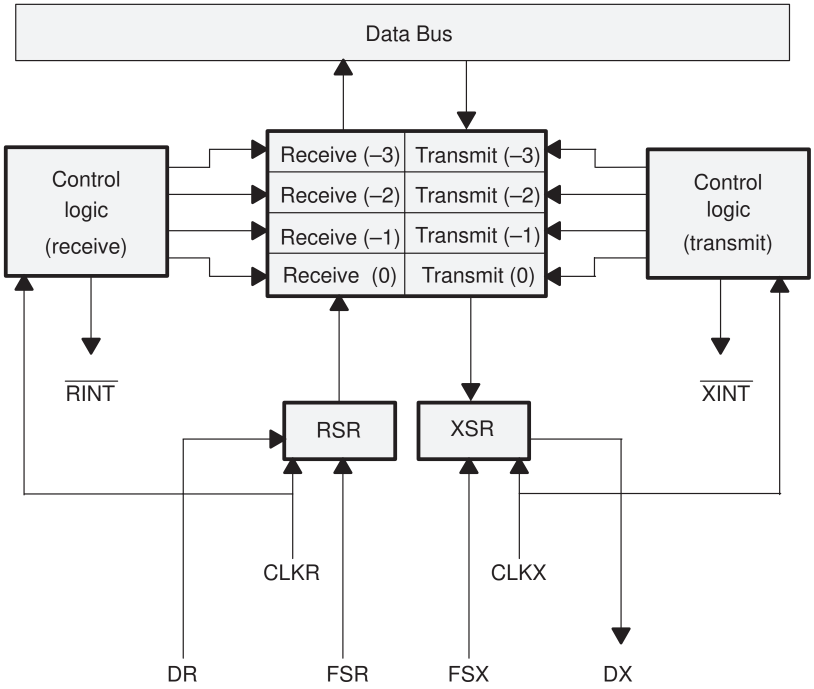
The ’C20x synchronous serial port features:

- Full-duplex framed synchronous serial port
- 4-word × 16-bit buffer to reduce interrupt service routine (ISR) overhead
- Serial port performance
  - 20 Mbps at 25 ns
  - 14.28 Mbps at 35 ns
  - 10 Mbps at 50 ns
- Transfer rate one-half that of the CPU rate

All ’C20x devices, except the ’C209, feature this serial port.

Figure 3–1 shows a block diagram of the enhanced synchronous serial port.

Figure 3–1. TMS320C20x Enhanced Synchronous Serial Port (ESSP)



3.4 TMS320C20x Asynchronous Serial Port

The ’C20x offers an asynchronous serial port that is full-duplex and double-buffered. It accepts 8-bit data and can be programmed through a register to accept baud rates of up to 2.5 Mbps. The asynchronous serial port can be used to communicate with other devices such as microcontrollers or for RS-232 connections that support data transfers of up to 115.2 kbps.

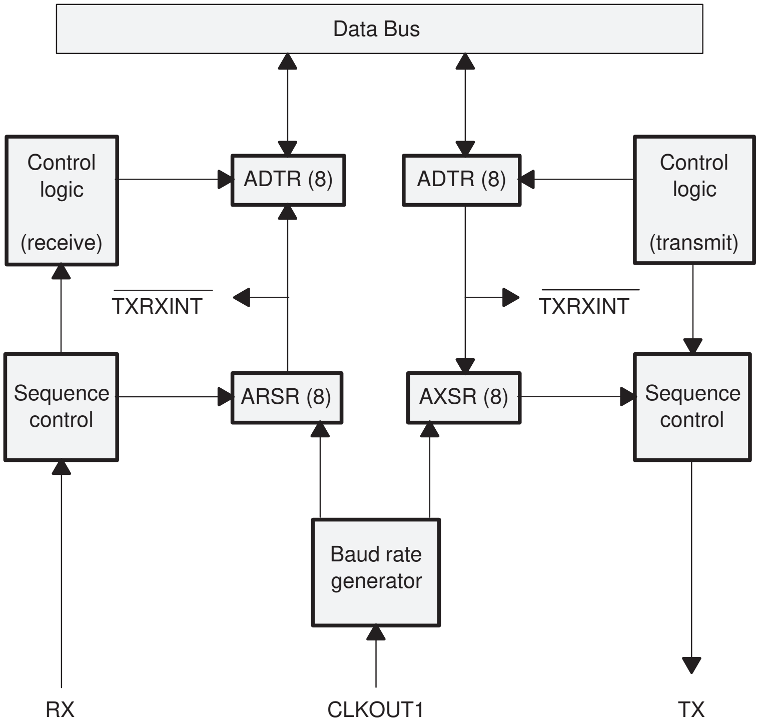
Features included with the ’C20x asynchronous serial port are:

- ❑ Full duplex
- ❑ Double buffered
- ❑ 8-bit data transfers
- ❑ 16-bit register for baud-rate generation
- ❑ Baud rates up to 2.5 Mbps (at a 25-ns instruction cycle time)

All ’C20x devices, except the ’C209, feature this asynchronous serial port.

Figure 3–2 shows a block diagram of the asynchronous serial port.

Figure 3–2. TMS320C2xx Asynchronous Serial Port



3.5 TMS320C20x Boot Loader

Some of the 'C20x devices have an on-chip hard-coded boot loader, which allows you to load code from an 8-bit external EPROM into internal or external RAM. The EPROM is mapped into global data memory. Once the boot loading operation begins, 8-bit data is read by the device and reassembled into 16-bit words to a user-specified destination. When complete, control of the device is passed to the start of the program.

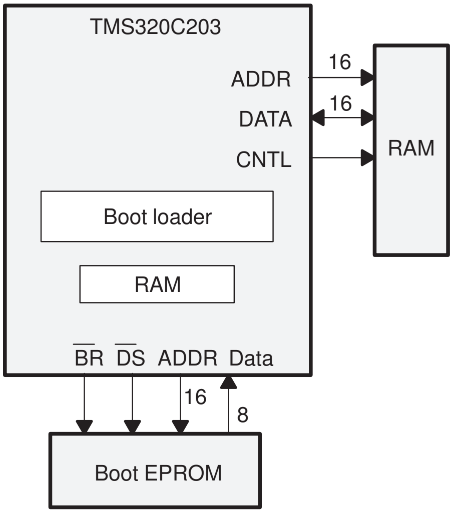
Features of the 'C20x boot loader are:

- Load from 8-bit external EPROM into internal/external RAM
- EPROM is mapped into global data memory space
- Dedicated boot loader pin
- Boot loader operation:
  - Reads 8-bit data
  - Reassembles into 16-bit words
  - Loads to user-specified destination
  - When complete, control passes to start of program

The 'C203 features this boot loader.

Figure 3–3 shows a block diagram of the boot loader.

Figure 3–3. TMS320C2xx Boot Loader



3.6 TMS320C20x Memory Bus

The ’C20x advanced Harvard-type architecture maximizes processing power by maintaining two separate memory bus structures, program and data, for full-speed execution. This multiple bus structure allows reading both data and instructions simultaneously. Instructions support data transfers between the two spaces. This architecture lets you store coefficients in program memory to be read in RAM, eliminating the need for a separate coefficient ROM. This, coupled with a 4-level-deep pipeline, allows the ’C20x to execute most instructions in a single cycle.

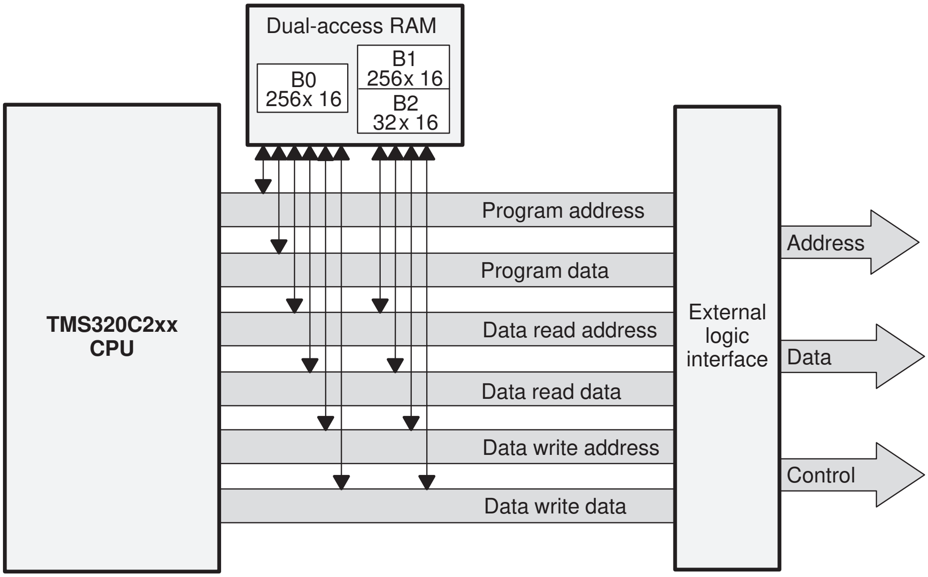
The ’C20x dual-access RAM (DARAM) allows writes to and reads from the RAM in the same cycle without the address restrictions of single-access RAM (SARAM). The DARAM is configured in three blocks: block B0, block B1, and block B2. Block B0 is a 256-word block that can be configured as data or program memory. Block B1 is 256 words in data memory and block B2 is 32 words in data memory.

Some ’C20x devices also have SARAM and/or ROM. Also, the ’F206 includes 32K words of flash memory. Flash memory offers an excellent alternative to RAM because of its lower cost and nonvolatile storage and is an alternative to ROM due to its reprogrammability.

The ability of the DARAM to perform two accesses in one cycle, coupled with the parallel nature of the ’C20x architecture, enables the ’C20x devices to perform three concurrent memory accesses in any given machine cycle.

Figure 3–4 shows a block diagram of the memory bus.

Figure 3–4. TMS320C20x Memory Bus



3.7 TMS320C203

The ’C203 incorporates all the features of the ’C20x core and adds two high-performance serial ports for enhanced communication with peripheral devices. There is one synchronous serial port with a 4-level-deep FIFO, which results in less intervention from the CPU, but at a low cost. One full-duplex asynchronous serial port, a programmable wait-state generator, and a 16-bit timer are also included. The serial ports, timer, and wait-state generator are mapped into I/O space. The ’C203 is packaged in a 100-pin TQFP.

Features of the ’C203 include:

- 5-V version
- 25-, 35-, and 50-ns instruction cycle times
- 544 words RAM
- 192K-word external address reach
- 20–40 million instructions per second (MIPS)
- Accepts source code from the TMS320C1x/’C2x generations
- ANSI C compiler
- +2, ×1, × 2, and × 4 PLL options
- IEEE 1149.1-standard (JTAG) emulator control
- Boot ROM option
- Full-duplex synchronous serial port with 4-level-deep FIFO
- Full-duplex asynchronous serial port (UART)
- 100-pin TQFP package

Figure 3–5 shows a block diagram of the ’C203.

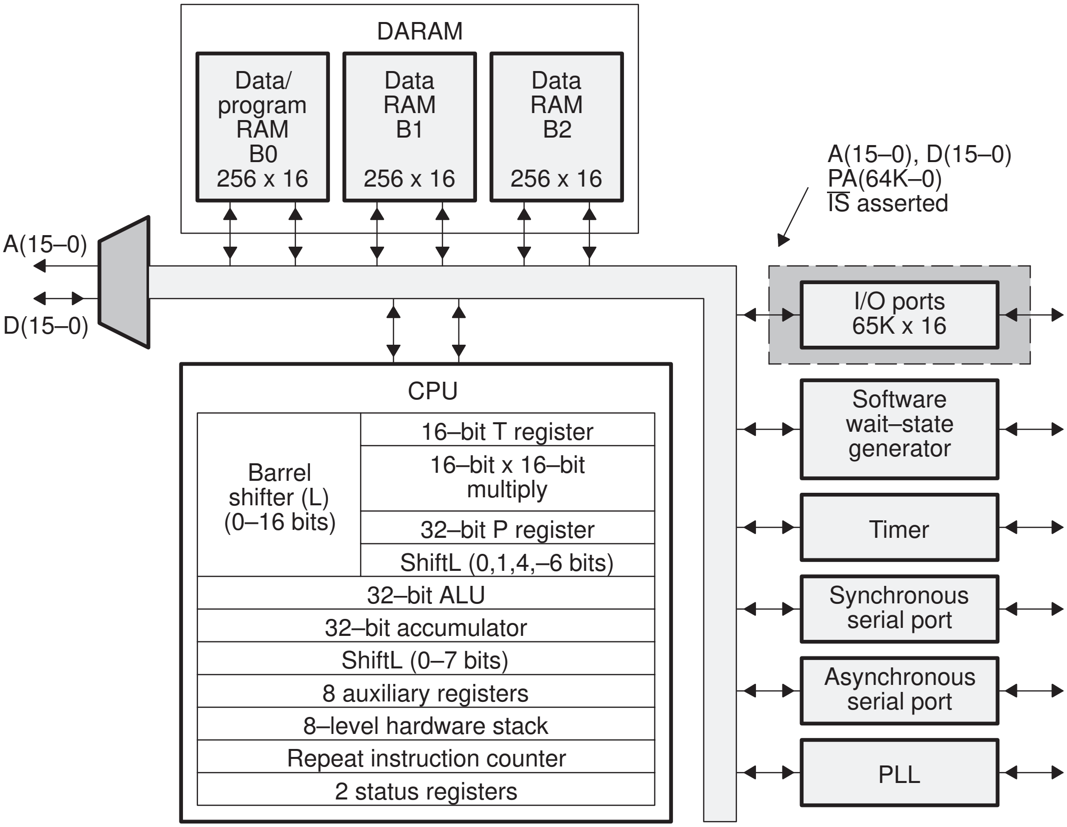
3.8 TMS320LC203

The ’LC203 is a low-power version of the TMS320C203.

Features of the TMS320C203 include:

- ❑ 3.3-V version
- ❑ 50-ns instruction cycle time
- ❑ 544 words RAM
- ❑ 192K word external address reach
- ❑ 20 MIPS
- ❑ ANSI C compiler
- ❑ Boot ROM option
- ❑ Two full duplex-synchronous serial ports with 4-level deep FIFO
- ❑ Two full duplex asynchronous serial ports (UART)
- ❑ 100-pin TQFP package

Figure 3–5. TMS320C203 Block Diagram



The synchronous serial port has full-duplex, framed operation at up to 20 Mbps throughput. The transfer rate is one-half the device clockout rate. Both the receive and transmit sides of the serial port have a first-in first-out (FIFO) buffer, allowing the CPU to accept an interrupt at levels 1, 2, 3, or 4. This capability means less intervention from the CPU, as well as increased flexibility and efficiency for data transfers.

The asynchronous serial port is a universal asynchronous receive/transmit (UART) device. It offers full-duplex and double-buffered operation. It accepts 8-bit data and can be programmed via a register to accept baud rates up to 2.5 Mbps. The asynchronous serial port can be used to communicate with other devices, such as microcontrollers or for RS-232 connections that support data transfers of up to 115.2K bps.

3.9 TMS320C206

The architecture of the ’C206 is based on that of the TMS320C2xx series and is optimized for low-power operation.

Features of the ’C206 include:

- 5-V I/O version (3.3V core)
- 4.5K words RAM
- 32K words ROM
- 192K-word external address reach
- Accepts source code from the ’C1x/’C2x generations
- ANSI C compiler
- ÷ 2, × 1, × 2, and × 4 PLL options
- IEEE 1149.1-standard JTAG emulator control
- Full-duplex enhanced synchronous serial ports (ESSP) with 4-level deep FIFOs
- Full-duplex asynchronous serial ports (UART)
- 100-pin TQFP packaging

Figure 3–7 shows a block diagram of the ’C206.

3.10 TMS320LC206

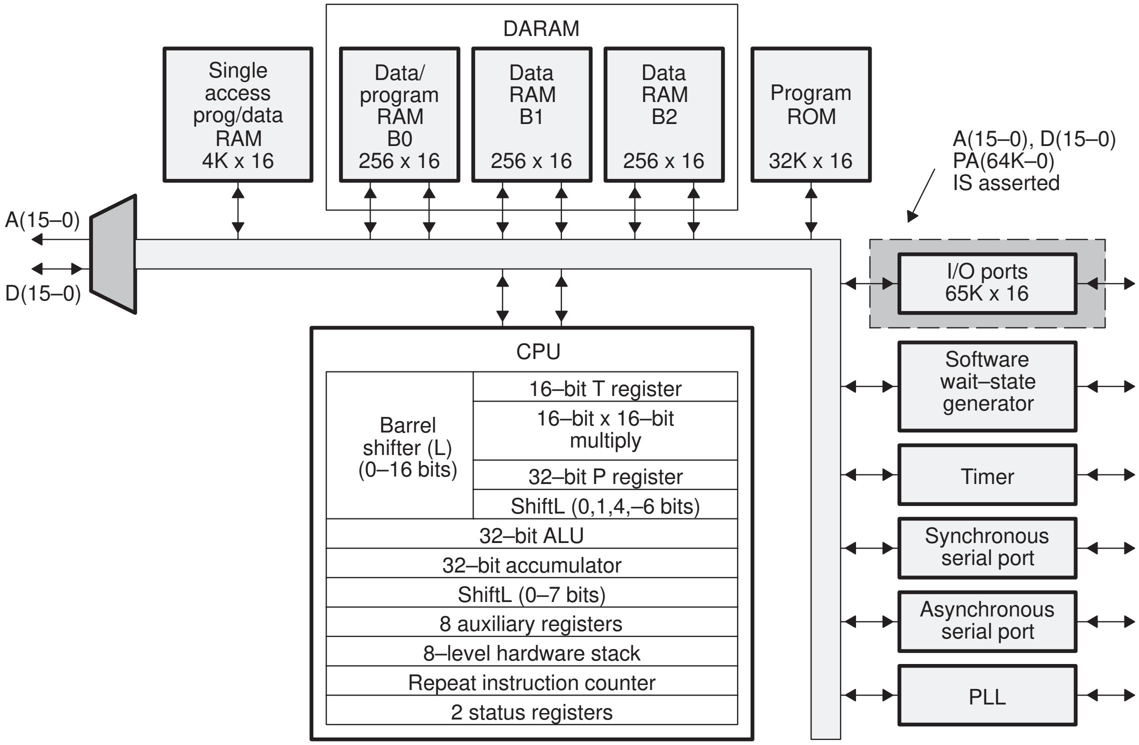
The ’LC206 is a low-power version of the ’C206.

Features of the TMS320LC206 include:

- ❑ 3.3-V version
- ❑ 4.5K words RAM
- ❑ 32K words ROM
- ❑ 192K-word external address reach
- ❑ Accepts source code from the ’C1x/’C2x generations
- ❑ ANSI C compiler
- ❑ ÷ 2, × 1, × 2, and × 4 PLL options
- ❑ IEEE 1149.1-standard JTAG emulator control
- ❑ Full-duplex synchronous serial ports with 4-level deep FIFOs
- ❑ Full-duplex asynchronous serial ports (UART)
- ❑ 100-pin TQFP packaging

Figure 3–7 shows a block diagram of the ’LC206.

Figure 3–6. TMS320C206/'LC206 Block Diagram



3.11 TMS320F206

The ’F206 is the first digital signal processor from Texas Instruments with on-chip flash memory. The ’F206 Flash DSP has 32K words of flash integrated into program memory. Flash memory is attractive for program memory because it has a lower cost than SRAM and more flexibility than ROM. The flash memory on the ’F206 can be programmed through the ’C20x emulator for easy changes in the program during prototyping. The reprogrammability of the flash allows quick changes to the product to adapt to new standards and to add new features to the end equipment.

In addition to the flash, the ’F206 also has a total of 4.5K of on-chip RAM. This level of memory integration allows a chip solution for many systems.

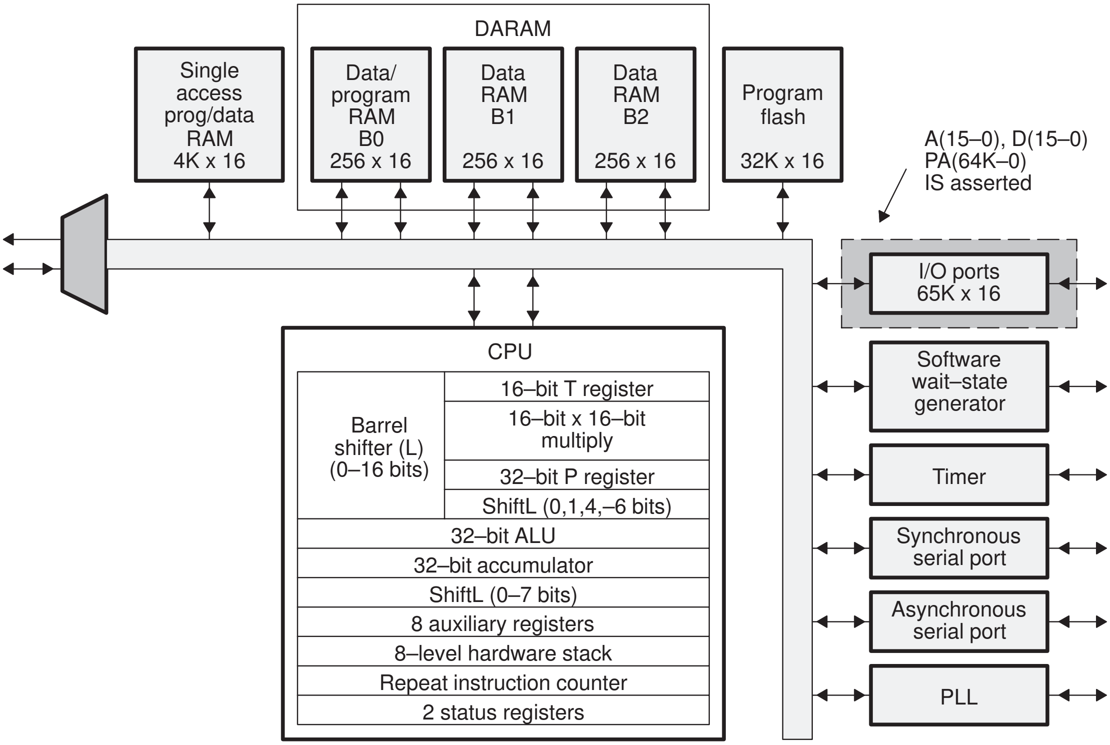
The ’F206 is based on the same ’C20x core as other members of this generation and incorporates the same serial ports as the ’C203. The ’F206 comes in a 100-pin TQFP package and is footprint-compatible with the ’C203, ’C206, and ’LC206.

Features of the ’F206 Flash DSP include:

- ❑ 50-ns instruction cycle times
- ❑ 4.5K words RAM
- ❑ 32K words on-chip flash
- ❑ 192K-word external address reach
- ❑ Accepts source code from the ’C1x/C2x generations
- ❑ ANSI C compiler
- ❑ ÷ 2, × 1, × 2, and × 4 PLL options
- ❑ IEEE 1149.1-standard (JTAG) emulator control
- ❑ Full-duplex synchronous serial ports with 4-level-deep FIFOs
- ❑ Full-duplex asynchronous serial port (UART)
- ❑ 100-pin TQFP package

Figure 3–7 shows a block diagram of the ’F206.

Figure 3–7. TMS320F206 Block Diagram



3.12 TMS320C209

The ’C209 was the first member of the ’C20x generation. It takes the ’C20x core and adds 4K words of ROM and an additional 4K words of RAM (total of 4.5K words). The large on-chip memory, small packaging, and low cost make this device attractive for space-constrained applications such as small form factor hard-disk drives.

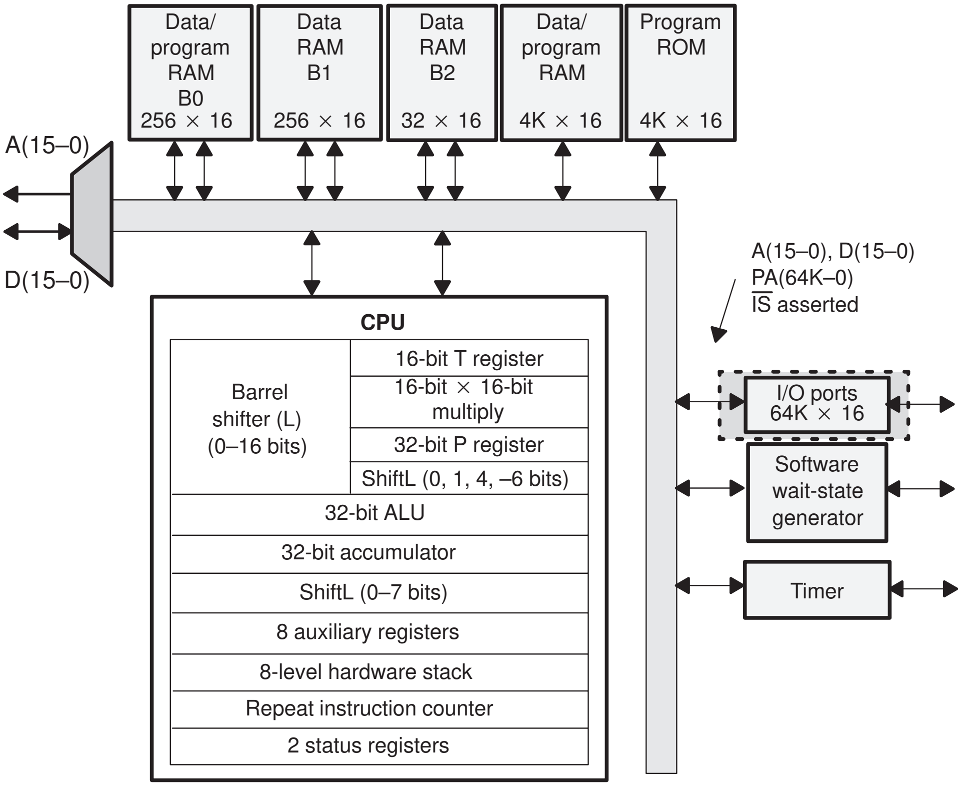
The ’C209 does not include the serial ports of the ’C203. It has performance ratings of 20 MIPS and 28.5 MIPS and comes in an 80-pin TQFP.

The TMS320C209 features:

❑ 35- and 50-ns instruction cycle times
❑ 4K 16-bit words of RAM
❑ 4K words ROM
❑ Accepts source code from the ’C1x/C2x generations
❑ ANSI C compiler
❑ ÷ 2, × 2 PLL option
❑ IEEE 1149.1-standard (JTAG) emulator control
❑ 80-pin TQFP package

Figure 3–8 shows a block diagram of the ’C209.

Figure 3–8. TMS320C209 Block Diagram



3.13 Tools for TMS320C20x

For information about the code generation and debugging tools available for the ’C20x devices, see Chapter 13, *Code Generation Tools* and Chapter 14, *System Integration and Debugging Tools* for the flash utilities. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

Chapter 4

TMS320C24x DSP Controllers

The ’C24x devices are an optimized set of digital motor-control processors. They include event managers specifically designed to allow robust and energy-efficient motor-control designs. These products are well suited for variable-speed control of brushless motors, including dc permanent magnet, ac induction, and switched reluctance motors.

Topic **Page**

4.1 TMS32024x DSP Controllers Introduction

The TMS320C24x DSP controllers are motor-control-optimized members of Texas Instruments TMS320C2xx DSP generation. They have been specifically designed for control system applications, and they combine all of the necessary integration for single-chip controller implementations. This continues the high performance of a DSP core with the on-chip peripherals of a microcontroller, yielding a high-performance DSP. At 50 ns instruction cycle times, the ’C24x offers 10 to 20 times the speed of traditional 16-bit microcontrollers and microprocessors.

The 16-bit, fixed-point core that is the DSP engine for the ’C24x devices provides analog designers a digital solution that does not sacrifice the precision and performance of their systems. In fact, system performance can be enhanced through the use of advanced control algorithms for techniques such as adaptive control, Kalman filtering, and state control. The TMS320 DSP controllers are reliable and programmable.

The high-speed central processing unit (CPU) of the TMS320 DSP controllers allows the digital designer to process algorithms in real time rather than approximate results through look-up tables. The instruction set of these DSP controllers, which incorporates both signal-processing instructions and general-purpose functions is coupled with the extensive development support available for the TMS320 DSP family. This reduces development time and provides the same ease of use as traditional 8- and 16-bit microcontrollers.

The ’C24x DSP controllers leverage a set of peripheral functions that allows TI to configure various family members for different price/performance points or application optimization. This library of both digital and mixed-signal peripherals includes:

❑ Timers
❑ Serial communication ports
❑ Data converters
❑ Event managers

4.2 TMS320C24x Architecture

The architectural design of the ’C24x is based on the ’C2xx DSP core. The operational flexibility and speed of the ’C24x is a result of an advanced, modified Harvard architecture (which has separate buses for program and data memory), a multilevel pipeline, on-chip peripherals, on-chip memory, and a highly specialized instruction set. The ’C24x performs up to 40 MIPS.

The ’C24x devices are source-code compatible with the ’C2x generation and are upwardly compatible with the ’C5x generation. The ’C5x assembler accepts ’C24x instructions as well as ’C5x instructions. The ’C24x instruction set incorporates all the ’C2x instructions and adds 11 new ones.

The core CPU of the ’C24x features 544 words of dual-access RAM. The external address reach of the ’C24x is 224K words; 64K words of data, program, and I/O, and 32K words of global memory for multiprocessor operation.

The ’C24x series is supported by a full suite of development tools that make designing with these devices easy. Texas Instruments offers a C compiler/ assembler/linker, a simulator, and an emulator/debugger that supports non-intrusive IEEE Standard 1149.1-compliant (JTAG) scan-based emulation. Numerous third-party companies also provide development tools for the ’C24x generation.

The ’C24x generation also features a DSP controller with on-chip flash memory. The TMS320F240 includes 16K words of flash memory, enabling designers to eliminate costly off-chip memory.

4.3 TMS320C24x Key Features

Following are the key features of the ’C24x series:

- ❑ Processing hardware
  - ■ 32-bit central arithmetic logic unit (CALU)
  - ■ 32-bit accumulator
  - ■ 16-bit × 16-bit parallel multiplier with a 32-bit product capability
  - ■ Three scaling shifters
  - ■ Eight 16-bit auxiliary registers with a dedicated arithmetic unit for indirect addressing of data memory
- ❑ Memory
  - ■ 224K words × 16-bit maximum addressable memory space (64K words program, 64K words data, and 64K words I/O, and 32K words global)
  - ■ On-chip dual-access RAM (DARAM)
  - ■ On-chip ROM or flash EPROM
  - ■ External memory interface module with software wait state, 16-bit address lines, and 16-bit data lines
  - ■ Support of hardware wait states
- ❑ Program control:
  - ■ 4-level pipeline operation
  - ■ 8-level hardware stack
  - ■ User-maskable interrupts
- ❑ Instruction set
  - ■ Single-instruction repeat operation
  - ■ Single-cycle multiply/accumulate instructions
  - ■ Memory block move instructions for program/data management
  - ■ Indexed-addressing capability
  - ■ Bit-reversed indexed-addressing capability for radix-2 FFTs

- Power:
  - Static CMOS technology
  - Four power-down modes to reduce power consumption
- Emulation: IEEE Standard 1149.1 boundary-scan logic interfaced to on-chip scan-based emulation logic
- Speed: 50-ns instruction cycle time, with most instructions single cycle
- Code compatibility with TMS320 fixed-point devices:
  - Source code compatible with the ’C25 and ’C2xx devices and upwardly compatible with the ’C5x generations of DSPs
- On-chip peripherals: See Table 4–1 and Figure 4–1.

Figure 4–2 shows a block diagram of the ’C24x DSP controllers.

Table 4–1. Device Configurations

Features	’F240/’C240	’F241/’C241	’F243	’C242
On-chip ROM/flash memory (words)	16K flash/ 16K ROM	8K flash 8K ROM	8K flash	4K ROM
On-chip dual-access RAM (words)	544	544	544	544
Total memory-address range (words)	64K program 64K data 64K I/O 32K global	64K program 64K data		
Timers	3 general-purpose with compares 1 watchdog	2 general-purpose 1 watchdog	2 general purpose 1 watchdog	2 general purpose 1 watchdog
Serial communications interface (SCI)	1	1	1	—
Serial peripheral interface (SPI)	1	1	1	—
Controller Area Network (CAN) module		1	1	
Compare outputs	9 total	5 total	5 total	5 total
Capture inputs	4 total 2 quadrature encoder pulse (QEP)	3	3	3
I/O functions	28	26	32	26
Analog-to-digital convertors (ADCs)	2 10-bit ADCs with 16 total inputs and 10 μs conversion	10 bit	10 bit	10 bit
Pulse-width modulation (PWM) outputs	12 total	8 total	8 total	8 total
Package	132-pin PQFP	68PLCC, 64 PQFP	144 TQFP	68PLCC, 64 PQFP

4.4 TMS320C240/F240

These are key features of the ’C240x and ’F240x:

- ❑ High-performance static CMOS technology
- ❑ Includes the T320C2xLP core CPU
  - ■ Source code compatible with TMS320C25;
  - ■ Upwardly compatible with TMS320C5x
  - ■ 50-ns instruction cycle time
- ❑ Memory
  - ■ 544K words × 16-bits of on-chip data/program dual-access RAM
  - ■ 16K words × ×16 bits of on-chip program ROM (’C240) / flash memory (’F240)
  - ■ 192K words × 16 bits of total memory address reach
- ❑ Event-manager module with 12 pulse width modulation (PWM) outputs on the ’F240/’C240, 8 PWMs on the ’F241/’C241, 8 PWMs on the ’F243, and 8 PWMs on the ’C242
- ❑ Dual 10-bit analog-to-digital conversion module
- ❑ 28 individually programmable, multiplexed I/O pins
- ❑ Phase-locked loop (PLL)-based clock module
- ❑ Watchdog timer module (with real-time interrupt)
- ❑ Serial communication interface (SCI) module
- ❑ Serial peripheral interface (SPI) module
- ❑ Six external interrupts
- ❑ Four power-down modes for low-power operation
- ❑ Scan-based emulation
- ❑ Extended temperature range options available

Figure 4–1. TMS320C240/F240 Peripheral Overview

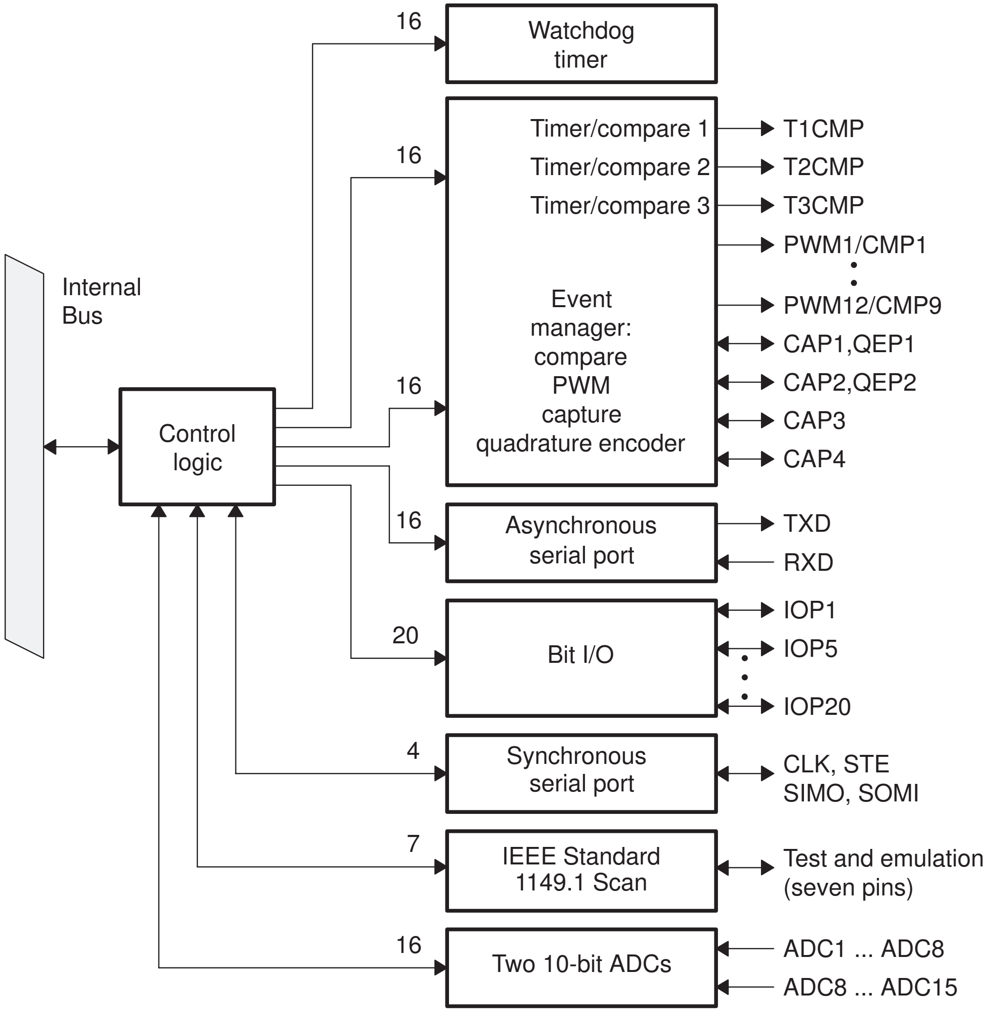
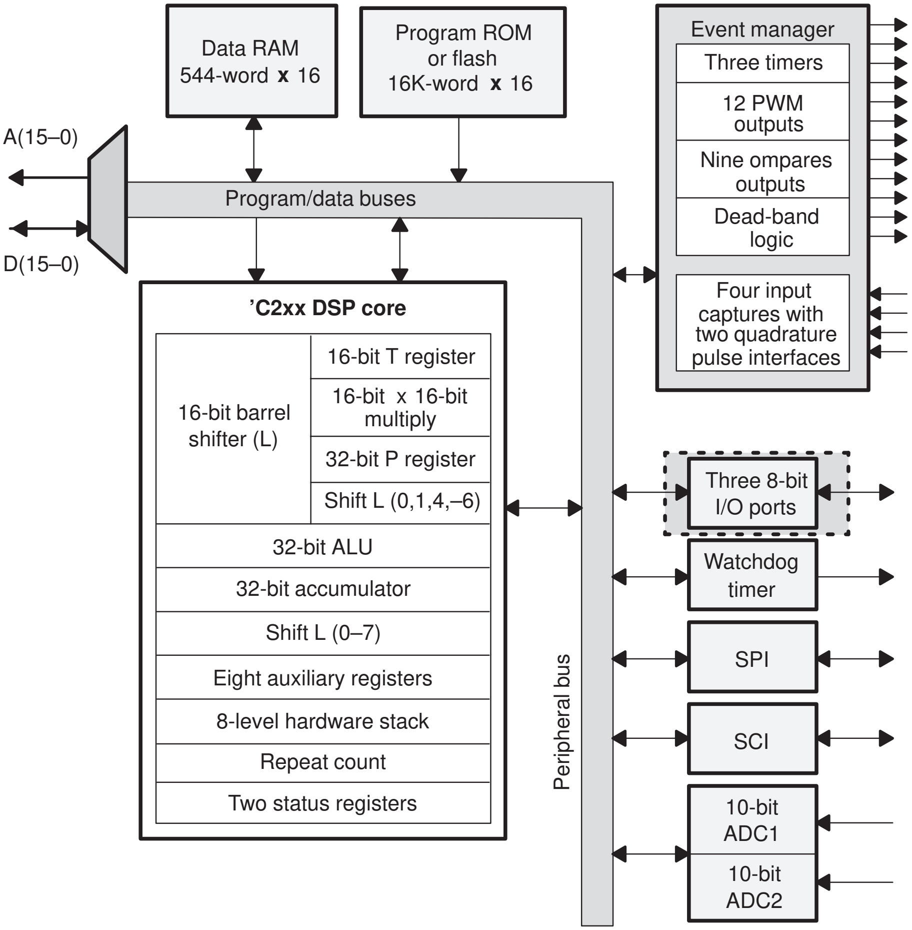


Figure 4–2. TMS320C240/F240 Block Diagram

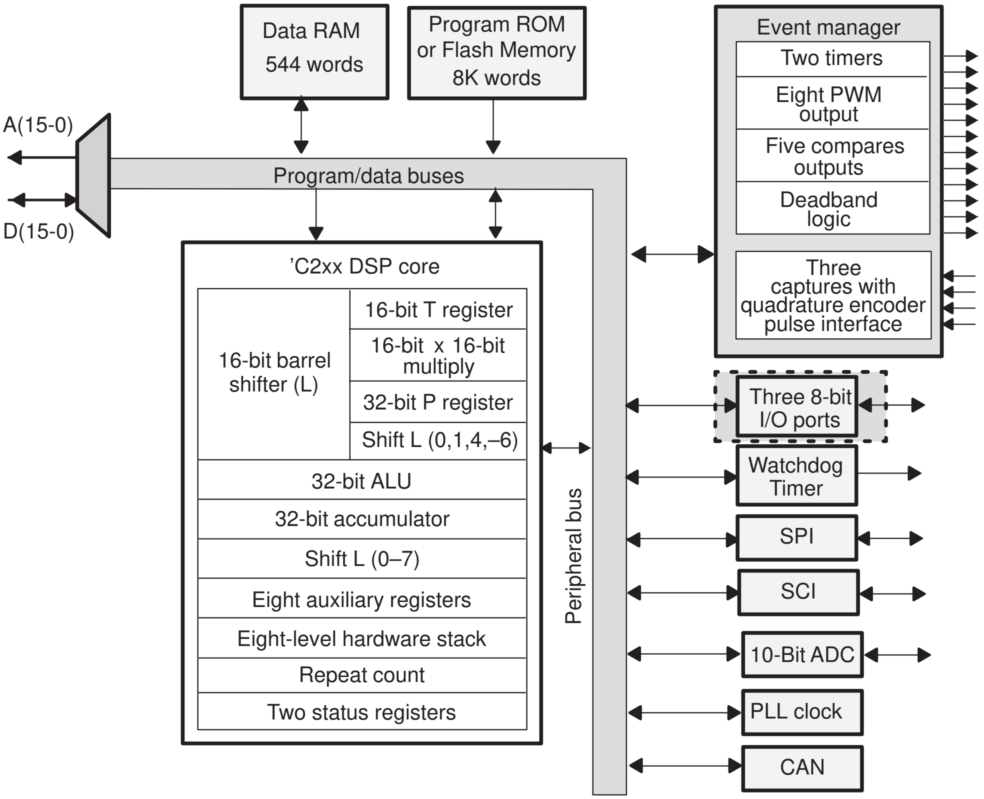


4.5 TMS320C241/F241

The ’C241 device has the following features:

- ❑ ’C2xx DSP core, 5-V
- ❑ 8K words × 16 bits of on-chip ROM (Flash memory for ’F241)
- ❑ Motor control-optimized event manager (8 PWM outputs)
- ❑ 10-bit A/D converter with 800-ns conversion time
- ❑ On-chip CAN module (meets specification 2.0B)
- ❑ ’C241 supported by the ’F241 for prototyping/preproduction needs
- ❑ Extended temperature range options
- ❑ 68-pin PLCC, 64-pin PQFP package
- ❑ ’F241 footprint-compatible with ’C241 for preproduction prototyping

Figure 4–3. TMS320C241/F241 Block Diagram

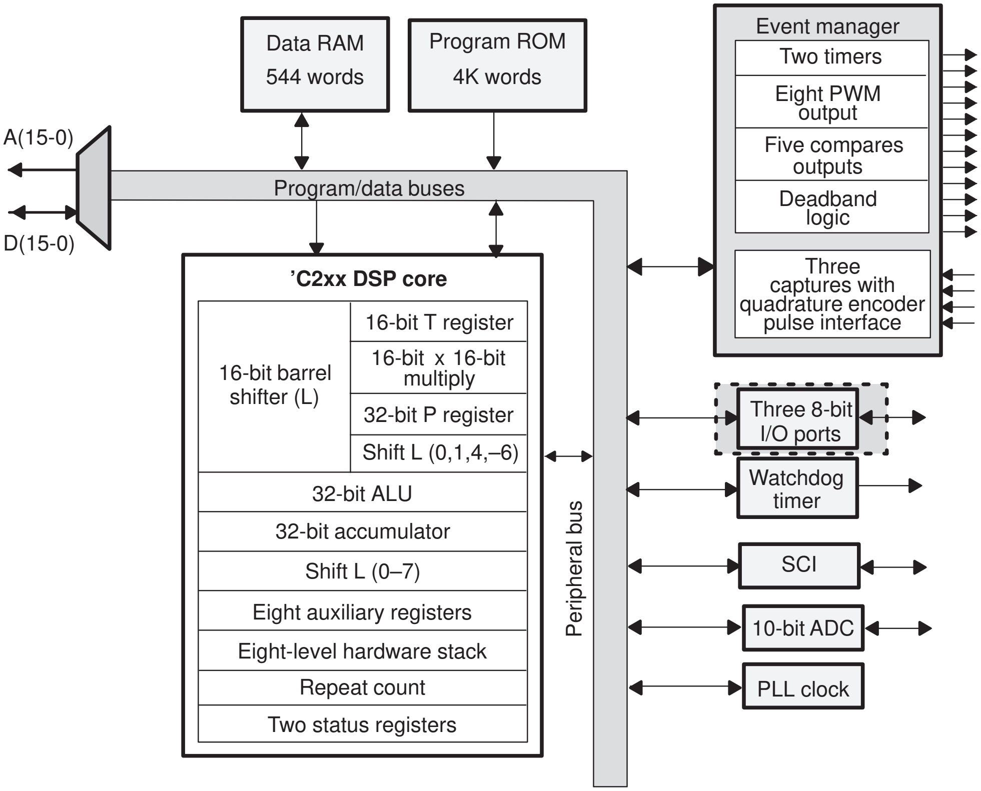


4.6 TMS320C242

The ’C242 device has the following features:

❑ ’C2xx DSP core, 5-V
❑ 4K words of on-chip ROM
❑ Motor control-optimized event manager (8 PWM outputs)
❑ 10-bit A/D converter with 800-ns conversion time
❑ Supported by the ’F243 for prototyping/preproduction needs
❑ No CAN and SPI
❑ Extended temperature range options
❑ 68-pin PLCC, 64-pin PQFP package

Figure 4–4. TMS320C242 Block Diagram

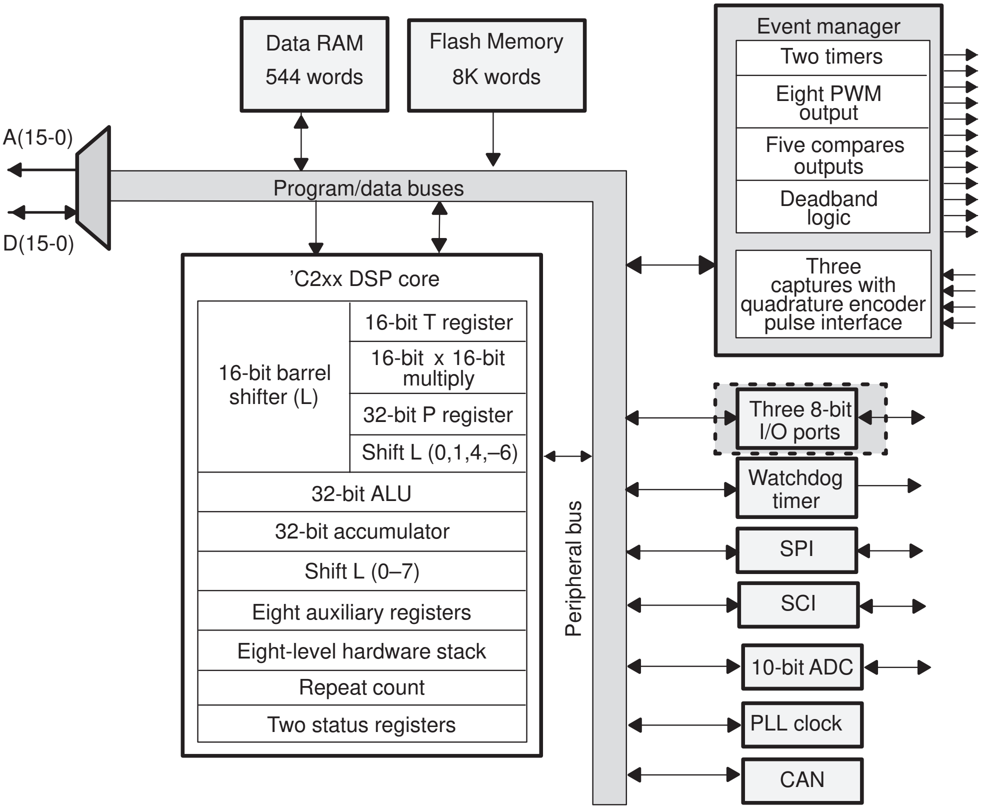


4.7 TMS320F243

The ’F243 device has the following features:

❑ ’C2xx DSP core, 5-V
❑ 8K words of on-chip flash memory
❑ Motor control-optimized event manager (8 PWM outputs)
❑ 10-bit A/D converter with 800-ns conversion time
❑ On-chip CAN module (meets specification 2.0B)
❑ 16-bit external memory interface
❑ Extended temperature range options
❑ 144-pin TQFP package

Figure 4–5. TMS320F243 Block Diagram



4.8 TMS320C24x Memory Bus

The ’C2xx advanced Harvard-type architecture maximizes processing power by maintaining two separate memory bus structures, program and data, for full-speed execution. This multiple bus structure allows reading both data and instructions simultaneously. Instructions support data transfers between the two spaces. This architecture lets you store coefficients in program memory to be read in RAM, eliminating the need for a separate coefficient ROM. This, coupled with a 4-level-deep pipeline, allows the ’C2xx to execute most instructions in a single cycle.

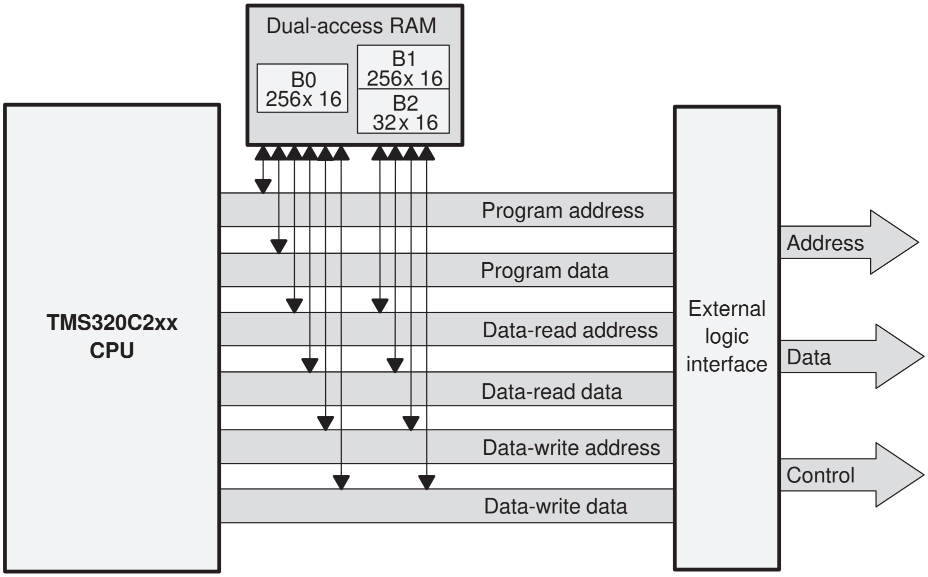
The ’C2xx DARAM allows writes to and reads from the RAM in the same cycle without the address restrictions of single-access RAM (SARAM). The DARAM is configured in three blocks: block B0, block B1, and block B2. Block B0 is a 256-word block that can be configured as data or program memory. Block B1 is 256 words in data memory and block B2 is 32 words in data memory.

The ’C240 includes 16K words of ROM. The ’F240 includes 16K words of flash memory. Flash memory offers an excellent alternative to RAM because of its lower cost and nonvolatile storage and is an alternative to ROM due to its reprogrammability.

The ability of the DARAM to perform two accesses in one cycle, coupled with the parallel nature of the ’C2xx architecture, enables the ’C2xx devices to perform three concurrent memory accesses in any given machine cycle.

Figure 4–6 shows a block diagram of the memory bus.

Figure 4–6. TMS320C24x Memory Bus



4.9 TMS320C240/F240 Event Manager (EV) Module

The ’C240/’F240 event manager module consists of timers, compare units, simple compares, a capture unit, and a quadrature encoder pulse (QEP) circuit. These items are discussed in the following subsections. Figure 4–7 shows a block diagram of the ’C240 event manager.

4.9.1 General-Purpose (GP) Timers

TMS320C240, TMS320F240, TMS320C241, TMS320F241, TMS320C242, and TMS320F243 devices contain GP timers. The GP timer includes:

- ❑ A 16-bit timer up-, up/down-counter
- ❑ A 16-bit timer compare register (with shadow register)
- ❑ A 16-bit timer period register (with shadow register)
- ❑ A 16-bit timer control register
- ❑ Selectable internal or external input clocks
- ❑ A programmable prescaler for either internal or external counter input clocks
- ❑ Control logic and four maskable interrupts and interrupt flags: underflow, overflow, timer compare, and period interrupts
- ❑ A timer compare output pin with configurable low- and high- active states
- ❑ A selectable direction (DIR) input pin (to count up or down when the directional up/down count mode is selected)

The GP timers can be operated independently or in synchronization. A 32-bit GP timer can be configured using GP timer2 and timer3. The compare register associated with each GP timer can be used for compare function and PWM waveform generation. There are two single and three continuous modes of operation for each GP timer in up- or up/down-counting operations. Internal or external input clocks with a prescaler are used for each GP timer. The state of each GP timer/compare output is configurable by the general-purpose timer control register (GPTCON). GP timers also provide a timebase for the ’C240 peripherals; GP timer1 for all the compares and PWM circuits, GP timer1 or timer2 for the simple compares to generate additional compare or PWMs, GP timer2 or timer3 for the capture units and the QEP counting operations.

4.9.2 Compare Units

There are three full compare units on the ’C240. These compare units use GP timer1 as the timebase and generate six outputs for compare and high-precision PWM waveforms generation using the programmable deadband circuit. The states of the six outputs are configurable independently.

4.9.2.1 Simple Compares

The ’C240 is equipped with three simple compares that can be used to generate three additional independent compare or high-precision PWM waveforms. GP timer1 or timer2 can be selected as the timebase for the three simple compares. The states of the outputs of the three simple compares are configurable as low-active or high-active, or can be forced low or high independently.

4.9.2.2 Compare/PWM Waveform Generation

Up to 12 compare and/or high-precision PWM waveforms (outputs) can be generated by the ’C240; three independent pairs (six outputs) by the three compare units with *programmable deadbands*, three independent compare or PWMs (three outputs) by the simple compares, and three independent compare and PWMs (three outputs) by the GP timer compares.

4.9.2.3 Compare/PWMs Characteristics

The characteristics of the compare/PWM waveforms include:

- ❏ 16-bit, 50-ns resolutions
- ❏ Programmable deadband for the PWM output pairs, from 0 to 102 μs
- ❏ Minimum deadband width of 50 ns
- ❏ On-the-fly change of the PWM carrier frequency, the PWM frequency wobbling
- ❏ On-the-fly change of the PWM pulse widths within and after each PWM period
- ❏ External maskable power and drive protection interrupts
- ❏ Pulse pattern generator circuit (Figure 4–7), for programmable generation of asymmetric, symmetric, and four-space vector PWM waveforms
- ❏ Minimized CPU overhead using auto-reload of the compare and period registers

4.9.3 Capture Unit

The capture unit provides a logging function for different events or transitions. The values of the GP timer2 counter and/or GP timer3 counter are captured and stored in the 2-level FIFO stacks when selected transitions are detected on capture input pins. The capture unit of the ’C240 consists of four capture circuits.

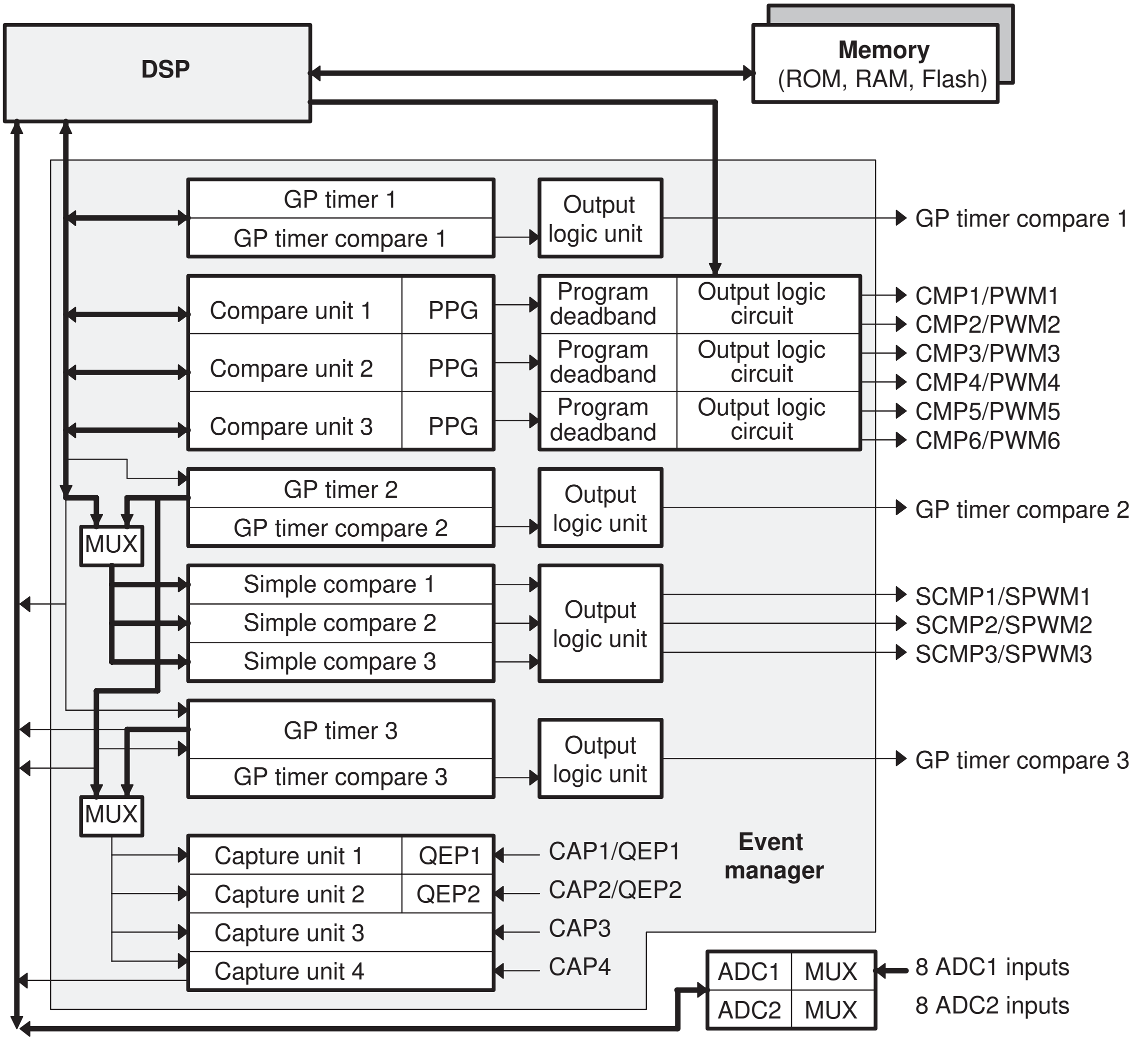
The capture unit includes the following features:

- One 16-bit capture control register
- One 16-bit capture FIFO status register
- Optional selection of GP timer2 and/or GP timer3 through two 16-bit multiplexers (MUXs)
- Four 16-bit by 2 FIFO stack registers
- Four possible Schmitt-triggered capture input pins
- User-specified edge-detection mode at the input pins
- Four maskable interrupts/flags

4.9.4 Quadrature Encoder Pulse (QEP) Circuit

Two capture inputs can be used to interface the on-chip QEP circuit with the quadrature pulses generated from an optical encoder. Full synchronization of these inputs is done on the chip. Direction or leading quadrature pulse sequence is detected.

Figure 4–7. TMS320C240 Event Manager Block Diagram/Functions



4.10 Serial Peripheral Interface (SPI) Module

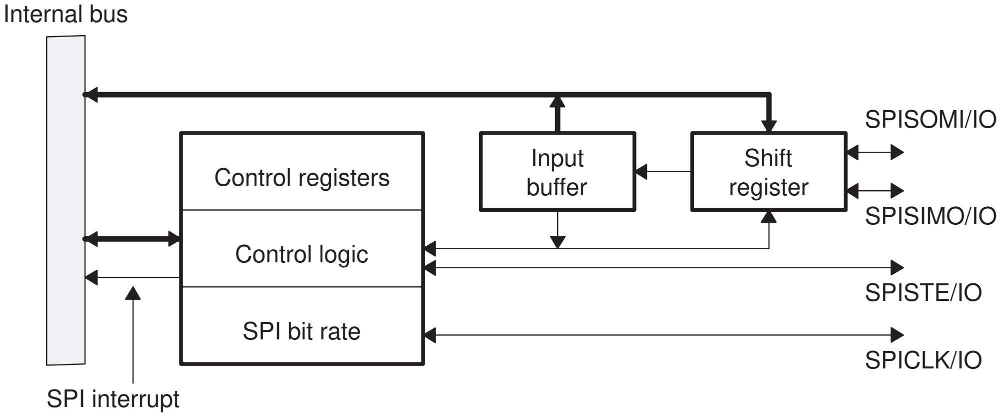
The ’C240, ’F240, ’F241, ’C241, and ’F243 devices include the 4-pin SPI module. The SPI is a high-speed, synchronous serial I/O port that allows a serial bit stream of programmed length (one to eight bits) to be shifted into and out of the device at a programmable bit-transfer rate. The SPI normally is used for communications between the microcontroller and external peripherals or another microcontroller. Typical applications include external I/O or peripheral expansion through devices such as shift registers, display drivers, and ADCs. Multidevice communications are supported by the master/slave operation of the SPI. The SPI module features include:

- ❑ Four external pins:
  - ■ SPISOMI: SPI slave output/master input pin, or general-purpose bidirectional I/O pin
  - ■ SPISIMO: SPI slave input/master output pin, or general-purpose bidirectional I/O pin
  - ■ SPISTB: SPI slave enable strobe pin, or general-purpose bidirectional I/O pin
  - ■ SPICLK: SPI serial clock pin, or general-purpose bidirectional I/O pin
- ❑ Two operational modes: master and slave
- ❑ Baud rate: 128 different programmable rates/2.5 Mbps at 10 MHz SYSCLK
- ❑ Data-word format: one to eight data bits
- ❑ Four clocking schemes controlled by the clock polarity and clock phase bits:
  - ■ Falling edge without phase delay: SPICLK inactive high. SPI transmits data on the falling edge of the SPICLK and receives data on the rising edge of the SPICLK signal.
  - ■ Falling edge with phase delay: SPICLK inactive high. SPI transmits data one-half cycle ahead of the falling edge of the SPICLK and receives data on the falling edge of the SPICLK signal.
  - ■ Rising edge without phase delay: SPICLK inactive low. SPI transmits data on the rising edge of the SPICLK and receives data on the falling edge of the SPICLK signal.
  - ■ Rising edge with phase delay: SPICLK inactive low. SPI transmits data one-half cycle ahead of the falling edge of the SPICLK and receives data on the rising edge of the SPICLK signal.

- Simultaneous receive and transmit operation (transmit function can be disabled in software)
- Transmitter and receiver operations are accomplished through either interrupt driven or polled algorithms.
- Ten SPI module control registers located in the control register frame beginning at address 7040h

Figure 4–8 shows a block diagram of the SPI module.

Figure 4–8. 4-Pin Serial Peripheral Interface (SPI) Module Block Diagram



4.11 Serial Communications Interface (SCI) Module

The 'C240, 'F240, 'F241, 'C241, and 'F243 devices include an SCI module. The SCI module supports digital communications between the CPU and other asynchronous peripherals that use the standard NRZ format. The SCI's receiver and transmitter are double-buffered and each has its own separate enable and interrupt bits. Both may be operated independently or simultaneously in the full-duplex mode. To ensure data integrity, the SCI checks received data for break detection, parity, overrun, and framing errors. The speed of the bit rate (baud) is programmable to over 65 000 different speeds through a 16-bit baud-select register.

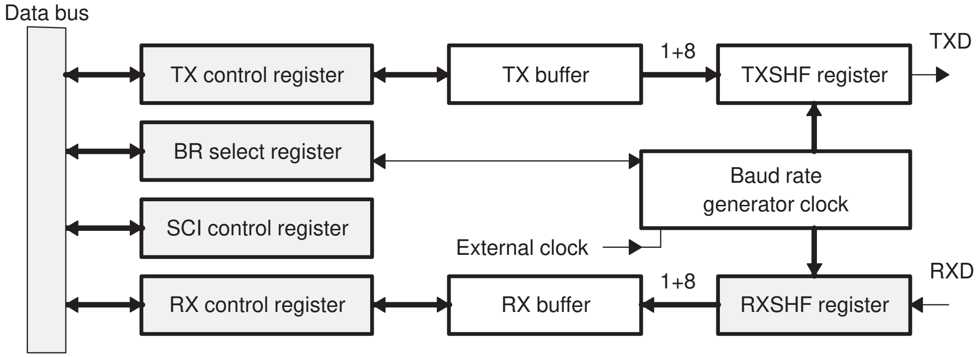
Features of the SCI module include:

- Three external pins:
  - SCITXD: SCI transmit output pin or general-purpose bidirectional I/O pin
  - SCIRXD: SCI receive input pin or general-purpose bidirectional I/O pin
  - SCICLK: SCI bidirectional serial clock pin or general-purpose bidirectional I/O pin
- Two communications modes: asynchronous and isosynchronous
- Baud rate: 64K different programmable rates
  - Asychronous mode: 312 Kbps at 10-MHz SYSCLK
  - Isosychronous mode: 5 Mbps at 10-MHz SYSCLK
- Data word format:
  - One start bit
  - Data word length programmable from one to eight bits
  - Optional even/odd/no parity bit
  - One or two stop bits
- Four error-detection flags: parity, overrun, framing, and break-detection
- Two wake-up multiprocessor modes: idle-line and address bit
- Half- or full-duplex operation
- Double-buffered receive and transmit functions

- Transmitter and receiver operations can be accomplished through either interrupt-driven or polled algorithms with status flags:
  - Transmitter: TXRDY flag (transmitter buffer register is ready to receive another character) and TX EMPTY flag (transmitter shift register is empty)
  - Receiver: RXRDY flag (receive buffer register ready to receive another character), BRKDT flag (break condition occurred), and RX ERROR (monitoring four interrupt conditions)
  - Separate enable bits for transmitter and receiver interrupts (except BRKDT)
  - Non-return-to-zero (NRZ) format
  - Twelve SCI module control registers located in the control register frame beginning at address 7050h

Figure 4–9 shows the SCI module block diagram.

Figure 4–9. TMS320C24x Serial Communication Interface (SCI)



4.12 Controller Area Network (CAN) Module

The ’C241, the ’F241, and the ’F243 devices include a CAN module.

The CAN uses a serial multimaster communication protocol that efficiently supports distributed real-time control, with a very high level of security and a communication rate of up to 1 megabit per second (@13-MHz system clock) or 769.23 kilobits per second (@10-MHz system clock). The CAN bus is ideal for applications operating in a noisy and harsh environment, such as in the automotive and other industrial fields requiring reliable serial communication or multiplexed wiring.

Prioritized messages of up to eight bytes in data length can be sent on a multimaster serial bus using an arbitration protocol and an error detection mechanism for a high level of data integrity.

The CAN module provides the CPU with the full functionality of the CAN specification, version 2.0B. The module minimizes the CPU’s load in communication overhead and enhances the CAN standard by providing additional features.

Message objects are used to transmit and receive messages on the CAN bus. The CAN module can configure up to 16 message objects with each message object having its own specific behavior, control segment, data buffer, identifier, and filtering scheme. The CAN handles either standard or extended frames (*standard frames* use a 11-bit identifier and *extended frames* use a 29-bit identifier.)

The CPU controls the CAN module and accesses message objects through a communication memory.

The following are properties of the CAN protocol:

- Prioritizes of messages
- Ensures latency times
- Provides configuration flexibility
- Provides multicast reception with time synchronization
- Provides system-wide data consistency
- Provides a multimaster communication protocol
- Provides error detection and error signaling
- Provides automatic retransmission of corrupted messages as soon as the bus is idle
- Provides distinction between temporary errors and permanent failures of nodes and autonomous switching off of defective notes.

4.12.1 CAN Module Features

The following are features of the CAN module:

- Full support for CAN Specification 2.0B
  - Standard data and remote frames
  - Extended data and remote frames
- Programmable transmission rate
  - Up to 1 Mbps (for devices that can operate at 13-MHz SYSCLK and higher)
  - Up to 769.23 kbps (for devices that can operate at a maximum of 10 MHz)
- Low DSP load for CAN operation and management
  - Simple initialization structure
  - Standalone remote frame management
  - Standalone error recovery management

- ❑ Definition of up to 16 message objects
  - ■ Full identifier and full mask filtering for each object
  - ■ Standard or extended frames handled by each object independently
  - ■ Management of transmitting message objects under consideration of the message priorities
  - ■ Fast access to message data, status and control registers
  - ■ Up to eight data bytes per message
- ❑ Selectable interrupt sources
  - ■ Message transmitted
  - ■ Message received
  - ■ Error condition
  - ■ Global interrupt enable/disable
- ❑ Internal fault test and confinement
  - ■ Self test
  - ■ Bus failure diagnostic
  - ■ Network monitoring
- ❑ Other features
  - ■ Automatic transmission timeout
  - ■ Low-power mode
  - ■ Real-time clock
  - ■ Automatic low-speed transceiver control

4.12.2 CAN Module Physical Description

The CAN module can be split into two layers:

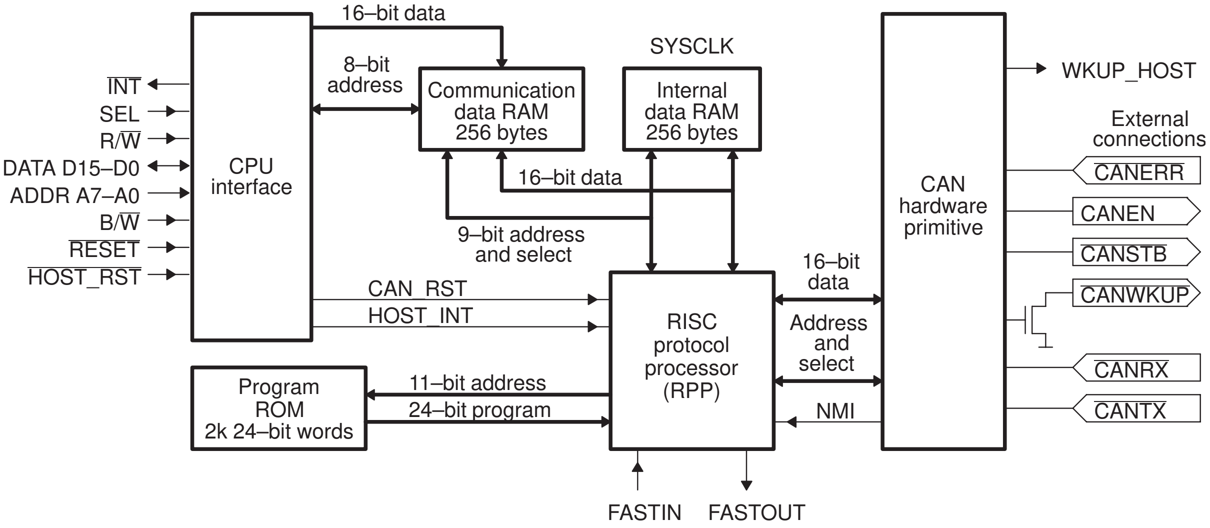
- ❑ A hardware layer (CAN-HP or CAN hardware primitive) that provides the low-level implementation of CAN protocol
- ❑ A software layer (CAN microkernel) that controls frame streams and provides the full implementation of the CAN protocol plus additional features

The software layer is executed by a RISC protocol processor (RPP), which accesses the CAN hardware layer through peripheral registers implemented in the CAN hardware-primitive module.

The CPU accesses the CAN module services through a communication memory. The communication memory holds all CAN messages received and to e sent, as well as all exchanged information.

Figure 4–10 shows the CAN interfaces and general architecture.

Figure 4–10. CAN Module



4.13 Analog-to Digital-Converter (ADC) Module

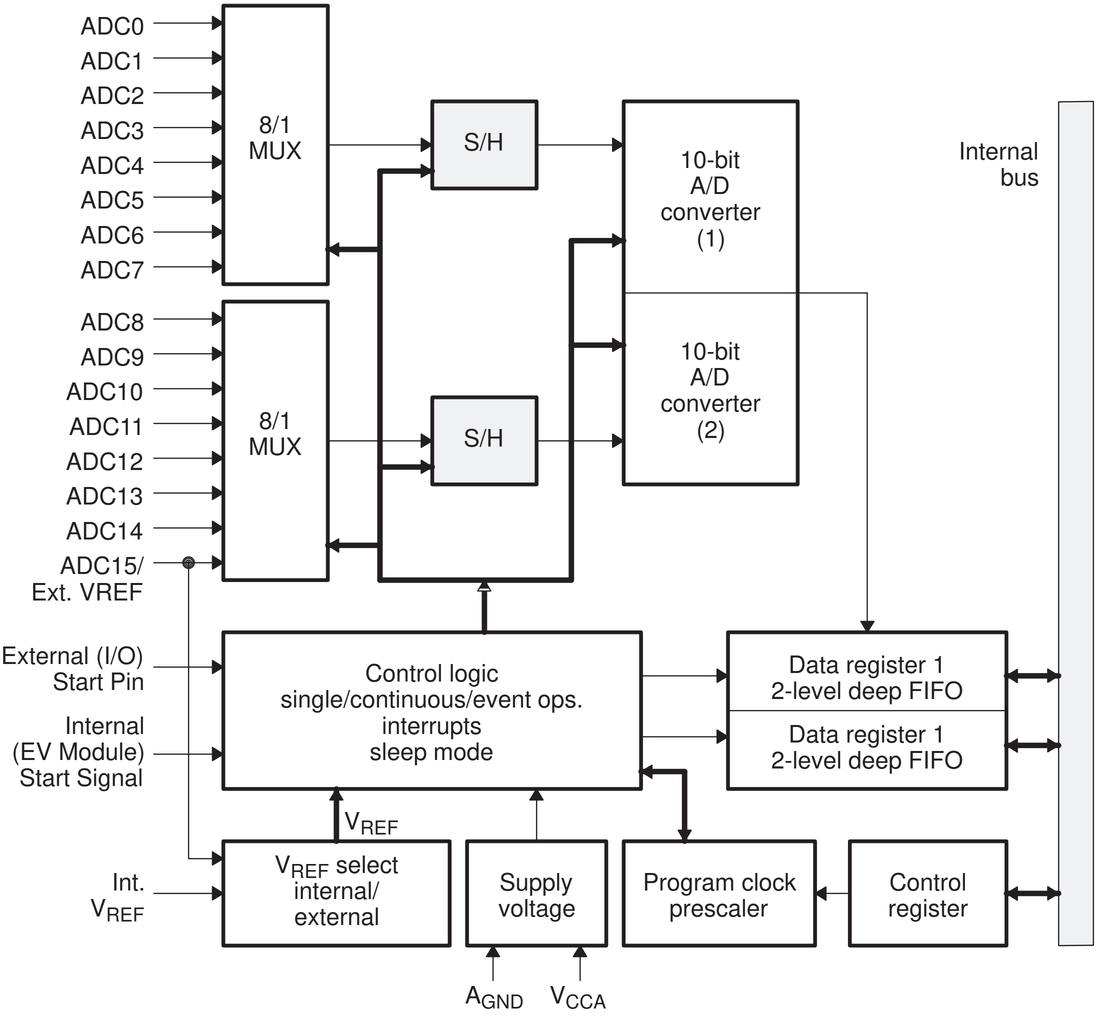
The ADC module consists of two 10-bit analog-to-digital converters with two built-in sample and hold circuits. A total of 16 analog input channels are available on the ’C240:

Features of the ’C240 include:

- ❑ Two input channels (one for each ADC unit) can be sampled and converted simultaneously
- ❑ Each ADC unit can perform single or continuous sample/hold and conversion operations.
- ❑ Two 2-level-deep FIFO result registers for ADC units 1 and 2
- ❑ ADC module (both A/D converters) can start operation by software instruction, by external signal transition on a device pin, or by the event manager events on each of the GP timer/compare outputs and the capture 4.
- ❑ The ADC control register is double buffered (with shadow register) and can be written to at any time. A new conversion of ADC can start either immediately or when the previous conversion process is completed according to the control-register bits.
- ❑ At the end of each conversion, an interrupt flag is set and an interrupt is generated if it is unmasked/enabled.
- ❑ The result of previous conversions stored in data register 1 for ADC1 and in data register 2 for ADC2 are lost when new results are generated.

Figure 4–11 shows a block diagram of the ’C240 ADC module.

Figure 4–11. TMS320C240 Analog-to-Digital Converter Module



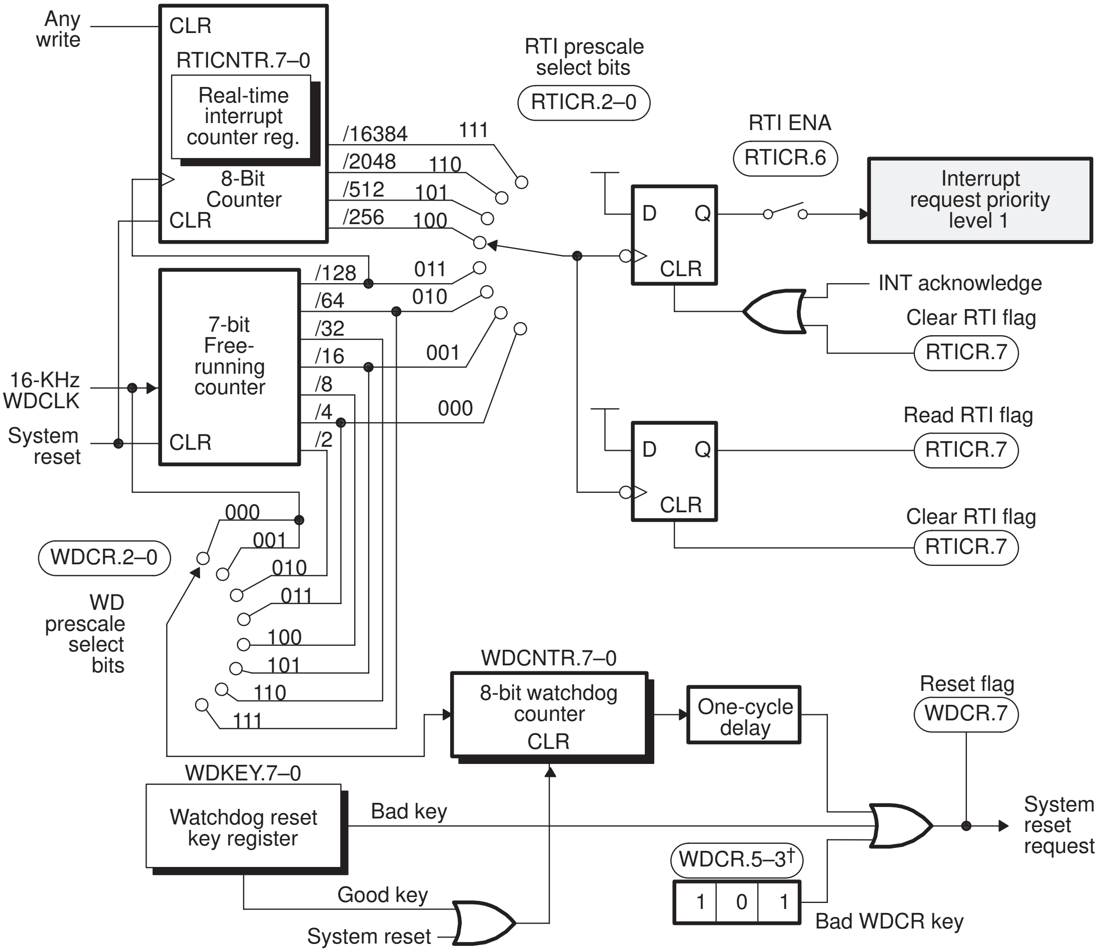
4.14 Watchdog and Real-Time Interrupt (RTI) Module

The ’C240 device includes a watchdog timer and a real-time interrupt module. The watchdog (WD) function of this module monitors software and hardware operation by generating a system reset if it is not periodically serviced by the software, by having the correct key written. The real-time interrupt (RTI) function provides interrupts at programmable intervals. The WD/RTI module features include:

- WD timer:
  - Seven different WD timer overflow rates ranging from 15.63 ms to 1 s
  - A WD reset key (WDKEY) register that clears the WD counter when a correct value is written and generates a system reset if an incorrect value is written to the register
  - A WD flag (WDFLAG) that indicates whether the WD timer initiated a system reset
  - WD check bits that initiate a system reset if an incorrect value is written to the watchdog-control register (WDCR)
  - Automatic activation of the WD timer, once system reset is released
  - Three WD control registers located in the control register frame beginning at address 7020h
- Real-time interrupt (RTI):
  - Interrupt generation at a programmable frequency from 1 to 4096 interrupts per second
  - Interrupt or polled operation
  - Two RTI control registers located in the control register frame beginning at address 7020h.

Figure 4–12 shows a block diagram of the WD/RTI module.

Figure 4–12. WD/RTI Module Block Diagram



† Writing to bits WDCR.5–3 with anything but the correct pattern (101) generates a system reset.

4.15 Flash EEPROM Module Overview

The flash EEPROM is a nonvolatile in-system programmable module that is typically used to program code, calibration tables, system status, or any other piece of information that needs to be available for the life of the device. It can be programmed late in the development cycle and can emulate EEPROM storage using vectored arrays and software block erase algorithms. The array size may be 4, 8, 12, 16, 20, 24, 32, 48, or 64K bytes and is accessible as word-wide (16-bit) memory.

4.15.1 Flash EEPROM Features

The key features of the flash EEPROM are:

- Programming
  - Programming the flash module
  - Erasing the segmented flash module
  - Selecting the flash programming mode
- In-circuit programming capability (no external voltages required
- Write protection—writes to the flash EEPROM are disabled under the following conditions:
  - Reset
  - Low-power modes
  - Not simultaneously writing the KEY1, KEY0, and EXE bits to their proper state to initiate the write or erase sequence

4.15.2 Flash EEPROM Memory

The actual physical address of the flash EEPROM control registers are device specific.

See the specific data sheet for the actual for the flash EEPROM control register you are using on the device you have selected.

4.16 Tools for TMS320C24x

For information about the code generation and debugging tools available for the ’C24x devices, see Chapter 13, *Code Generation Tools*, and Chapter 14, S*ystem Integration and Debugging Tools*. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

Chapter 5

TMS320C3x Devices

The TMS320C3x generation is the first of TI's floating-point digital signal processors. The 'C3x devices provide an easy-to-use, high-performance architecture, which allows users to develop breakthrough products quickly.

'C3x devices can be used in a wide variety of areas including automotive applications, digital audio, industrial automation and control, data communications, and office equipment such as multifunction peripherals, copiers, and laser printers.

Topic **Page**

5.1 TMS320C3x Introduction

The ’C3x has a von Neumann architecture, which provides a unified data and program space. To further enhance performance, the ’C3x devices have four internal data buses and at least one external data bus. The base architecture is enhanced with numerous on-chip peripherals.

The CPU has an independent multiplier and ALU to offer up to 60 million floating-point operations per second (MFLOPS) and up to 30 MIPS.

The DMA controller has its own data bus and operates in parallel with the CPU. The DMA controller is programmed to input and output data sets, freeing the CPU for arithmetic operations. The DMA controller can reach anywhere in the memory map, including on-chip, off-chip, and memory-mapped peripheral registers.

The total memory space of the ’C3x is 16M 32-bit words. Having the data, program, and I/O space contained in this 16M word address range maximizes memory usage and allows you to allocate the memory space as desired. Both 1K-word, 32-bit RAM blocks can support two CPU accesses in a single cycle. The separate program, data, and DMA buses allow parallel program fetches, data reads/writes, and DMA operations. To maintain high performance when using slower external memory, the ’C3x contains a 64-word on-chip cache.

Development tools available from TI for the ’C3x family include the software simulator, C compiler/linker/assembler, hardware emulator, evaluation module, and a DSP Starter Kit. Additionally, there are hundreds of software algorithms and scores of application and development tools available through TI’s extensive third-party network.

5.2 TMS320C3x Key Features

Key features of the ’C3x generation are listed in paragraphs that follow. When only certain devices have a particular feature, they are named in parentheses following the feature’s description.

- ❑ CPU

  33-ns single-cycle instruction execution time (’C31, ’C32)

  60-MHz clock
  60 MFLOPS
  30 MIPS

  40-ns single-cycle instruction execution time with 50-MHz clock on all devices

  50-MHz clock
  50 MFLOPS
  25 MIPS

  50-ns single-cycle instruction execution time

  40-MHz clock
  40 MFLOPS
  20 MIPS

  60-ns single-cycle instruction execution time (’C30)

  33-MHz clock
  33.3 MFLOPS
  16.7 MIPS

  32-bit instruction words, 32-bit data words, and 24-bit addresses

  24/32-bit integer, 32/40-bit floating-point, and 32-bit logical operations

  2-and 3-operand instructions

  Parallel ALU and multiplier instructions in a single cycle

  Block repeat capability

Zero-overhead loops and single-cycle branches

Conditional calls and returns

Interlocked instructions for multiprocessing support

Two address generators with eight auxiliary registers and two auxiliary register arithmetic units

Eight 40-bit, extended-precision registers

32-bit barrel shifter

❑ Peripherals

On-chip memory-mapped DMA controller for concurrent CPU and I/O operations

2-channel configurable priority (’C32)

Memory-mapped serial ports to support 8-, 16-, 24-, or 32-bit full-duplex transfers

One serial port (’C31/C32)
Two serial ports (’C30)

Two memory-mapped 32-bit timers

Two general-purpose external flags and four external interrupts

Scan logic for test and evaluation

❑ Memory

64 × 32-bit instruction cache

One 4K 32-bit-word single-cycle dual-access on-chip ROM block (’C30)

Two 1K 32-bit-word single-cycle dual-access on-chip RAM blocks (’C30/C31)

Two 256 32-bit-word RAM blocks (’C32)

16M word addressing space

Preprogrammed bootloader (’C31/C32)

❑ Memory interfaces

Two memory/I/O expansion buses (’C30)
One external memory bus (’C31/C32)
Flexible 8-, 16-, or 32-bit data size in 8-, 16-, or 32-bit memory (’C32)

- Low power (3.3 V–’C31) with two power-down modes: 2-MHz operation and idle
- CMOS technology
- Packages
  - 181-pin PGA (’C30)
  - 132-pin PQFP (’C31/LC31)
  - 144-pin PQFP
- Temperature rages:
  - Commercial– 0/85°C
  - Extended–40/125°C

5.3 TMS320C3x CPU

The TMS320 CPU has an independent multiplier and accumulator and achieves up to 60 MFLOPS. Results are stored in any one of eight extended-precision registers. These are 40-bit registers that store values with a 32-bit mantissa and an 8-bit exponent. These registers can serve as both the source and destination for any arithmetic operation. The extended-precision registers are an extremely valuable resource for programming in assembly or C. These registers allow you to maintain intermediate results without storing data in memory. This results in high-performance assembly code and an efficient C compiler.

To sustain 60 MFLOPS, the CPU has two independent auxiliary register arithmetic units (ARAUs). The two ARAUs generate 24-bit addresses that are accessed through the eight auxiliary registers. The ARAUs can perform any of these functions:

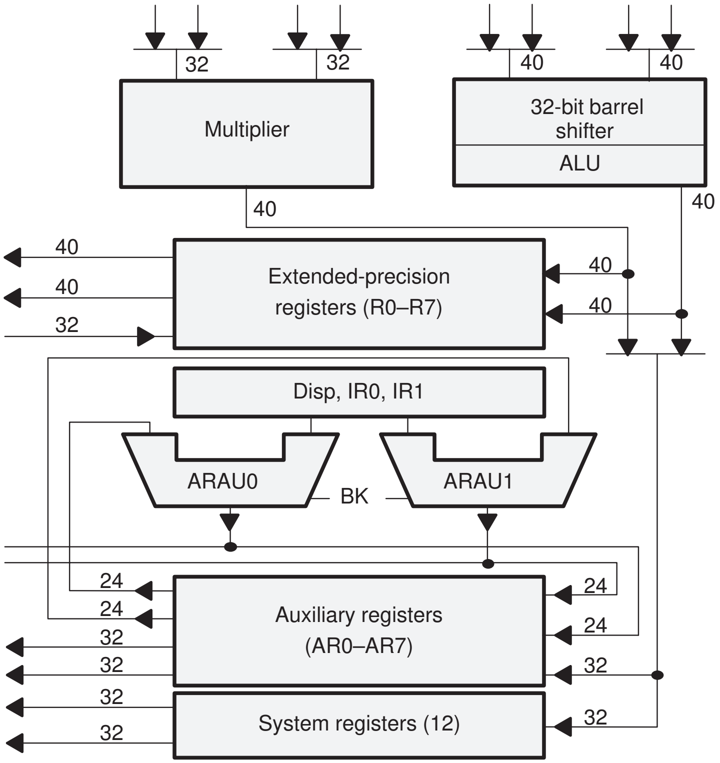
- Pre- or post increment or decrement
- Index offset for increment and decrement values other than 1
- Circular addressing to support circular buffers
- Bit-reversed addressing for FFTs

Features of the TMS320C3x CPU include:

- 60-MFLOP CPU
- Register-based CPU
- 32 or 40 floating-point/integer multiplier
- 32 or 40 floating-point/integer ALU
- 32-bit barrel shifter
- Eight 40-bit extended-precision registers
- Two address generators
- Two index registers
- Eight indirect-address registers

Figure 5–1 shows a block diagram of the ’C3x CPU.

Figure 5–1. TMS320C3x CPU



5.4 TMS320C3x Memory

To realize the full performance of the ’C3x CPU, it is important to have a bus and memory architecture that can keep pace. The ’C3x fetches up to four words per cycle. These consist of a program opcode, two CPU data operands, and a DMA data transfer. The internal buses can transfer all four words in parallel, relying on seven memory sources for data.

The ’C3x uses seven internal buses to access on-chip resources:

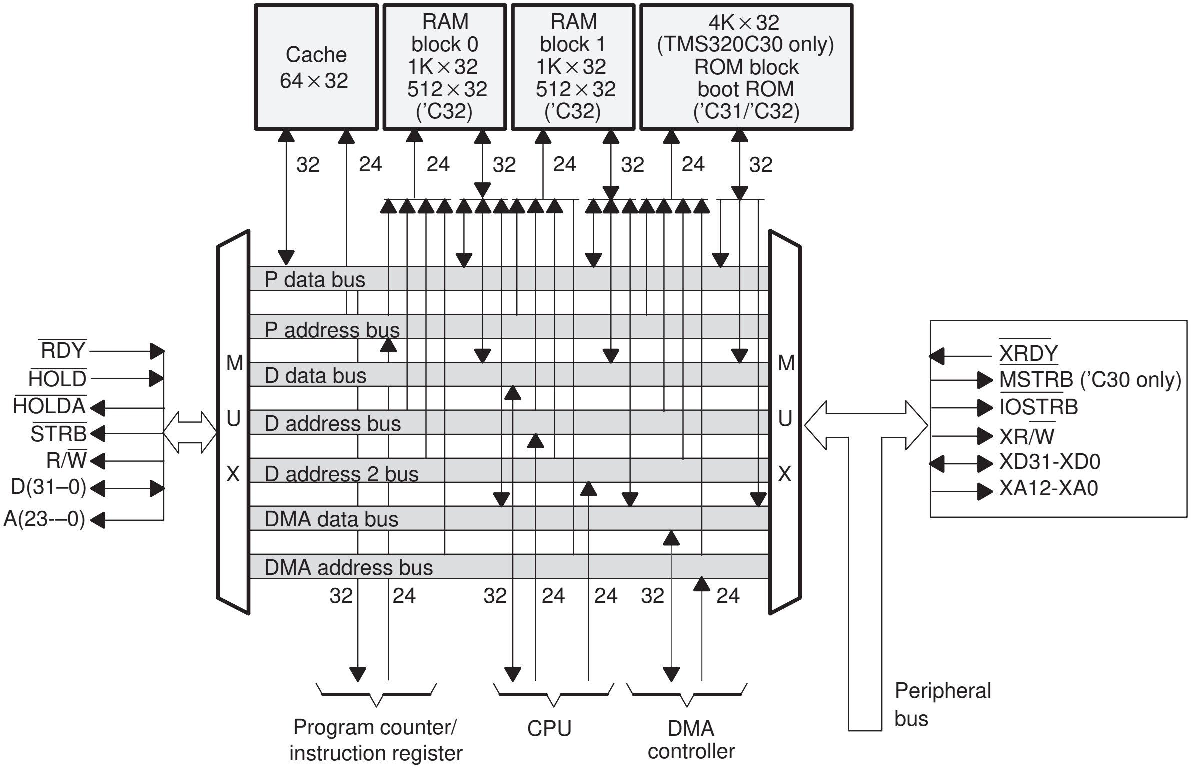
- Program address/data: The CPU uses these buses to maintain instruction fetches every cycle.
- Data address/data: In any cycle, the CPU can fetch two data operands, because it has two data address buses and one data bus that can be accessed twice in a single cycle.
- DMA address/data: The DMA uses these buses to perform DMA transfers in parallel with CPU operation.

With the internal buses in place to feed the DMA and CPU, the ’C3x devices can use both internal and external data and program memory. The ’C30 and ’C31 have two 1K 32-bit-word blocks of dual-access RAM, while the ’C32 has two 256K 32-bit words of on-chip RAM. This memory provides up to four words of program or data in a single cycle. All ’C3x devices feature an on-chip cache to boost system performance. The primary bus for each device has 16M words of address reach. The ’C30 features an expansion bus that has an 8K-word address reach, which is often used to interface to peripherals.

The ’C32 offers the ability to access 8-, 16-, and 32-bit data stored in 8-, 16-, and 32-bit wide external memory, giving the flexibility of nine memory interface options. This feature can significantly affect total system cost savings. Additionally, the ’C32 memory interface allows for storage of the 32-bit instruction word in either 16- or 32-bit-wide external memory.

Figure 5–2 shows a block diagram of the ’C3x memory.

Figure 5–2. TMS320C3x Memory



5.5 TMS320C3x DMA Controller

The DMA controller transfers data between memory resources. The serial ports and timers on the ’C3x are memory mapped, allowing DMA transfers to and from these peripherals. To perform a transfer, the DMA reads a memory location pointed to by the source address register and then writes to the memory location pointed to by the destination address register. The source and destination addresses are incremented or decremented after each transfer, depending on the value of the global control register. The DMA controller performs continuous transfers over the DMA bus until the value in the transfer counter register reaches 0, and a programmable interrupt is sent to the CPU.

For example, an application might use the DMA to transfer 512 words from slow external memory to the on-chip RAM. At the completion of the transfer, an interrupt is sent to the CPU to process and output results while the DMA transfers a new set of 512 words to on-chip RAM. By off-loading data input, the DMA controller allows sustained CPU performance for arithmetic calculations. In this case, the CPU always has zero-wait-state access to data, even though the external memory requires one or more wait states.

The ’C32 offers the programmer the flexibility of designating priority on the bus. There are three options:

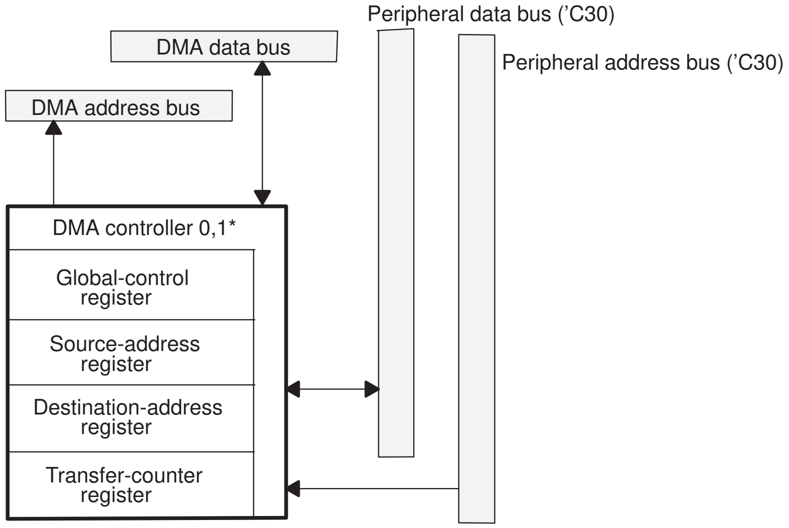
- ❑ The CPU has priority over the DMA at all times (’C30 and ’C31).
- ❑ The DMA controller has priority over the CPU.
- ❑ The CPU and DMA share a rotating priority with the CPU having first access.

Features of the DMA controller include:

- ❑ Increased CPU-sustained performance by virtually eliminating CPU I/O
- ❑ Memory-to-memory transfers
- ❑ 2-channel configurable priority (’C32 only)
- ❑ Programmable increment or decrement of addresses

Figure 5–3 shows a block diagram of the DMA controller.

Figure 5–3. TMS320C3x DMA Controller


Peripheral data bus (’C30)
DMA data bus
Peripheral address bus (’C30)
DMA address bus
DMA controller 0,1*
Global-control register
Source-address register
Destination-address register
Transfer-counter register

* DMA1 is available only on the ’C32

5.6 TMS320C3x Sum of Products Example

The following code for a sum of products is typical of DSP algorithms and demonstrates the power of the ’C3x architecture. First, note that the repeat single instruction (RPTS) is used for a zero-overhead loop. The parallel bars next to the ADDF instruction indicate that the addition is executed in parallel with the multiplication. Auxiliary registers 0 and 1 are used to fetch the two data operands with a post increment of 1. The multiplier results are placed in one of the eight extended-precision registers. The extended-precision register set is further used as the input and output for the addition. Finally, the DMA can transfer the next set of data for the CPU to process in parallel with the multiply and accumulate. All of these operations take place in a single cycle, illustrating the parallelism in the ’C3x architecture.

Following is an example of a sum-of-products function for
y = a1 ×x1 + a2×x2 + ... + an×xn.

```
        MPY      *AR0++ , *AR1++ , R0
        MPY      *AR0++ , *AR1++ , R2
     RPTS     n-3
        MPYF    *AR0++ , *AR1++ , R0
| |  ADDF     R0 , R2 , R2

     ADDF      R0 , R2
```

5.7 TMS320C30

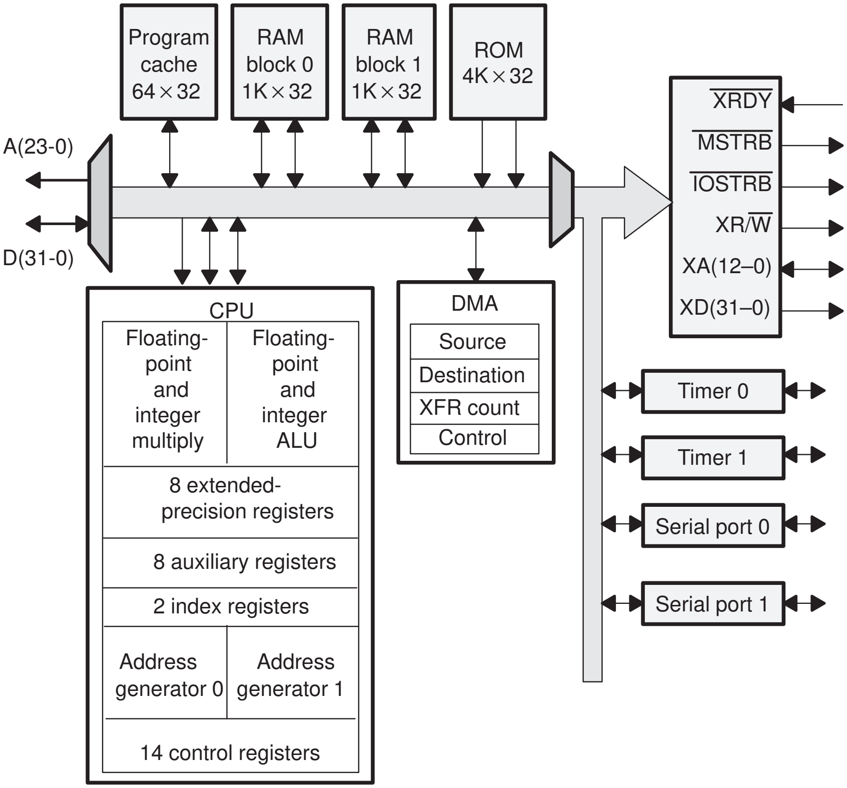
The ’C30 features a second external data bus, two timers, and two serial ports. The expansion bus has a 13-bit address bus and a 32-bit data bus. Each serial port has independent double-buffered transmit and receive sections with a maximum data rate of 15 Mbps with a 60-MHz input clock.

Features of the TMS320C30 include:

- ❑ 40-, 50-, and 60-ns instruction cycle times
- ❑ 16M-word external-address reach
- ❑ Single-cycle multiply and accumulate (MAC) operation
- ❑ Two serial ports
- ❑ Two timers
- ❑ 4K-words on-chip ROM
- ❑ Optimizing ANSI C compiler
- ❑ On-chip DMA
- ❑ Packaging: 181-pin PGA

Figure 5–4 shows a block diagram of the ’C30.

Figure 5–4. TMS320C30 Block Diagram



5.8 TMS320C31

The ’C31 is the second member of the ’C3x generation and is object-code compatible with the ’C30. The ’C31 has the same fast CPU as all other members of the ’C3x generation, but offers a different mix of peripherals to achieve a unique price/performance point.

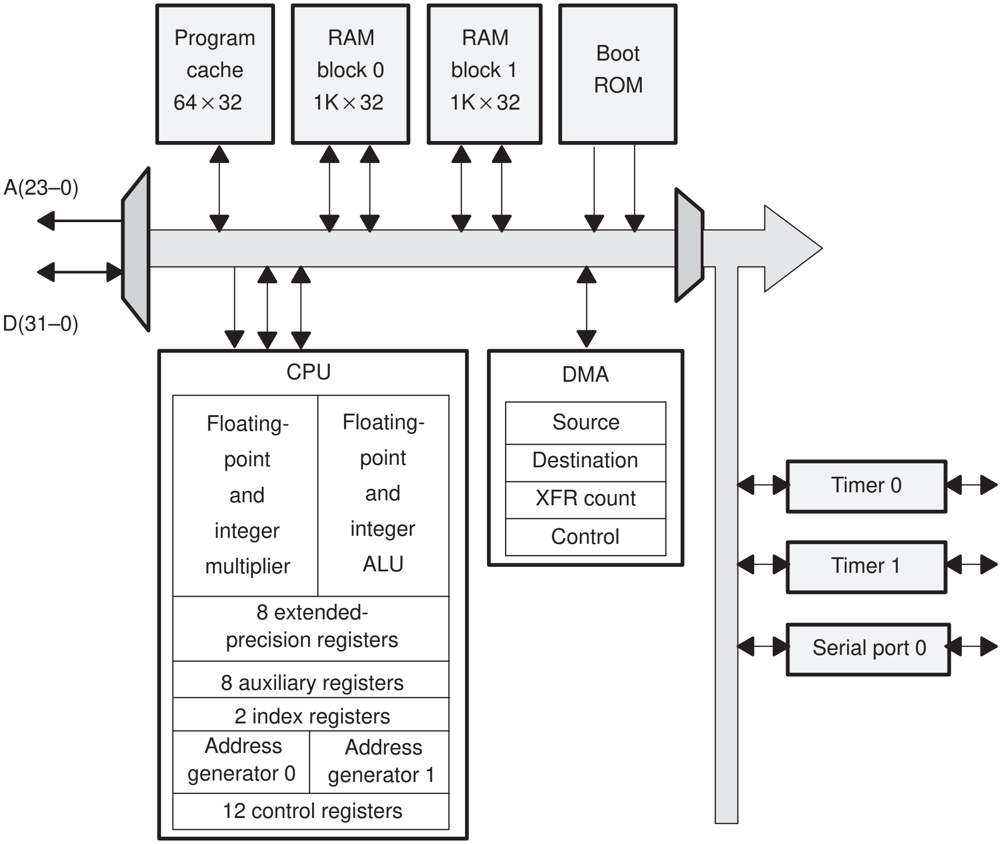
The ’C31 offers a lower cost than the ’C30 by removing the expansion bus and one of the serial ports and replacing the 4K 32-bit words of internal ROM with a boot ROM. A low-power version of the ’C31 is available at 40 MHz at 3.3 V, which significantly reduces power consumption. The ’C31 comes in a 132-pin PQFP.

The TMS320C31 features:

- 33-, 40- and 50-ns MHz clock rates
- 16M word external address reach
- Single-cycle multiply and accumulate (MAC) operation
- Optimizing ANSI C compiler
- On-chip DMA
- Boot ROM
- 3.3-V version up to 40 MHz
- Two low-power modes
- Two 32-bit timers
- Serial port
- 64-word cache
- 132-pin PQFP

Figure 5–5 shows a block diagram of the ’C31.

Figure 5–5. TMS320C31 Block Diagram



5.9 TMS320C32

The ’C32 is the lowest cost floating-point device TI offers. The ’C32 is object-code compatible with the ’C30 and ’C31. The ’C32 has a flexible memory interface that supports 8-, 16-, or 32-bit data types in 8-, 16-, or 32-bit memory. Additionally, it supports program storage in 16- or 32-bit memory. This can lead to considerable savings in system cost.

There are also two low-power modes on the ’C32. One reduces the clock rate of the device but continues execution, while the other suspends instruction execution and puts the device on hold. These are valuable features in power-critical applications.

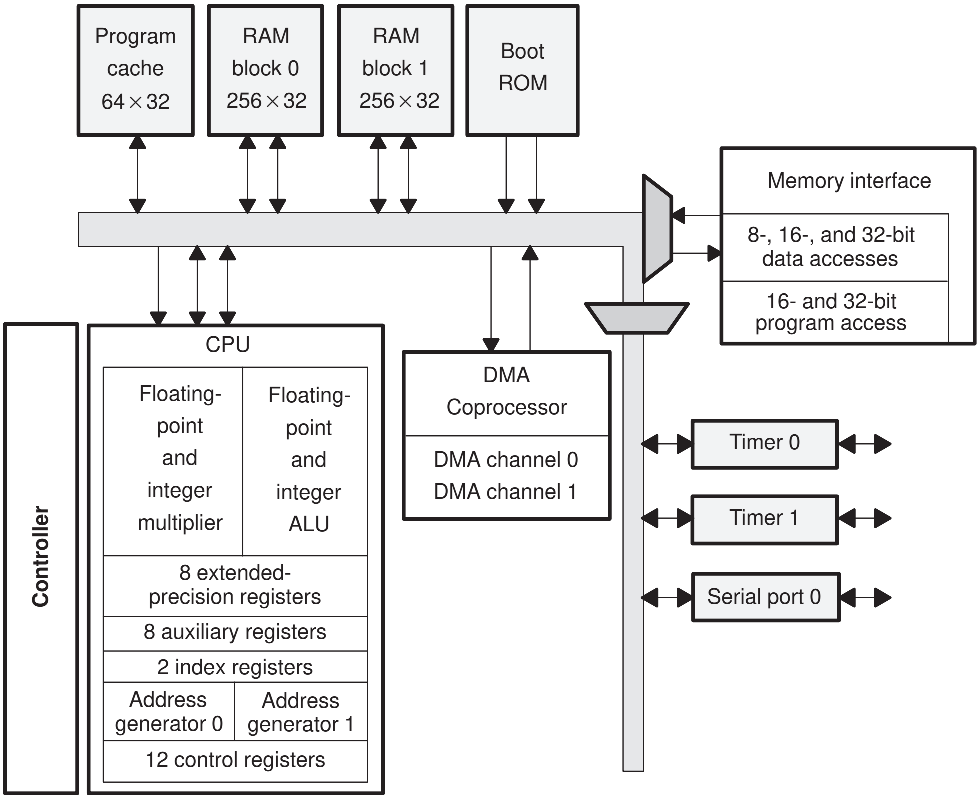
The ’C32 features the same boot ROM as the ’C31, has two 256-word, 32-bit blocks of on-chip RAM, and comes in a 144-pin PQFP.

Features of the TMS320C32 include:

❑ 33-, 40-, and 50-ns instruction cycle times
❑ Object-code compatibility with the ’C30/C31
❑ 16M-word external-address reach
❑ Flexible memory interface (8, 16, or 32 bits)
❑ 2-channel DMA with configurable priorities
❑ Low-power modes
❑ 64-word program cache
❑ Two 32-bit timers
❑ 144-pin PQFP
❑ Serial port

Figure 5–6 shows the ’C32 boot ROM block diagram.

Figure 5–6. TMS320C32 Block Diagram



5.10 Tools for TMS320C3x

For information about the code generation and debugging tools available for the ’C3x devices, see Chapter 13, *Code Generation Tools*, and Chapter 14, *System Integration and Debugging Tools*. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

Chapter 6

TMS320C4x Devices

The TMS320C4x devices are 32-bit floating-point DSPs optimized for parallel processing. The 'C4x family combines a high-performance CPU and DMA controller with up to six communication ports to meet the needs of multiprocessor and I/O-intensive applications.

Key applications of the 'C4x family include 3-dimensional graphics, image processing, networking, and telecommunications base stations.

Topic **Page**

6.1 TMS320C4x Introduction

All ’C4x devices are compatible with TI’s multichip development environment. Each device contains an on-chip analysis module that supports hardware breakpoints for parallel-processing development and debugging. The ’C4x family accepts source code from the TMS320C3x family of floating-point DSPs.

6.2 TMS320C4x Key Features

Key features of the TMS320C4x generation are as follows;

- ❑ Up to 30-MIPS and 60-MFLOPS performance with 384 Mbytes/s of I/O capability
  - ■ IEEE floating-point conversion for ease of use
  - ■ Register-based CPU
  - ■ Single-cycle byte and half-word manipulation capabilities
  - ■ Divide and square root support for improved performance
- ❑ On-chip memory includes 2K words of RAM, 128 words of program cache, and a bootloader
- ❑ Two external buses providing an address reach of up to 4G words
- ❑ Two memory-mapped 32-bit timers
- ❑ 6- and 12-channel DMA
- ❑ Up to six communication ports for multiprocessor communication
- ❑ Idle mode for reduced power consumption
- ❑ Optimizing ANSI C compiler

6.3 TMS320C4x CPU

The ’C4x has an independent multiplier and accumulator and achieves up to 60 MFLOPS. Results are stored in any one of 12 extended-precision registers. These are 40-bit registers that store values with a 32-bit mantissa and an 8-bit exponent. These registers can serve as both the source and destination for any arithmetic operation. The extended-precision registers are an extremely valuable resource for programming in assembly or C. These registers allow maintenance of intermediate results without storing data in memory. This results in high performance assembly code and an efficient C compiler.

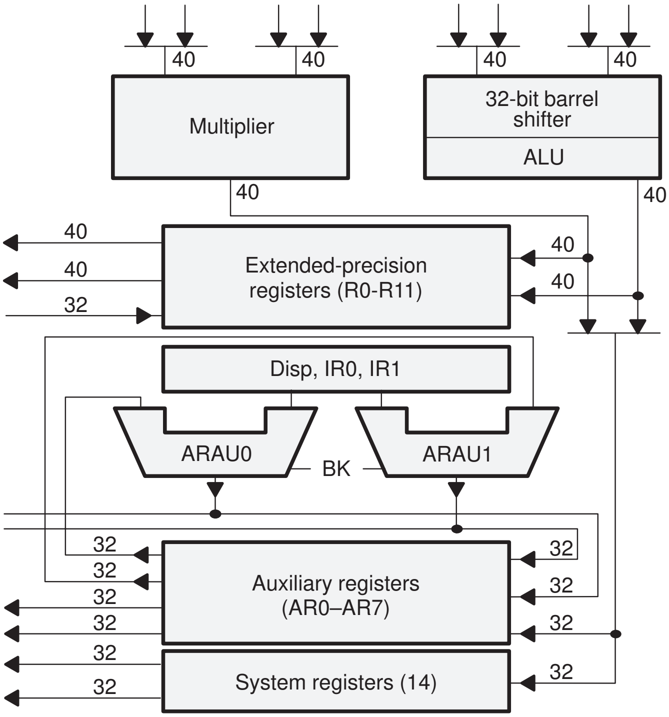
To sustain 60 MFLOPS, the CPU has two independent auxiliary register arithmetic units (ARAUs), which can generate two addresses in a single cycle. The two ARAUs operate in parallel with the multiplier and ALU. They support addressing with displacements, addressing with index registers (IR0 and IR1), circular addressing, and bit-reversed addressing.

Features of the ’C4x CPU are:

- ❑ High-speed internal parallelism: eight operations per cycle for maximum sustained performance
  - ■ Floating-point/integer multiply
  - ■ Floating-point/integer addition
  - ■ Two data accesses
  - ■ Zero-overhead branch and loop counter update
- ❑ IEEE floating-point conversion
- ❑ Divide and square root support for improved performance
- ❑ Single-cycle byte and halfword manipulation capabilities
- ❑ Register-based CPU

Figure 6–1 shows a block diagram of the ’C4x CPU.

Figure 6–1. TMS320C4x CPU



6.4 TMS320C4x Memory and Bus Structure

To realize the full performance of the ’C4x CPU, it is important to have a bus and memory architecture that can keep pace. The ’C4x fetches up to four 32-bit words each cycle: a program opcode, two CPU data operands, and a DMA data transfer. The internal buses can transfer all four words in parallel, relying on seven memory sources for data.

The ’C4x uses seven internal buses to access on-chip resources.

❑ Program address/data: The CPU uses these buses to maintain instruction fetches every cycle.

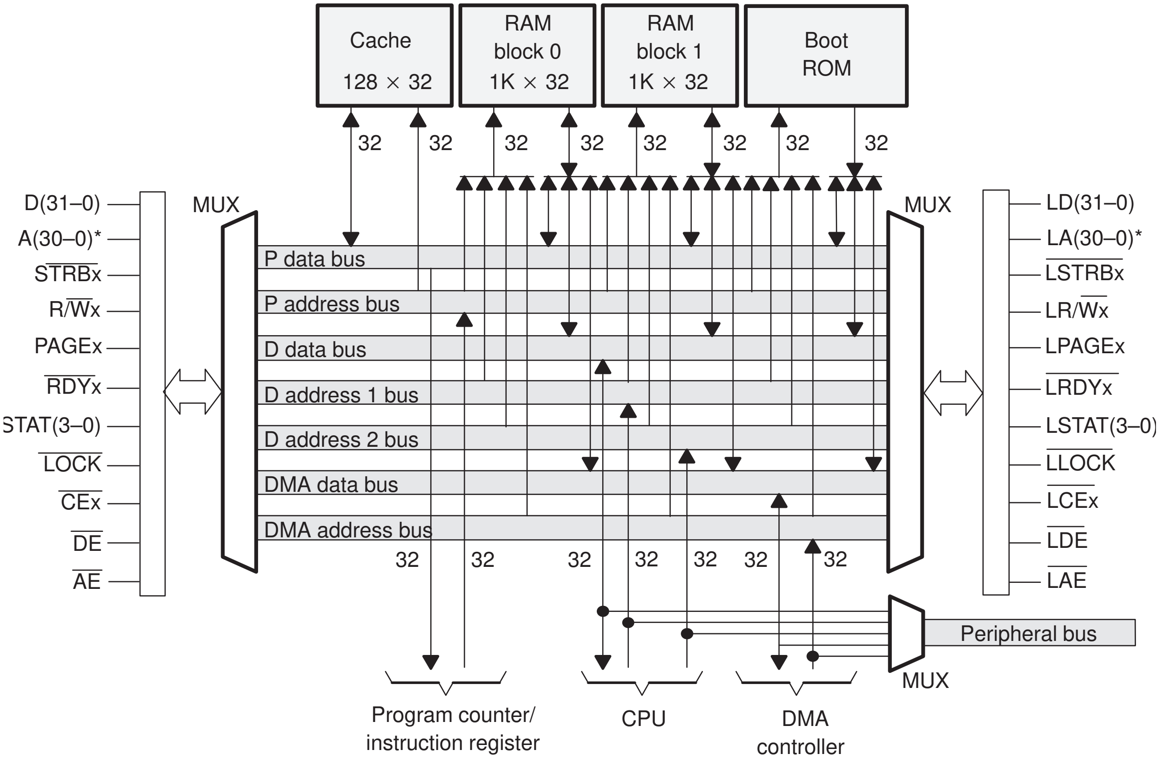
❑ Data address/data: In any cycle, the CPU can fetch two data operands, because it has two data address buses and one data bus that can be accessed twice in a single cycle.

❑ DMA address/data: The DMA uses these buses to perform DMA transfers in parallel with CPU operation.

With the internal buses in place to feed the DMA and CPU, the ’C4x devices can use both internal and external data and program memory. Internally, the ’C4x has two 1K by 32-bit-word blocks of dual-access RAM, providing up to four words of program or data in a single cycle. For external memory, the ’C40 has two identical 32-bit buses, which address up to 2G words of memory each. The ’C44 has two 24-bit external address buses, which address up to 16M words each. Each device has an on-chip instruction cache to boost performance when using slower external memory.

Figure 6–2 shows a block diagram of the ’C4x memory and bus structure.

Figure 6–2. TMS320C4x Memory and Bus Structure



* 24-bit address bus in TMS320C44

6.5 TMS320C4x Communication Ports

The communication ports on the 'C4x generation transfer up to 24 Mbytes/s each for asynchronous interprocessor communications or for servicing intensive I/O needs. The 'C40 has six ports and the 'C44 has four. Each port has four control pins and eight data pins. These 12 pins provide a glueless interface to another 'C4x. The control pins combined with the control logic arbitrate with another device to determine data transfer timing and direction. Because the communication ports have built-in arbitration and control circuitry, you simply need to read data from and write data to the memory-mapped input and output FIFOs.

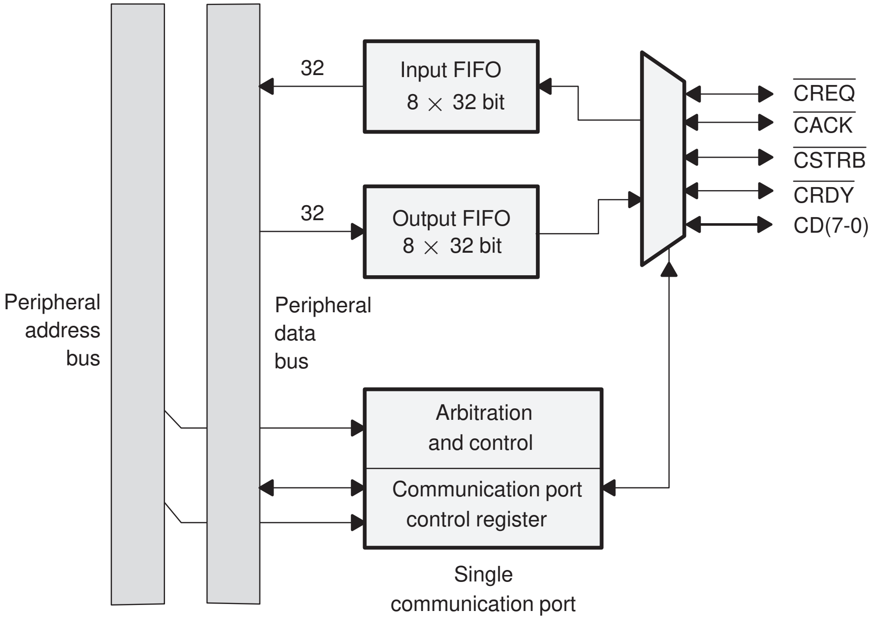
In a typical transfer, the DMA coprocessor or the CPU first writes to the output FIFO. Next, the communication port sends a request signal to the destination processor, which responds with an acknowledge. The communication port then transfers the word as four successive bytes. The destination processor receives the word in its input FIFO, where the destination DMA or CPU can read the contents. Note that the input and output FIFOs provide a 16-word by 32-bit buffer between the communication ports.

Features of the 'C4x communication ports include:

- ❑ Up to 24 Mbytes/s bidirectional interface on each communication port for high-speed and low-cost parallel-processor interface
- ❑ Eight-word-deep input FIFO and 8-word-deep output FIFO buffer
- ❑ Automatic arbitration and handshaking for direct processor-to-processor connection

Figure 6–3 shows a block diagram of the 'C4x communication ports.

Figure 6–3. TMS320C4x Communication Ports



6.6 TMS320C4x DMA Coprocessor

With as many as six communication ports and two external buses, the ’C4x has an I/O capability of as much as 384 Mbytes/s. To service this tremendous speed, the ’C4x has a 6- or 12-channel DMA. The DMA operates independently of the CPU and has dedicated address and data buses to avoid bus conflicts.

The DMA is programmed to transfer data from any memory location to any other memory location (communication ports are memory mapped). The DMA can begin a task based on CPU or external interrupts and can interrupt the CPU at the completion of a task. The DMA also includes a link pointer register that allows the DMA to program its next task without CPU intervention.

Since each communication port has transmit and receive capability, 12 DMA channels are needed if all six communication ports are being used in a bidirectional mode. The DMA has a split-mode operation dedicated to this function, allowing the DMA to service the 12 input and output FIFOs in the communication ports.

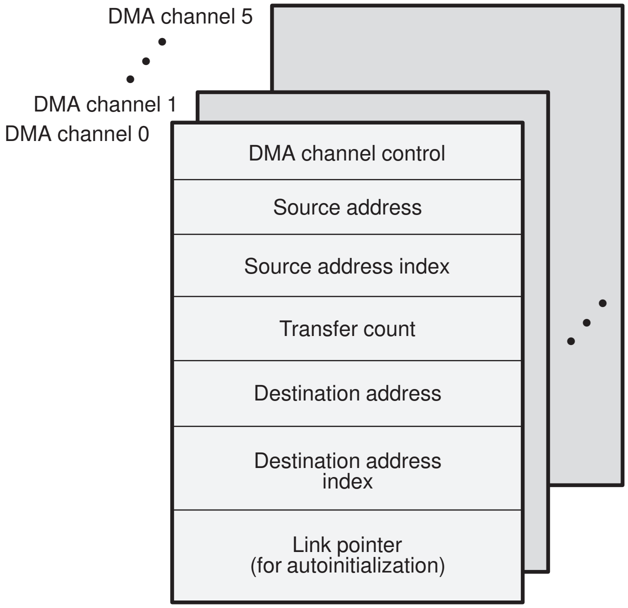
In the event that both the CPU and DMA access the same resource, priorities can be assigned to resolve the conflict. Priority can be assigned to the CPU, the DMA, or mixed, where the CPU gets the first access followed by the DMA.

The ’C4x DMA coprocessor features:

- Concurrent I/O to maximize sustained CPU performance
- Autoinitialization
- Up to 6 or 12 DMA channels for parallel data transfers
  - Data transfers to and from anywhere in memory
  - Three operations per cycle
    - 32-bit data transfer
    - Address register update
    - Transfer counter update
  - Performance of 90 MOPS

Figure 6–4 shows a block diagram of the ’C4x DMA coprocessor.

Figure 6–4. TMS320C4x DMA Coprocessor



6.7 TMS320C40

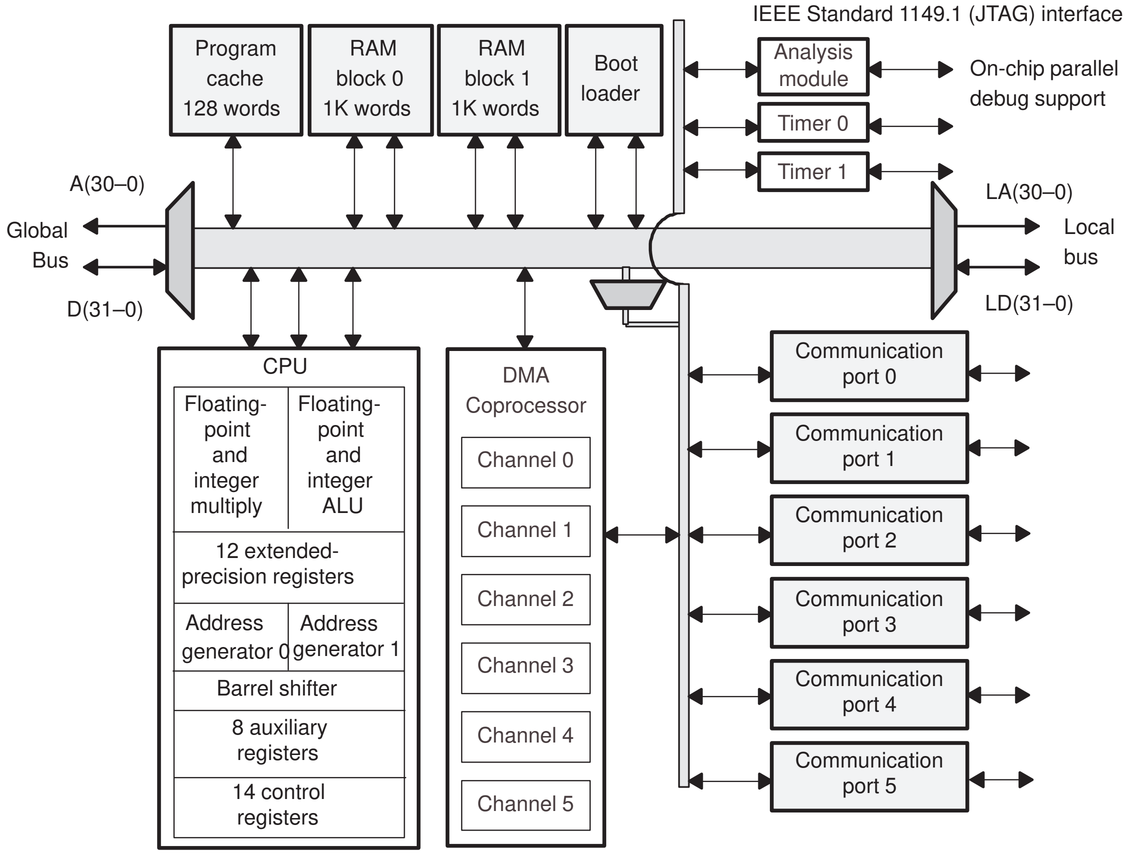
The ’C40 is the original member of the ’C4x family. It features a CPU that can deliver up to 30 MIPS and 60 MFLOPS with a maximum I/O bandwidth of 384 Mbytes/s. The ’C40 has 2K words of on-chip RAM, 128 words of program cache, and a bootloader. Two external buses provide an address reach of 4G of unified memory space. The ’C40 is available in a 325-pin PGA.

Features of the ’C40 include:

❑ 33-and 50-ns instruction cycle times
❑ 4G-word external address reach
❑ Optimizing ANSI C compiler
❑ IEEE floating-point conversion for ease of use
❑ 6- or 12-channel on-chip DMA
❑ Six communication ports
❑ Accepts source code from ’C3x

Figure 6–5 shows a block diagram of the ’C40.

Figure 6–5. TMS320C40 Block Diagram



6.8 TMS320C44

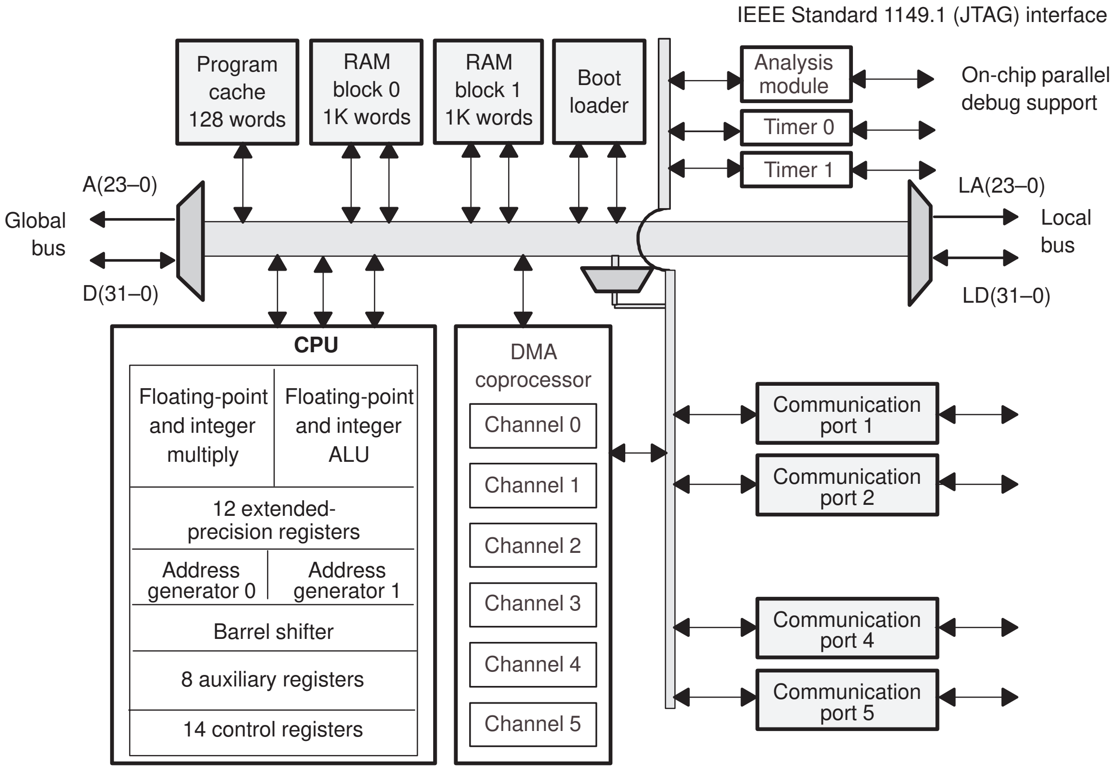
The ’C44 is a lower-cost version of the ’C40, used for parallel-processing applications that are more price-sensitive. The ’C44 features four communication ports and has an external address reach of 32M words over two external buses. To further reduce cost, the ’C44 comes in a 304-pin PQFP. The TMS320C44 can deliver up to 30-MIPS/60-MFLOPS performance with a maximum I/O bandwidth of 336 Mbytes/s. The ’C44 is source-code compatible with the ’C40.

Features of the ’C44 include:

❏ 33- and 40-ns instruction cycle times
❏ Idle mode for reduced power consumption
❏ Optimizing ANSI C compiler
❏ IEEE floating-point conversion for ease of use
❏ 6- or 12-channel on-chip DMA
❏ 304-pin PQFP/388 BGA
❏ 32M word external address reach
❏ Four communication ports

Figure 6–6 shows a block diagram of the ’C44.

Figure 6–6. TMS320C44 Block Diagram



6.9 Tools for TMS320C4x

For information about the code generation and debugging tools available for the ’C4x devices, see Chapter 13, *Code Generation Tools*, and Chapter 14, *System Integration and Debugging Tools*. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

Chapter 7

TMS320C5x Devices

The TMS320C5x generation is TI's most widely used fixed-point DSP. The 'C5x devices offer performance of 20-50 MIPS with power consumption reduced to 2.35 mA/MIPS for typical applications. The 3-V versions maintain 40-MIPS performance and reduce power consumption to 1.15 mA/MIPS.

Topic **Page**

7.1 TMS320C5x Introduction

A programmable software wait-state generator has been added to the ’C5x allowing interfaces with up to seven wait states. Wait states are programmed on 8K-word boundaries allowing the designer to optimize memory configuration for performance and cost.

The standard serial port operates at up to 12.5 Mbps. This is a synchronous double-buffered serial port with independent transmit and receive sections.

The time division multiplexed (TDM) serial port has all the same features as the standard serial port with the addition of time division multiplexing,. This will make it well suited for interprocessor communication in multiprocessor DSP systems.

The buffered serial port (BSP) comprises a full-duplex, double-buffered serial port interface and an autobuffering unit. The BSP operates at up to 40 Mbps with no CPU intervention. The serial port interface block of the BSP is an enhanced version of the standard serial port interface.

The host port interface (HPI) is an 8-bit parallel port used to interface a host device to the ’C5x. HPI memory is a 2K by 16-bit single-access RAM (SARAM) block that can also be used as general-purpose on-chip or program SARAM.

Multiple on-chip phase-locked-loop (PLL) options are available, depending on which ’C5x is chosen. The PLL allows designers to maintain CPU speed but use a slower external clock frequency. This reduces power and electromagnetic emissions.

For systems requiring significant off-chip resources, the ’C5x family addresses 64K by 16-bits in program, data, and I/O spaces. Each uses its own select pin.

7.2 TMS320C5x Key Features

Following are the key features of the TMS320C5x generation. When only certain devices have a particular feature, they are named in parentheses following the feature's description.

❑ CPU

20-ns, single-cycle instruction execution time, 50 MIPS

25-ns, single-cycle instruction execution time, 40 MIPS

35-ns, single-cycle instruction execution time, 28.6 MIPS

Single-cycle multiply/accumulate (MAC) for program code

Single-cycle/single-word repeats and block repeats for program code

Block memory moves for good program/data management

Four-level-deep executable pipeline to handle delayed branch, call, and return instructions

Indexed addressing mode

Bit-reversed/indexed addressing mode to facilitate Fast Fourier Transforms (FFTs)

Power-down modes

32-bit ALU, 32-bit accumulator, and 32-bit accumulator buffer

Eight auxiliary registers with a dedicated ALU for indirect addressing

16-bit parallel logic unit (PLU) for fast bit manipulation

16 by 16-bit parallel multiplier with a 32-bit product capacity

0- to 16-bit right and left barrel shifters

64-bit incremental data shifter

Two indirectly addressed circular data buffers for circular addressing

❑ Peripherals

Eight-level hardware stack

11 context-switch registers to shadow the contents of strategic, CPU-controlled registers during interrupts

Full-duplex, synchronous serial port, which directly interfaces to the codec

Time-division-multiplexed (TDM) serial port ('C50/C51/C53)

Buffered serial port (BSP) (’LC56, ’LC57, ’BC57S, ’C57S, ’LC57S)

Interval timer with period and control registers for software stops, starts, and resets

Concurrent external DMA performance, using extended holds

On-chip clock generator

Various phase-locked-loop (PLL) options for reduced EMI and system power dissipation

PLL clock generator (×1, ×2, ×3, ×4, ×5, ×9) (’LC56, ’LC57, ’BC57S, ’C57S)

IEEE 1149.1-standard (JTAG) scan-path test bus for system test and emulation

Host-port interface (’LC57, ’BC57S)

- Memory

  10K 16-bit words of on-chip program/data RAM (’C50)
  2K 16-bit words of on-chip program/data RAM (’C51)
  1K 16-bit words of on-chip program/data RAM (’C52)
  4K 16-bit words of on-chip program/data RAM (’C53, ’C53S)
  7K 16-bit words of on-chip program/data RAM (’LC56, ’LC57, ’BC57S)
  2K 16-bit words of single-cycle on-chip boot ROM (’C50, ’BC57S)
  8K 16-bit words of single-cycle on-chip program ROM (’C51)
  4K 16- bit words of on-chip program ROM (’C52)
  16K 16-bit words of on-chip program ROM (’C53, ’C53S)
  32K 16 ROM (’LC56, ’LC57)
  Bit words of on-chip program ROM

- Memory interfaces

  16 programmable software wait-state generators for program, data, and I/O memories

  224K-word 16-bit maximum addressable external memory space (64K-word program, 64K-word data, 64K-word I/O, and 32K-word global)

- Source code upward compatible with all ’C1x, ’C2x, and ’C2xx devices
- Single 5-V or 3.3-V supply
- Static CMOS technology
- Packaging:

  100-pin TQFP (’C51, ’C52, ’C53S, ’LC56)
  132-pin QFP (’C50, ’C51, ’C53)
  100-pin QFP (’C52)
  144-pin QFP (’BC57S)
  128-pin TQFP (’LC57)

7.3 TMS320LC57/BC57S Host Port Interface

The host port interface (HPI) is an 8-bit parallel port available on the ’LC57 and the ’BC57S. The HPI provides a glueless interface to standard microprocessors as well as to other TI devices. The HPI appears as a 2K-word block of shared memory that is available in either a FIFO or standard random-access configuration. Most importantly, the HPI has the ability to maintain its high level of functionality as the interface between an external CPU and the ’C57, even while the ’C57 is idle or in reset. This significantly reduces system power consumption by offloading standard I/O tasks from the DSP.

Features of the HPI include:

❑ Byte-wide register addressability
❑ 8-bit parallel port
❑ High-speed back-to-back accesses
❑ Dedicated bus to 2K 16-bit words of SARAM
❑ Shared-access mode (SAM)

  Normal mode of operation
  Allows DSP and host to have HPI memory access
  Asynchronous host accesses resynchronized internally
  45.7 Mbps at 57 MHz
  64.0 Mbps at 80 MHz

❑ Host-only mode (HOM)

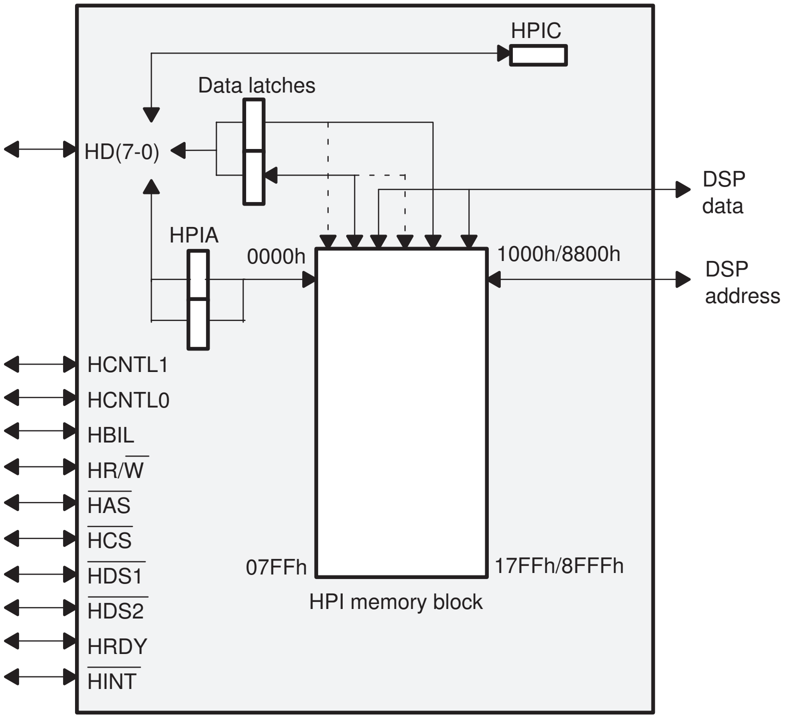
  Allows host to access HPI memory while ’C57 is in IDLE2 or in reset mode

  5-µA power dissipation (IDLE2)

  160 Mbps, independent of clock

Figure 7–1 shows a block diagram of the ’LC57/BC57S HPI.

Figure 7–1. TMS320LC57/BC57S Host Port Interface

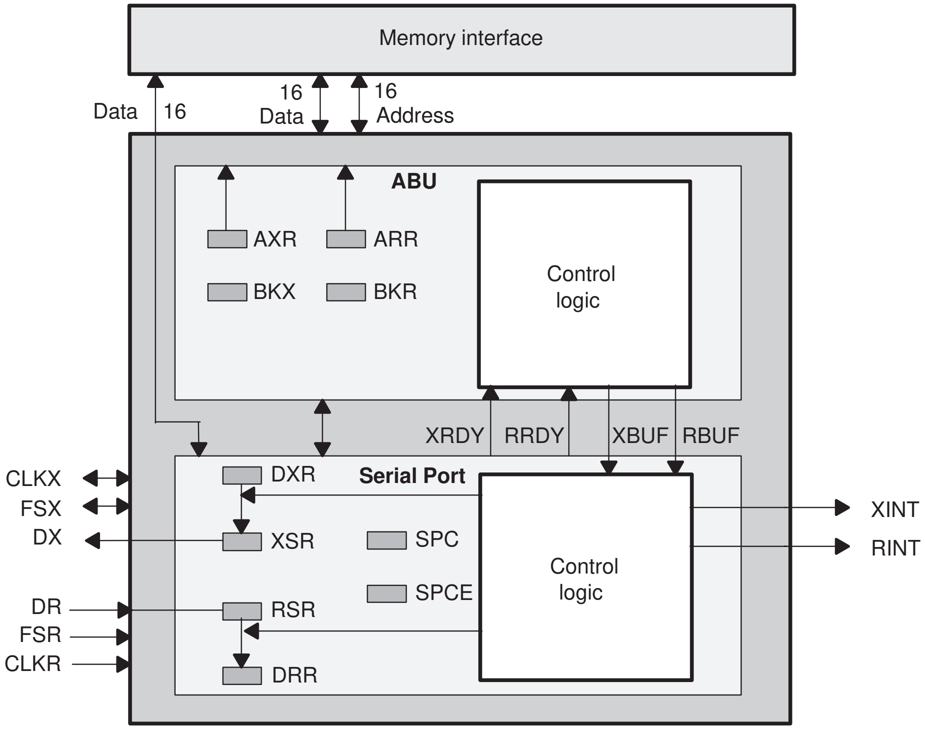


7.4 TMS320LC56/LC57/BC57S Buffered Serial Port

The buffered serial port (BSP) provides a no-overhead mechanism to interface serially with codecs, ADCs, and other peripherals. The BSP supports 8-, 10-, 12-, and 16-bit serial data packets and uses a 2K-byte buffer to hold input or output data. Downward code compatible with the standard serial ports, the BSP is designed to be completely flexible and programmable. The BSP has the ability to simultaneously receive data from and transmit data to a programmable on-chip buffer through a dedicated memory bus, freeing the CPU to execute other tasks without memory bus contention.

Figure 7–2 shows a block diagram of the BSP.

Figure 7–2. TMS320LC56/LC57/BC57S Buffered Serial Port



7.5 TMS320C5x Multiplier/ALU Features

The performance and parallelism of the ’C5x architecture become evident by studying the multiplier and accumulator sections of the device. The on-chip multiplier performs a 16-bit by 16-bit multiply with a 32-bit product in a single cycle.

To double the performance, the ’C5x devices can fetch two data operands when executing the multiply and accumulate (MAC) instruction. Scaling and fractional math shifts occur in parallel with all arithmetic operations.

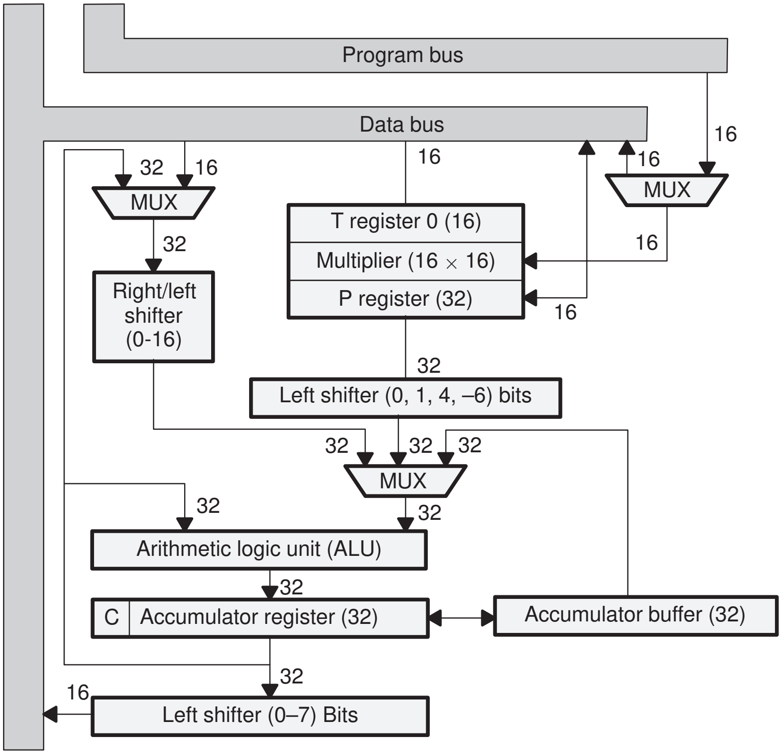
The ’C5x devices are the first to include an accumulator buffer. The accumulator buffer provides quick comparisons to the accumulator contents to determine the minimum or maximum in a table of values, a common operation for both general-purpose and DSP algorithms. This is very useful in implementing Viterbi-decoding algorithms. Additionally, the accumulator buffer allows quick temporary storage of a commonly referenced value, eliminating the extra cycles it takes to store the value in memory. It also allows up to 65 bit shifts among accumulator, accumulator buffer, and carry.

Features of the ’C5x multiplier/ALU include:

- ❑ Single-cycle 16-bit x 16-bit multiply, yielding a 32-bit product
- ❑ Ability to access program and two data operands simultaneously
- ❑ Simultaneous ALU and multiplier operations performed by the CPU
- ❑ Zero-overhead shift registers
- ❑ Accumulator buffer with path back into ALU

Figure 7–3 shows a block diagram of the ’C5x multiplier/ALU.

Figure 7–3. TMS320C5x Multiplier/ALU Features



7.6 TMS320C5x Parallel Logic Unit

The parallel logic unit (PLU) performs logical operations without corrupting the ALU contents. This allows data to be checked and decisions made without the overhead of storing and restoring the ALU content.

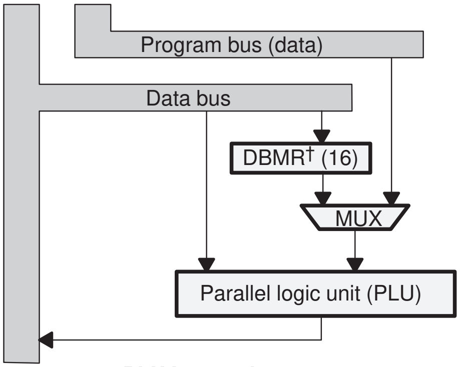
The PLU supports AND, OR, XOR, and compare functions. These functions are commonly used to set, clear, and toggle bits within a data-memory location. To make these functions easy to use, dedicated instructions perform them, using either immediate or register data to compare with the data-memory location.

Features of the ’C5x PLU are:

- Performs logical operations without disturbing ALU arithmetic registers
- Sets, clears, and toggles any number or combinations of bits in data memory
- Compares bits individually or collectively
- Supports long-immediate and register values

Figure 7–4 shows a block diagram of the ’C5x PLU.

Figure 7–4. TMS320C5x Parallel Logic Unit



PLU Instructions

Mnemonic	Description
APL #	AND DBMR†/constant with data memory
CPL #	Compare DBMR/constant with data memory
OPL #	OR DBMR/constant with data memory
XPL #	XOR DBMR/constant with data memory
SPLK #	Store Long-immediate to data memory

† DBMR: Dynamic bit-manipulation register

7.7 TMS320C5x Interrupts

The ’C5x supports four external interrupts, two serial port interrupts, one timer interrupt, and a trap instruction.

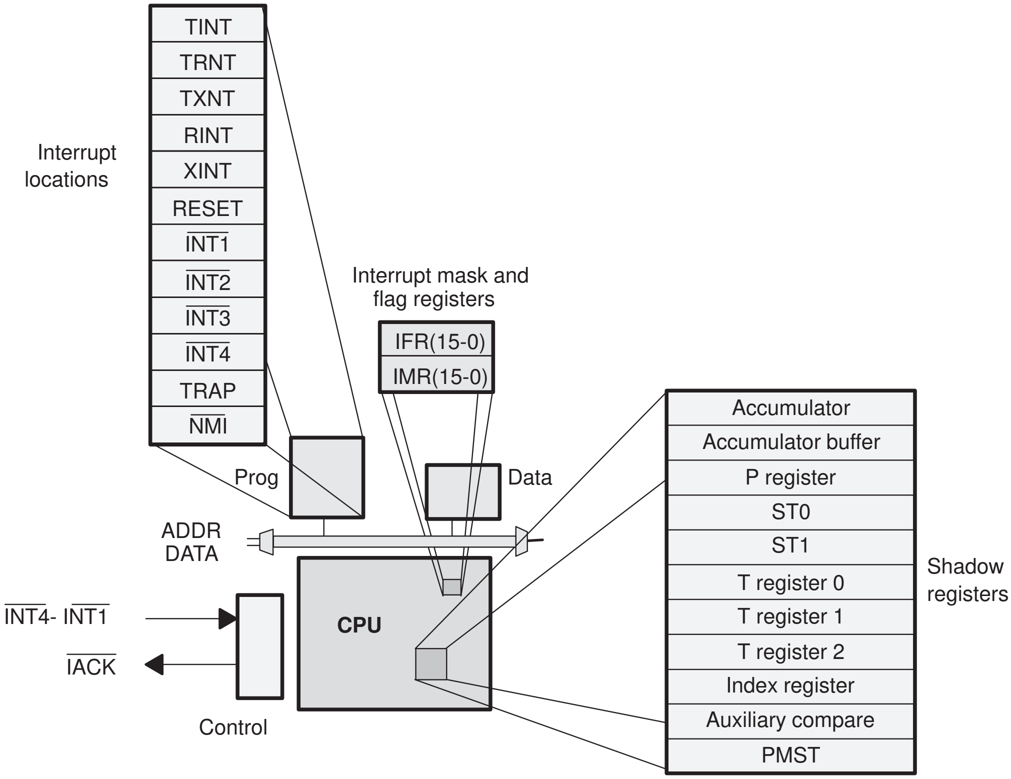
To speed up interrupt service routines (ISRs), the key registers are shadowed with a 1-level-deep stack and are saved in a single cycle. This saves 22 cycles when an interrupt occurs by eliminating the need to push and pop these 11 registers.

The ’C5x interrupt features include:

- ❑ Interrupt sources: four external, two serial port, one timer, one trap
- ❑ One-level-deep stack on strategic registers for zero-overhead context switching
- ❑ Program-accessible interrupt flag register (IFR) for software interrupt polling
- ❑ Automatic global interrupt enable on return from interrupt service routine
- ❑ Relocatable interrupt vectors

Figure 7–5 shows a block diagram of the ’C5x interrupts.

Figure 7–5. TMS320C5x Interrupts



7.8 TMS320C5x Circular Addressing

Circular addressing is a method to address aging data sets for common DSP algorithms. For example, after performing the first set of calculations on a finite impulse response (FIR) filter data set, a new data value must be brought in and the oldest value discarded. Data move (DMOV) allows you to do this by treating data in a circular fashion, with the oldest and newest values located side by side. At the end of each calculation, a new data value is placed in memory at the location of the oldest value. You update the circular buffer start address to correspond to the newest value and the circular buffer end address to correspond to the oldest value.

Circular addressing features are:

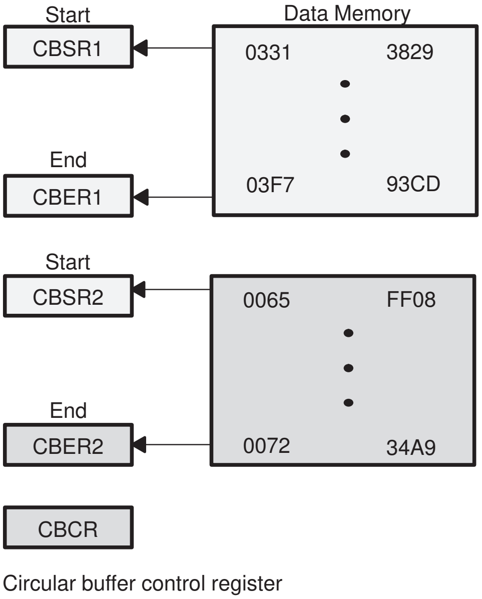
❑ Two circular-addressing data buffers

❑ Memory-mapped auxiliary registers to indirectly address each circular buffer

❑ Effective use of data-memory space to increase performance

Figure 7–6 shows a block diagram of the ’C5x circular addressing.

Figure 7–6. TMS320C5x Circular Addressing

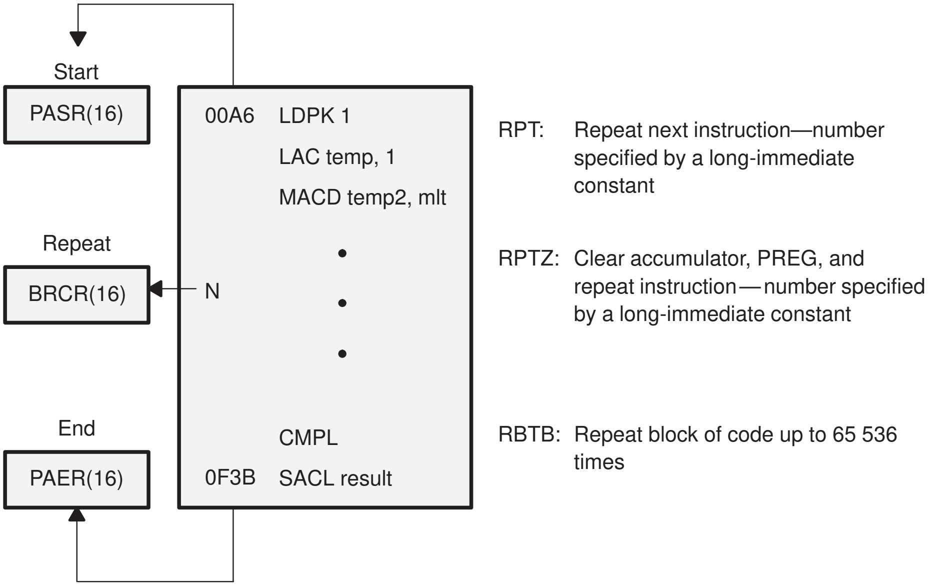


7.9 Repeat and Block Instructions

The ’C25 was the first device to offer a zero-overhead looping instruction, the RPT instruction. The ’C5x expands this capability with the RPTZ and RPTB instructions. The RPTZ command is a logical extension of the RPT command that also clears the accumulator and product registers. This saves two cycles in typical routines. The repeat block instruction (RPTB) allows you to repeat a block of code up to 64K times without any overhead required to branch to the beginning of the code segment. This allows you to benefit from a faster instruction cycle (due to the pipeline) without paying a penalty for program discontinuity.

Figure 7–7 shows a block diagram of the repeat and block instructions.

Figure 7–7. Repeat and Block Instructions



Start
PASR(16)
Repeat
BRCR(16)
N
End
PAER(16)
00A6
LDPK 1
LAC temp, 1
MACD temp2, mlt
CMPL
0F3B
SACL result
RPT: Repeat next instruction—number specified by a long-immediate constant
RPTZ: Clear accumulator, PREG, and repeat instruction — number specified by a long-immediate constant
RBTB: Repeat block of code up to 65 536 times

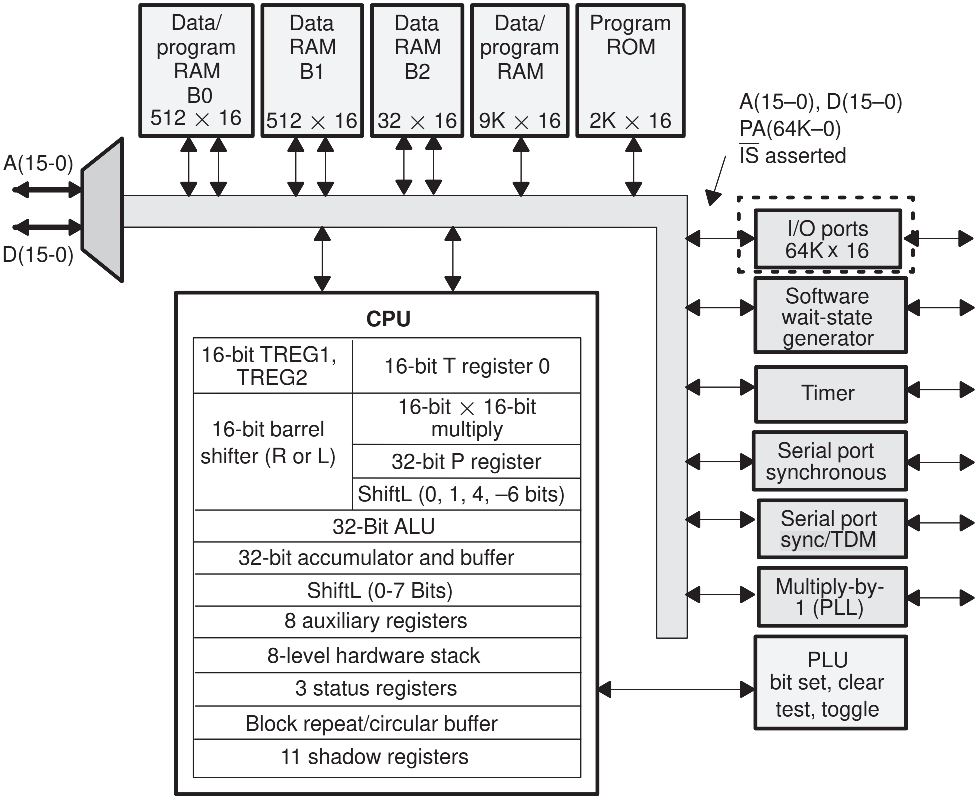
7.10 TMS320C50 and TMS320LC50

The ’C50 and ’LC50 feature large, on-chip RAM blocks, which are ideal for form-function emulation. The ’LC50 is the low-voltage (3.3 V) version of the ’C50 device and significantly reduces power consumption. The ’C50 is a highly integrated DSP, offering a complete system on a single chip. With a boot ROM and 10K 16-bit words of on-chip RAM, an entire DSP can be integrated into a 132-pin QFP. By integrating the memory on-chip, you reduce both power and board space. Integrating the off-chip memory can be especially important in high-performance systems to eliminate expensive high-speed static RAM (SRAM).

The ’C50 offers 25- and 35-,-ns instruction cycle times and accepts source code from the ’C1x, ’C2x, and ’C2xx generations. It has a full-duplex synchronous serial port, time division multiplexed (TDM) communication port, and preprogrammed ROM bootloader. Other features of the ’C50 include a 192K-word external address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control.

Figure 7–8 shows a block diagram of the ’C50/LC50.

Figure 7–8. TMS320C50/LC50 Block Diagram



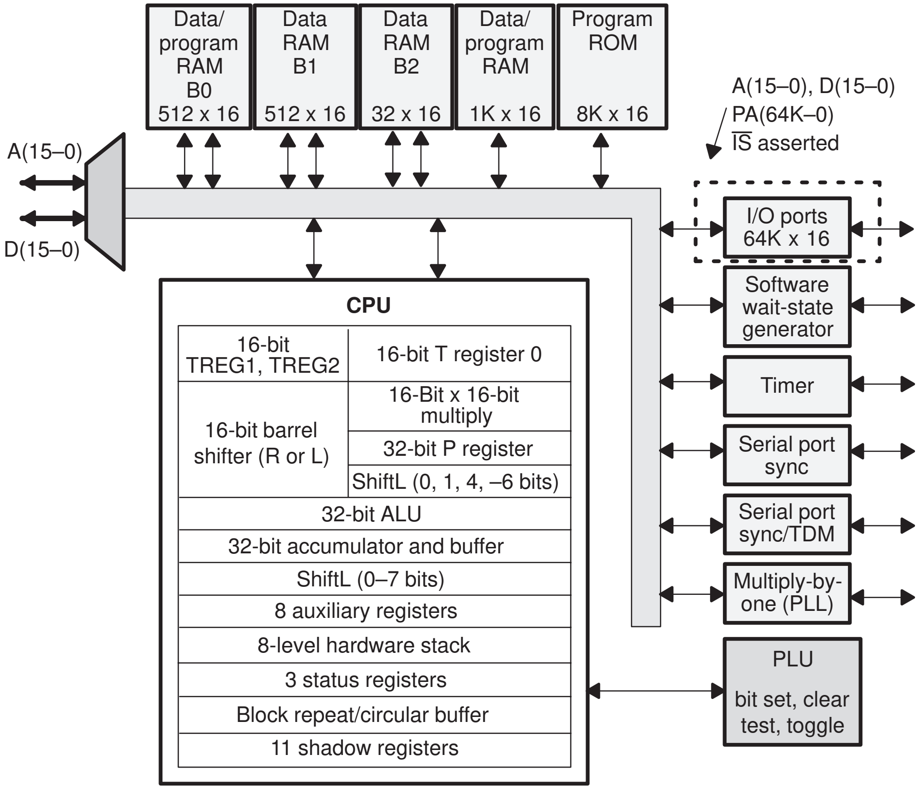
7.11 TMS320C51 and TMS320LC51

The ’C51 features a user-maskable ROM for implementing information that was preprogrammed by TI. It is object-code and pin compatible with the ’C50 device. The ’LC51 is the low-voltage version of the ’C51. In the ’C51, the ’C50’s 9K-word block of RAM is replaced by 8K 16-bit words of on-chip ROM. This provides a considerable advantage in cost and performance for users who require large amounts of on-chip program space. With this ROM and 2K 16-bit words of RAM, sophisticated DSP algorithms can fit on a single device. This device is available in a 132-pin QFP and a 100-pin TQFP measuring only 14 by 14 by 1.4 mm, for designs that require both small board area and reduced height. The ’C51 is available in 3.3 V or 5 V and 35-, 25-, or 20-ns cycle times. It has communication ports similar to those of the ’C50, and has an optional preprogrammed ROM bootloader.

The ’C51 accepts source code from the ’C1x, ’C2x, and ’C2xx generations. It also has a full-duplex synchronous serial port and a TDM serial port. Other features of the ’C51 include a 192K-word external-address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control.

Figure 7–9 shows a block diagram of the ’C51/LC51.

Figure 7–9. TMS320C51/LC51 Block Diagram



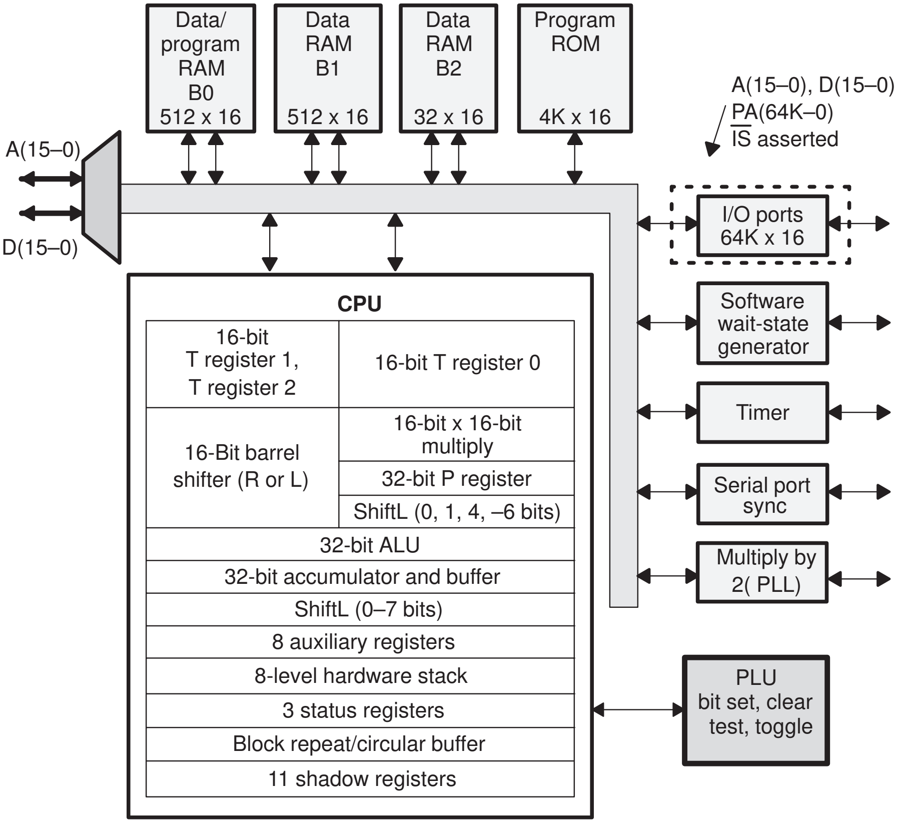
7.12 TMS320C52 and TMS320LC52

The ’C52 is a cost-effective implementation of the ’C5x architecture. The ’LC52 is the low-voltage version. Traditionally, devices in the same price range have offered 10-MIPS performance. The ’C52 provides twice the performance at 20 MIPS and has room to grow into a 50-MIPS version. The 4K 16-bit words of ROM and 1K 16-bit words of RAM can be configured with your code or purchased preprogrammed with a bootloader. The ’C52 is available in 3.3 V or 5 V; has 35-, 25-, or 20-ns cycle times; comes in a 100-pin QFP and a 100-pin TQFP; has a full-duplex synchronous serial port; and has an optional preprogrammed ROM boot loader.

The ’C52 accepts source code from the ’C1x, ’C2x, and ’C2xx generations. It also has a 192K-word external address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control.

Figure 7–10 shows a block diagram of the ’C52/LC52.

Figure 7–10. TMS320C52/LC52 Block Diagram



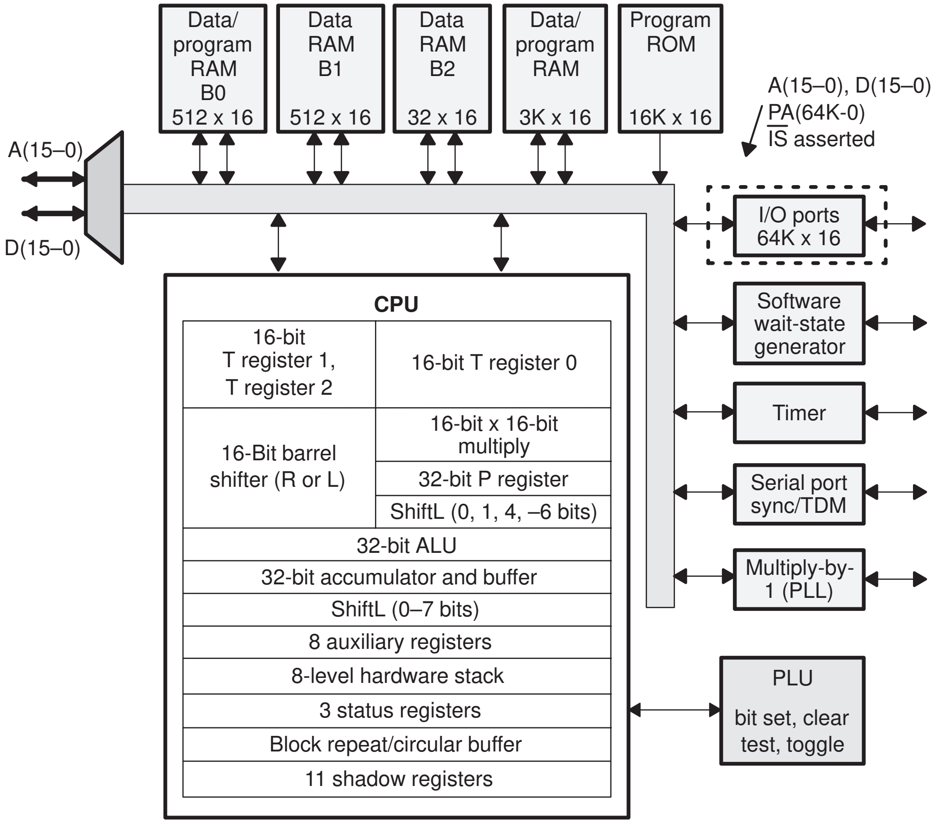
7.13 TMS320C53 and TMS320LC53

The ’C53 and ’LC53 provide greater integration of on-chip ROM than the ’C51. The advantage of bringing memory on chip reduces power and board-space requirements; on-chip memory also provides a considerable advantage in cost and performance. The 16K 16-bit words of ROM and 4K 16-bit words of on-chip RAM can be configured with your code or purchased preprogrammed with a bootloader. The ’C53 is available in 3.3 V or 5 V; has 50-, 35-, or 25-ns cycle times; comes in a 132-pin PQFP; and has a full-duplex synchronous serial port, and a TDM communications port.

The ’C53 accepts source code from the ’C1x, ’C2x, and ’C2xx generations. It also has a 192K-word external address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control.

Figure 7–11 shows a block diagram of the ’C53/LC53.

Figure 7–11. TMS320C53/LC53 Block Diagram



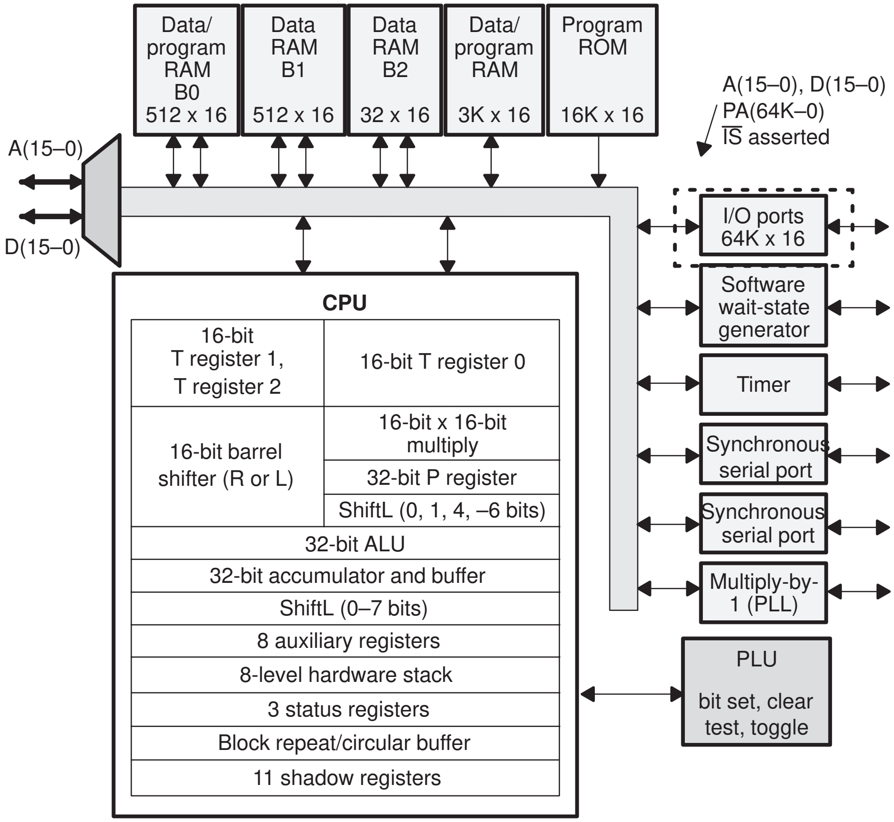
7.14 TMS320C53S and TMS320LC53S

The ’C53S and ’LC53S are similar in most respects to the ’C53 and LC53. To accommodate the integration capabilities of the ’C53 into a 100-pin TQFP, some features were removed from the standard ’C53. The ’C53S offers two standard serial ports instead of the TDM and standard serial ports in the ’C53. The ’C53S has reduced capability of the on-chip analysis block and has no boundary scan.

The ’C53S offers 25-and 35-ns instruction cycle times and accepts source code from the ’C1x, ’C2x, and ’C2xx generations. Other features of the ’C53s include a 192K-word external address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control. The ’C53S also offers an optional preprogrammed ROM boot loader.

Figure 7–12 shows a block diagram of the ’C53S/LC53S.

Figure 7–12. TMS320C53S/LC53S Block Diagram



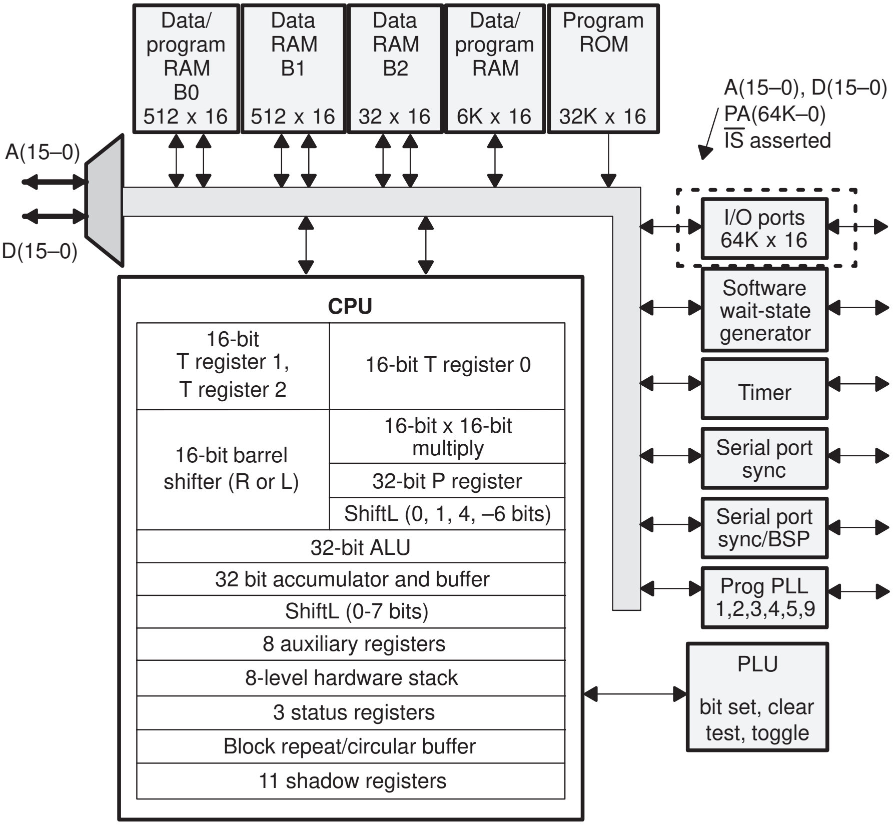
7.15 TMS320LC56

The ’LC56 provides greater integration of on-chip ROM than the ’C53. With 32K 16-bit words of on-chip ROM and 7K 16-bit words of on-chip RAM, the ’LC56 can accommodate large program and data spaces on-chip, thereby minimizing off-chip accesses. The communications ports comprise a full-duplex synchronous serial port and a very fast BSP with a dedicated bus. The BSP is capable of 40 Mbps at a 25-ns instruction cycle time. The ’LC56 is optimized for high-performance, low-power applications; as a result, it operates at 3.3 V only.

The ’LC56 offers 25- and 35-ns instruction cycle times and accepts source code from the ’C1x, ’C2x, and ’C2xx generations. Other features of the ’LC56 include a 192K-word external address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control, a boot ROM option, and programmable PLL. The ’LC56 comes in a 100-pin TQFP.

Figure 7–13 shows a block diagram of the ’LC56.

Figure 7–13. TMS320LC56 Block Diagram



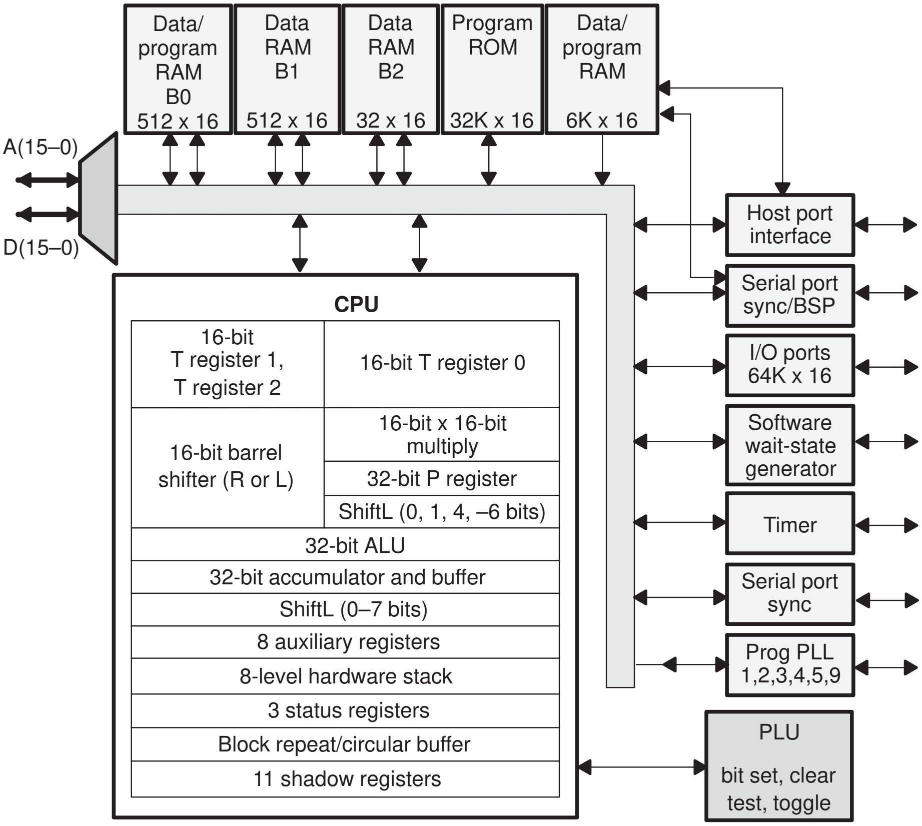
7.16 TMS320LC57

The ’LC57 incorporates the same amount of on-chip memory as the ’LC56 and offers a high-throughput buffered serial port. In addition, the ’LC57 provides an 8-bit wide host port interface (HPI), which can be used to communicate with other ’LC57 devices or embedded microprocessors. The ’LC57, like the ’LC56, is capable of 25-ns instruction cycle time at 3.3 V.

The ’LC57 offers 25- and 35-ns instruction cycle times and accepts source code from the ’C1x, ’C2x, and ’C2xx generations. Other features of the ’LC57 include a boot load through the HPI or a standard serial port, a 192K-word external address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control. The ’LC57 also offers a boot ROM option, a full-duplex synchronous serial port, programmable PLL, a BSP with a dedicated bus, and an HPI with a dedicated bus. The ’LC57 is packaged in a 128-pin TQFP.

Figure 7–14 shows a block diagram of the ’LC57.

Figure 7–14. TMS320LC57 Block Diagram



7.17 TMS320BC57S

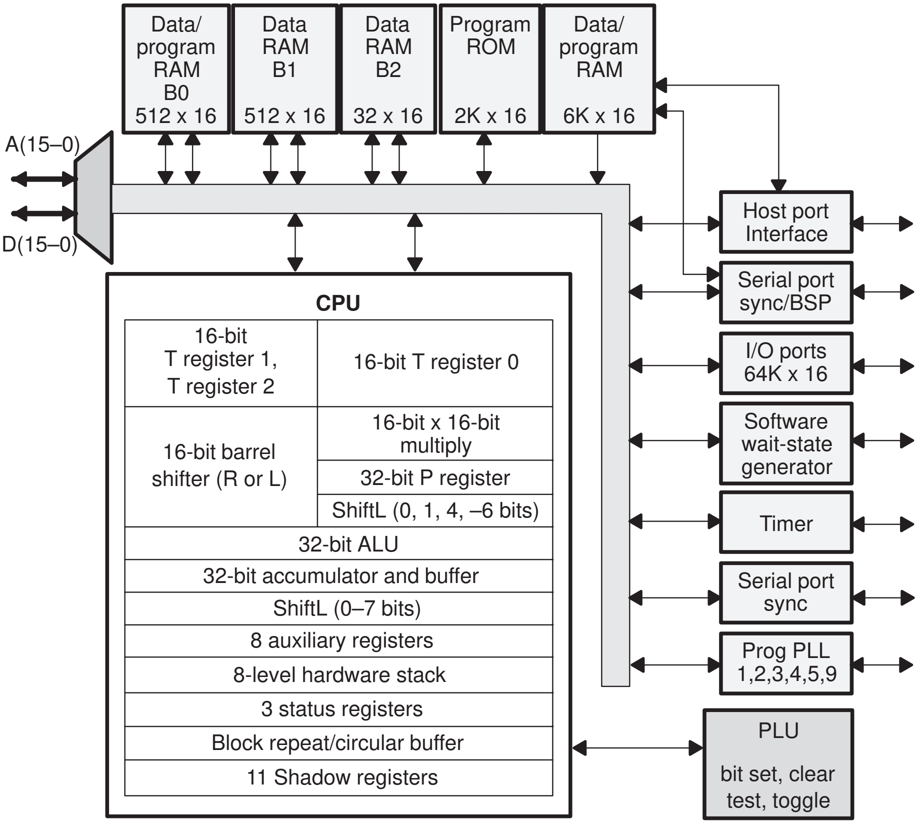
The ’BC57S takes advantage of the same peripherals that are on the ’LC57 to make the ’BC57S a cost-effective embedded data I/O engine. In order to provide a broad-based appeal, the ’BC57S differs from the ’LC57 in four ways:

- ❑ The 32K 16-bit words of ROM space has been replaced with a boot ROM.
- ❑ The ’BC57S can operate at 5 V.
- ❑ The 144-pin package of the ’BC57S has a wider lead pitch than the package of the ’LC57.
- ❑ The ’BC57S is lower in cost.

The ’BC57S offers 25 and 35-ns instruction cycle times at 5 V; accepts source code from the ’C1x, ’C2x, and ’C2xx generations; and has 7K 16-bit words of RAM. Other features include a 192K-word external address reach, an ANSI C compiler, and an IEEE 1149.1-standard (JTAG) emulator control. The communication ports include a full-duplex synchronous serial port, an HPI with a dedicated bus, and a BSP with a dedicated bus. The ’LC57S is packaged in a 144-pin TQFP.

Figure 7–15 shows a block diagram of the ’BC57S.

Figure 7–15. TMS320BC57S Block Diagram



7.18 Tools for TMS320C5x

For information about the code generation and debugging tools available for the ’C5x devices see Chapter 13, *Code Generation Tools*, and Chapter 14, *System Integration and Debugging Tools*. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

Chapter 8

TMS320C54x Devices

The TMS320C54x devices are fixed-point digital signal processors (DSPs) in the TMS320 family. The ’54x meets the specific needs of real-time embedded applications, such as telecommunications. The ’54x central processing unit (CPU), with its modified Harvard architecture, features minimized power consumption and a high degree of parallelism. Also, the versatile addressing modes and instruction set improve the overall system performance.

The ’54x devices offer these advantages:

❑ Enhanced Harvard architecture built around one program and three data buses for increased performance and versatility

❑ Advanced CPU design with a high degree of parallelism and application-specific hardware logic for increased performance

❑ A highly specialized instruction set for faster algorithms and for optimized high-level language operation

❑ Modular architecture design for fast development of spinoff devices

❑ Advanced IC processing technology for increased performance and low power consumption

❑ Low power consumption and increased radiation hardness because of new static design techniques

Topic **Page**

8.1 TMS320C54x Introduction

With two 40-bit accumulators, the ’C54x is an accumulator-based DSP. A 40-bit adder dedicated to multiply and accumulate (MAC) operations has a separate 40-bit arithmetic logic unit (ALU) that feeds the accumulators. The ALU and two accumulators support eight special parallel instructions that execute in one cycle. The ALU also features a dual 16-bit configuration that enables dual single-cycle operations. The 40-bit adder, at the output of the multiplier, allows unpipelined MAC operations as well as dual addition and multiplication operations in parallel. The multiplier performs 17 by17-bit multiplies to allow 16-bit signed or unsigned multiplication, with rounding and saturation control in one cycle. Single-cycle normalization and exponential encoding support floating-point arithmetic.

The ’C54x instruction set complements the parallelism of the architecture. It supports many two- and three-operand instructions, as well as some 32-bit operands. Eight individually addressable auxiliary registers and a software stack aid a C compiler’s efficiency.

The standard serial port operates at up to 12.5 Mbps. This is a synchronous double-buffered serial port with independent transmit and receive sections. The time division multiplexed (TDM) serial port has all of the same features as the standard serial port with the addition of time division multiplexing, making it well suited for interprocessor communication in multiprocessor DSP systems.

The buffered serial port (BSP) comprises a full-duplex, double-buffered serial port interface and an auto-buffering unit. The serial port interface block of the BSP is an enhanced version of the standard serial port interface.

The host port interface (HPI) is an 8-bit parallel port used to interface a host processor to the TMS320C54x. The HPI memory is a 2K by 6-bit single-access RAM block that can also be used as general-purpose on-chip or program single-access RAM (SARAM).

The ’C54x architectural efficiencies, 100 MIPS, and low power dissipation make it an ideal device for a variety of wireless and wireline communications systems. Standard wireless applications like cellular handsets can take advantage of the ’C54x power-down modes, high performance, and large ROM memory spaces for integrating entire algorithms on chip. The high performance and large RAM of the ’C542, ’C548, and ’C549 are useful for reconfigurable systems where a variety of different algorithms must be executed from on-chip memory. Data communications or telephony applications can increase system functionality while lowering system cost by allowing the ’C54x to absorb the existing system functionality of multiple-DSP implementations.

8.2 TMS320C54x CPU Key Features

The ’C54x, ’LC54x, and ’VC54x fixed-point digital signal processors are fabricated with an advanced modified Harvard architecture that has one program memory bus and three data memory buses. The core’s key features include a 17-bit by 17-bit multiplier (16-bit signed or unsigned); a dedicated 40-bit adder for unpipelined MAC (multiply/accumulate) operation; a separate 40-bit ALU for increased parallelism; two 4-bit accumulators; and a compare, select, store unit (Viterbi accelerator). The ’C54x uses a highly specialized dual-operand instruction set that is the basis of the operational flexibility and speed of these DSPs. The ’C54x also includes eight auxiliary registers and a software stack to enable a highly-optimized C compiler. The devices lower power consumption, reduce chip count, and enable system cost savings for communications applications.

The ’C54x CPU key features include:

- ❑ Advanced multibus architecture with three separate 16-bit data buses and one program bus
- ❑ 40-bit ALU, including a 40-bit barrel shifter and two independent 40-bit accumulators
- ❑ 17-bit by 17-bit parallel multiplier coupled to a 40-bit dedicated adder for nonpipelined, single-cycle multiply and accumulate (MAC) operation
- ❑ Compare, select, store unit (CSSU) for the add/compare selection of the Viterbi operator
- ❑ Exponent encoder to compute the exponent of a 40-bit accumulator value in a single cycle
- ❑ Two address generators, including eight auxiliary registers and two auxiliary register arithmetic units

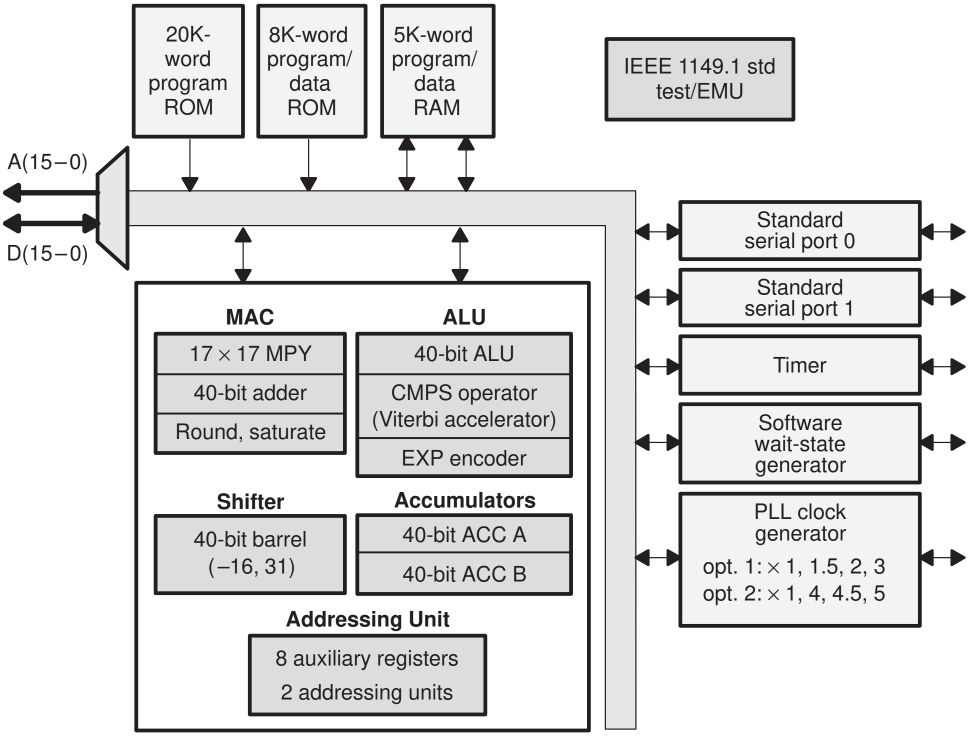
8.3 TMS320C541, TMS320LC541

The ’C541 family consists of advanced performance, low-power digital signal processors optimized for communications applications. Ther low-voltage version is the ’LC541 (3.3 V) . The device offers 28K 16-bit words of ROM and 5K 16-bit words of RAM on chip, two standard serial ports, a timer, a software wait-state generator, and a phase-locked loop (PLL) clock generator all integrated into the 100-pin TQFP, measuring 14 by 14 by 1.4 mm.

The ’C541 also features an integrated Viterbi accelerator; powerful single-cycle instructions (dual-operand, parallel, and conditional instructions); low active-mode power dissipation (less than 35 mW to run VSELP); low-power standby modes; and multiple PLL options. The ’LC541 offers instruction cycle times of 25 ns (40 MIPS) or 20 ns (50 MIPS); the 5-V version of the ’C541 is only available at 25 ns.

Figure 8–1 shows a block diagram of the ’C541.

Figure 8–1. TMS320C541/’LC541 Block Diagram



Note: The ’LC541 is available with one of two different PLL options.
You choose one of the options listed.

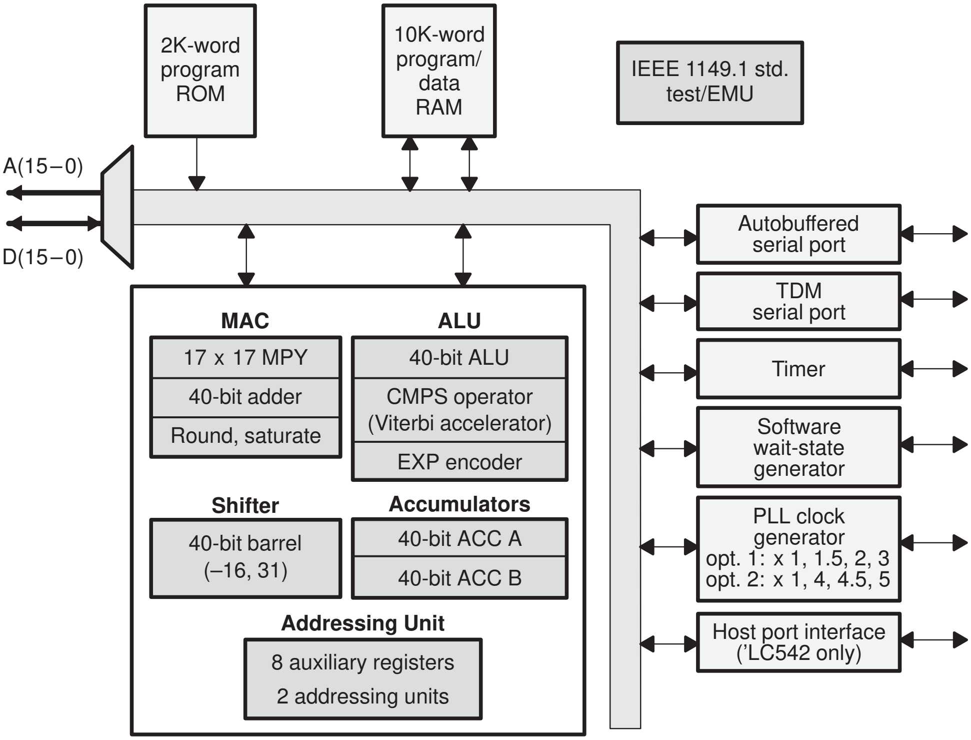
8.4 TMS320C542, TMS320LC542, and TMS320LC543

The ’C542 offers designers a low-cost, low-power, high-performance digital signal processor with 2K 16-bit words of boot ROM and 10K 16-bit words of RAM on chip. The low-voltage version is the ’LC542 (3.3. V). Peripherals include a BSP, an HPI (which interfaces with other DSPs or embedded microprocessors), a timer, and a TDM serial port.

The ’C542 family devices feature an integrated Viterbi accelerator; powerful single-cycle instructions (dual-operand, parallel, and conditional instructions); low active-mode power dissipation (less than 35 mW to run VSELP); and low-power standby modes. These devices offer instruction cycle times of 25 ns (40 MIPS) or 20 ns (50 MIPS); the 5-V version of the ’C542 is available only at 25 ns.

The ’C542 is available in a 144-pin TQFP (20 by 20 by 1.4 mm), while the ’LC542 and ’VC542 are available in a 128- or 144-pin TQFP. The ’LC543 offers a cost reduction by removing the HPI and are available in a 100-pin TQFP, measuring 14 by 14 by 1.4 mm.

Figure 8–2. TMS320C542/LC543 Block Diagram



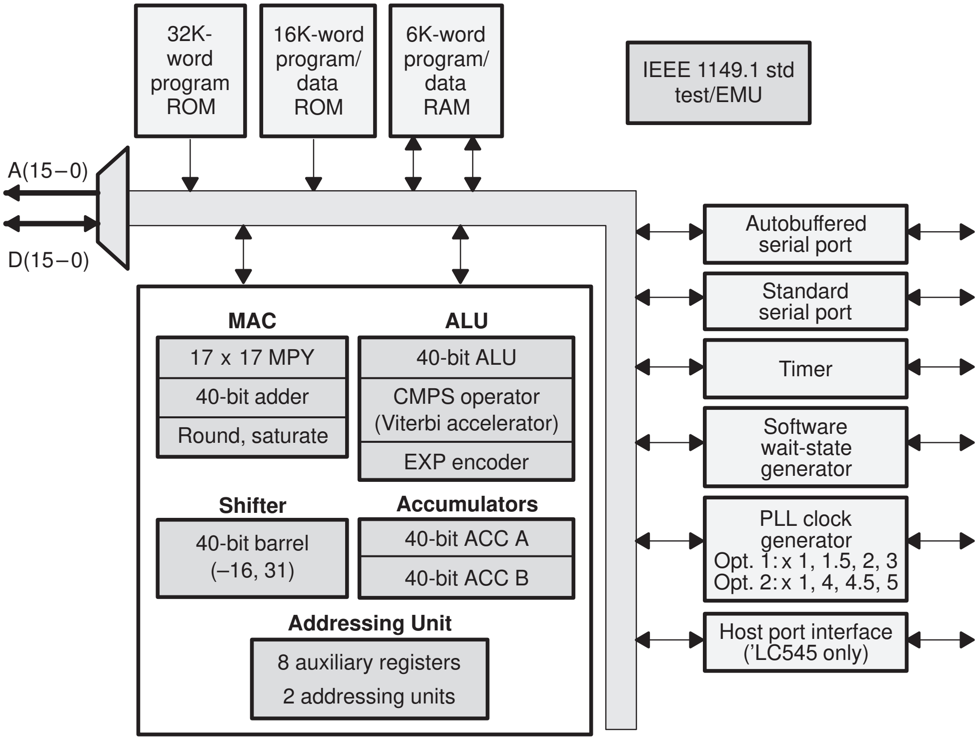
Note: The ’LC542 and ’LC543 are available with one of two different PLL options.
You choose one of the options listed.

8.5 TMS320LC545 and TMS320LC546

The ’LC545 and ’LC546 DSPs offer low power consumption and provide the first single-DSP solutions for next-generation cellular standards. The devices have the on-chip memory (48K 16-bit words of ROM, 6K 16-bit words of RAM), performance, and intelligent peripherals required to support the half-rate GSM, half-rate PDC, and CDMA enhanced variable-rate coder (EVRC). Complementing the large amount of on-chip memory is an autobuffered serial port, an HPI for interfacing with other DSPs or embedded microprocessors, a standard serial port, a software wait-state generator, and a PLL clock. The ’LC545 and ’LC546 devices feature an integrated Viterbi accelerator; powerful single-cycle instructions (dual-operand, parallel, and conditional instructions); low active-mode power dissipation (less than 35 mW to run VSELP); and low-power standby modes. These devices offer instruction cycle times of 25 ns (40 MIPS) or 20 ns (50 MIPS). The ’LC545 is available in a 128-pin TQFP. The ’LC546 offers a cost reduction by removing the HPI and is packaged in a 100-pin TQFP.

Figure 8–3 shows a block diagram of the ’LC545/LC546.

Figure 8–3. TMS320LC545/LC546 Block Diagram



Note: The ’LC545 and ’LC546 are available with one of two different PLL options.
You choose one of the options listed.

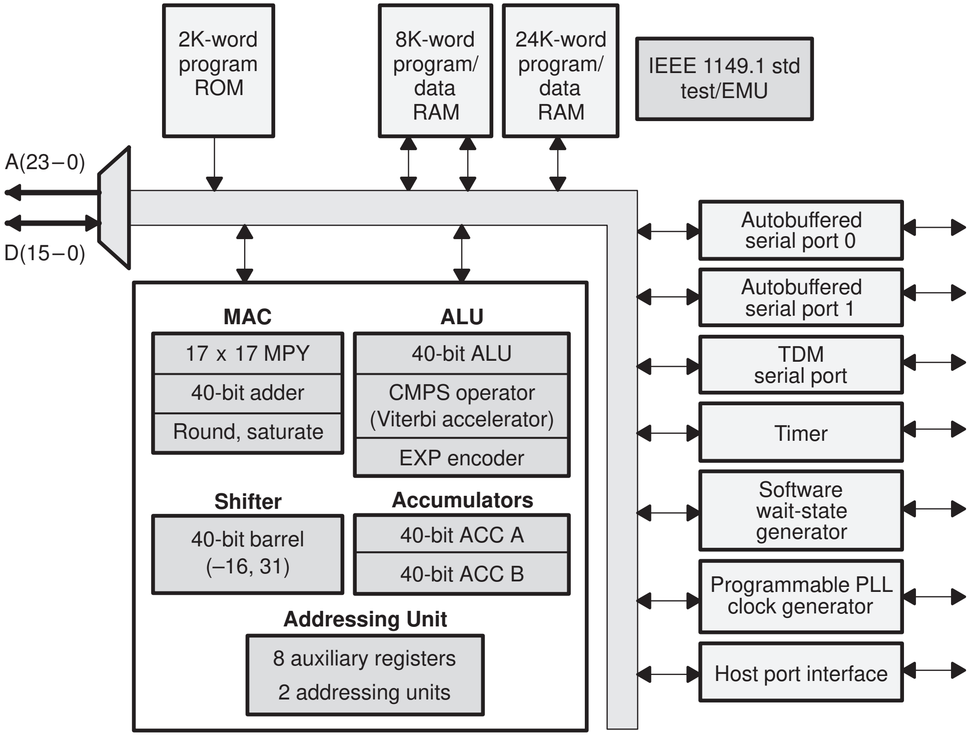
8.6 TMS320LC548

The TMS320LC548 combines high performance with low power consumption, making it well suited for wireless telecommunications and other mobile systems that need to perform complex functions while also conserving battery power. The 'LC548 includes two high-speed BSPs, one TDM serial port, a HPI, and a 16-bit on chip timer. The 'LC548 can operate at 15 ns (66 MIPS), 12.5 ns (80 MIPS), or 10 ns (100 MIPS) with a supply voltage of 3.3 V. These devices offer 32K 16-bit words of RAM and 2K 16-bit words of boot ROM on-chip, as well as an extended addressing mode for 8M by 16-bit maximum addressable external program space.

These devices also feature an integrated Viterbi accelerator; powerful single-cycle instructions (dual-operand, parallel, and conditional instructions); low active-mode power dissipation (less than 35 mW to run VSELP); low-power standby modes; and JTAG with boundary scan.

Figure 8–4 shows a block diagram of the 'LC548/VC548.

Figure 8–4. TMS320LC548 Block Diagram



8.7 TMS320LC549 and TMS320VC549

The TMS320LC549 and TMS320VC549 are identical devices except that the core voltage supply on the ’VC549 is 2.5 V, allowing it to operate at 10-ns (100 MIPS) instruction cycle time.

The TMS320LC549 and TMS320VC549 feature:

❑ 10–ns (100 MIPS), 2.5-ns (80 MIPS), or15-ns (66MIPS) instruction time cycles

❑ 2.5- and 3.3-V operation

❑ 32K 16-bit words of RAM and 16K 16-bit words of boot ROM on-chip

❑ Extended addressing mode for 8M x 16–bit maximum addressable external program space

❑ Two autobuffered serial ports (BSPs)

❑ TDM serial port

❑ Host port interface

❑ Integrated Veterbi accelerator

❑ Powerful single-cycle instructions (dual operand, parallel instructions, conditional instructions)

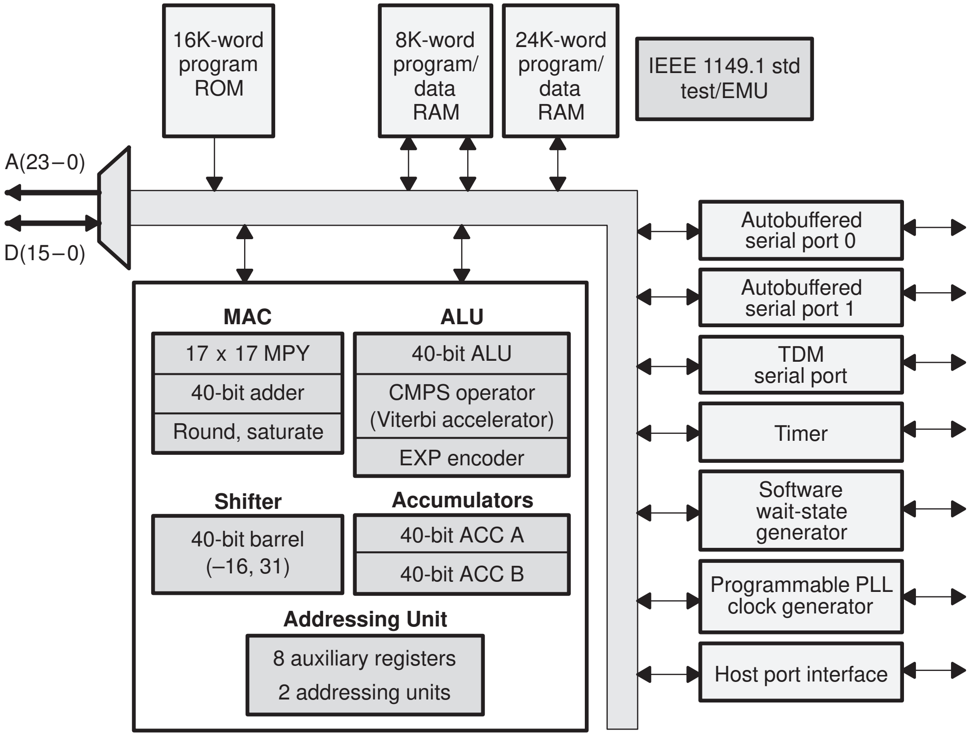
❑ Low active-mode power dissipation (less than 35 mW to run VSELP

❑ Low–power standby mode

❑ JTAG with boundary scan

❑ 16-bit on-chip timer

Figure 8–5. TMS320LC549/VC549 Block Diagram



8.8 Tools for TMS320C54x

For information about the code generation and debugging tools available for the ’C54x devices, see Chapter 13, *Code Generation Tools*, and Chapter 14, *System Integration and Debugging Tools*. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

Chapter 9

TMS320C6x Devices

The TMS320C6x devices are the first devices to feature VelociTI™, an advanced very long instruction word (VLIW) architecture developed by Texas Instruments, which allows performance of up to 1600 million instructions per second (MIPS). The first device in the series is the TMS320C6201, a fixed-point digital signal processor (DSP). TI announced the TMS320C6701, a floating-point version of the 'C6201, performing 1 GFLOPS (one billion floating-point operations per second), in April 1998.

Topic **Page**

9.1 TMS320C6x Introduction

With a complete set of development tools, the ’C6x devices offer cost-effective solutions to high-performance DSP programming challenges. The ’C6x development tools include a new C compiler, an assembly optimizer, and a Windows-based debugger. VelociTI combines an advanced VLIW architecture with a high degree of parallelism to produce a device that enables applications such as:

- ❑ Unlimited Internet bandwidth
- ❑ Universal wireless communications
- ❑ New telephony features
- ❑ Remote medical diagnostics
- ❑ Automated cruise control
- ❑ Personal home base station
- ❑ Personalized home security

The ’C6x devices also can be used for improved performance on existing applications, such as:

- ❑ Wireless local-loop base stations
- ❑ Beam-forming base stations
- ❑ Pooled modems and remote access servers
- ❑ Virtual reality 3-D graphics
- ❑ Speech recognition
- ❑ Atmospheric modeling
- ❑ Finite element analysis and imaging (for example, fingerprint recognition, ultrasound, and MRI)
- ❑ Next-generation xDSL modems and cable modems
- ❑ Multichannel telephony platforms, including central office switches, PBXs, and voice-messaging systems
- ❑ Multimedia systems
- ❑ Digital imaging
- ❑ Network switches and routers

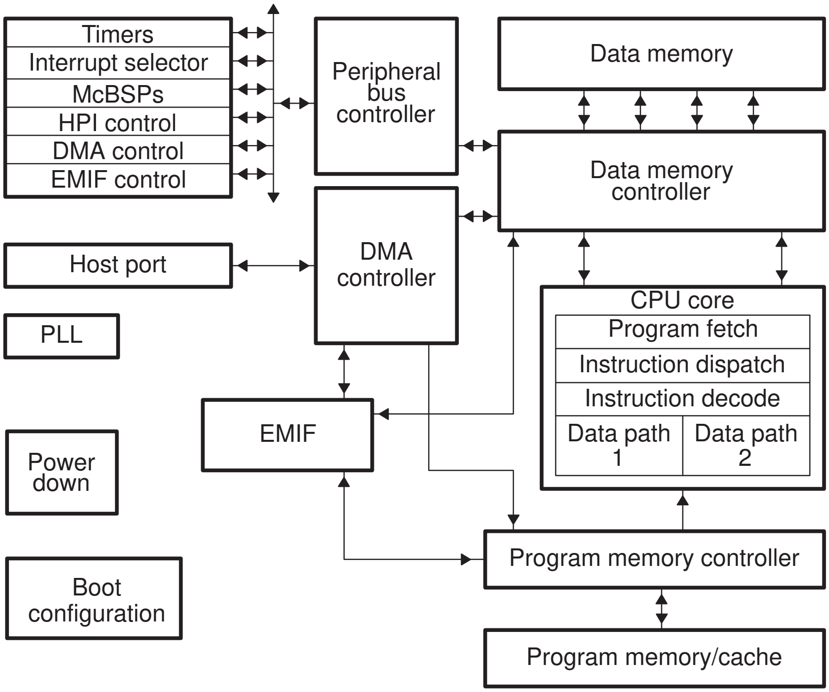
9.2 TMS320C62x Key Features

The TMS320C6201 is the first fixed-point processor in the ’C6x generation. Following are key features of the TMS320C6201:

- ❑ VelociTI advanced very long instruction word (VLIW) architecture
  - Load/store architecture
  - Instruction packing for reduced code size
  - 100% conditional instructions for faster execution
  - Intuitive, reduced instruction set computing (RISC)-like instruction set
- ❑ CPU
  - Eight independent functional units (including two 16-bit multipliers with 32-bit results and six arithmetic logic units [ALUs] with 32/40-bit results)
  - 32 32-bit registers
  - 1600 million instructions per second (MIPS)
  - 5-ns cycle time
  - Up to eight 32-bit instructions per cycle
  - Byte-addressable 8-, 16-, 32-bit data
  - 32-bit address range
  - Dual-endian support
  - Saturation
  - Normalization
  - Bit-field instructions (extract, clear, left most bit detection)
- ❑ Memory/peripherals
  - Synchronous external memory interface (EMIF)
  - Two multichannel buffered serial ports (McBSPs)
  - 4-channel direct memory access (DMA)
  - Two timers
  - ×4 phase-locked-loop (PLL) option
  - Host-port interface (HPI)
  - 1M-bit on-chip memory (divided into 2K by 256 bits of program memory and 64K bytes of data memory)
- ❑ 352-pin ball-grid array package

Figure 9–1 shows the CPU core and peripherals for a TMS320C62x device.

Figure 9–1. ’C62x CPU Core With Peripherals



9.3 TMS320C67x Key Features

The TMS320C6701 is a floating-point version of the 'C6201.The 'C6201 and 'C6701 are pin-for-pin compatible and all 'C62x instructions run unchanged on the 'C67x devices. Following are key features of the TMS320C6701:

- ❑ VelociTI advanced very long instruction word (VLIW) architecture
  - ■ Load/store architecture
  - ■ Instruction packing for reduced code size
  - ■ 100% conditional instructions for faster execution
  - ■ Intuitive, reduced instruction set computing (RISC)-like instruction set
- ❑ CPU
  - ■ Eight independent functional units (including two 16-bit multipliers with 32-bit results and six arithmetic logic units [ALUs] with 32/40-bit results)
  - ■ 32 32-bit registers
  - ■ 1336 million instructions per second (MIPS)/ 1 bilion floating-point operations per second (1 GFLOPS)
  - ■ 5-ns cycle time
  - ■ Up to eight 32-bit instructions per cycle
  - ■ Byte-addressable 8-, 16-, 32-bit data
  - ■ 32-bit address range
  - ■ Dual-endian support
  - ■ Saturation
  - ■ Normalization
  - ■ Bit-field instructions (extract, clear, left most bit detection)
  - ■ IEEE single-and-double-precision hardware support
- ❑ Memory/peripherals
  - ■ Synchronous external memory interface (EMIF)
  - ■ Two multichannel buffered serial ports (McBSPs)
  - ■ 4-channel direct memory access (DMA)
  - ■ Two timers
  - ■ ×4 phase-locked-loop (PLL) option
  - ■ Host-port interface (HPI)
  - ■ 1M-bit on-chip memory (divided into 2K by 256 bits of program memory and 64K bytes of data memory)
- ❑ 352-pin ball-grid array package

9.3.1 Addressing Modes

The addressing mode options on the ’C62x and ’C67x are either linear or circular, as specified by the addressing-mode register (AMR).

For more information on addressing modes, see the *TMS320C62x/ TMS320C67x CPU and Instruction Set Reference Guide*.

9.3.2 Interrupts

The CPU has 14 interrupts. These are the reset interrupt, the nonmaskable interrupt (NMI), and interrupts 4–15. These interrupts correspond to the $\overline{\text{RESET}}$, NMI, and INT4–INT15 signals on the CPU boundary. In some ’C62x and ’C67x 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 *TMS320C62x/TMS320C67x Instruction Set Reference Guide*.

9.4 Central Processing Unit (CPU)

The central processing unit (CPU) is the central building block of all the TMS320C62x and TMS320C67x devices. The common CPU features include:

- ❑ Program fetch unit
- ❑ Instruction dispatch unit
- ❑ Instruction decode unit
- ❑ 32 general-purpose, 32-bit registers
- ❑ Two data paths, each with four functional units, including one multiplier and three arithmetic logic units (ALUs) on each data path
- ❑ 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 and a register file containing 16 32-bit registers. The functional units execute logic, shift, multiply, and data address operations. All instructions operate on the registers. The two sets of data-addressing units are exclusively responsible for all data transfers between the register files and the memory.

9.5 TMS320C6201/C6701 Internal Memory

The internal memory consists of 512K bits of on-chip program/cache memory and 512K bits of on-chip data memory. The program memory, configurable as cache or program, is organized in 2K of 256-bit fetch packets. The ’C6201/C6701 fetches all instructions one fetch packet at a time. The packets are processed at the maximum rate of one packet (eight 32-bit instructions) per CPU cycle or at a minimum of one instruction per cycle. The internal data memory is byte addressable by the CPU (for reads as well as writes) and supports bytes, half-words, and full word transfers.

9.5.1 TMS320C6201/C6701 Data-Memory System

The ’C6201/C6701 data-memory system includes SRAM and a memory controller. The 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 per 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 ’C6201/C6701 data-memory system also supports direct-memory access (DMA) and external host accesses. For more information on the data-memory system, see the *TMS320C6201/C6701 Peripherals Reference Guide*.

9.5.2 TMS320C6201/C6701 Program-Memory System

The ’C6201/C6701 program-memory system includes on-chip SRAM and a memory/cache controller. The program memory can operate as either an internal program memory or as a directly mapped program cache. There are four modes under which the 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.

For details on cache modes, see the *TMS320C6201/C6701 Peripherals Reference Guide*.

9.6 TMS320C6201/C6701 Peripherals

In addition to on-chip memory, the ’C6201/C6701 contains the following peripherals:

- External memory interface (EMIF)
- 4-channel direct-memory access (DMA) controller
- Host-port interface (HPI)
- Power-down logic
- Two multichannel buffered serial ports (McBSPs)
- Two 32-bit timers

9.6.1 TMS320C6201/C6701 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 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 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.

For more information on memory, see the *TMS320C62x/C7x CPU and Instruction Set Reference Guide.* For more information on the EMIF, see the *TMS320C6201/C6701 Peripherals Reference Guide*.

9.6.2 TMS320C6201/C6701 Direct-Memory Access (DMA)

The on-chip DMA offers four independent, programmable channels that 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 to keep 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 ’C6201/C6701 modules:

❑ Internal data memory

❑ Internal program memory that is not configured as cache as a destination of a transfer

❑ EMIF

The processor can use one of the DMA channels 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. However, the DMA cannot use program memory as the source of a transfer and it cannot access memories configured as cache or memory-mapped on-chip peripheral registers.

See the data sheet for the specific device to find the memory mapping of DMA control registers. These are 2-bit-wide registers and must be accessed through 32-bit accesses from the CPU. For more information on the DMA operations, see the *TMS20C6201/C6701 Peripheral Reference Guide*.

9.6.3 TMS320C6201/C6701 Host-Port Interface

The HPI is a parallel port that can access the CPU memory space directly as an asynchronous interface. A host (external) processor can read from and write to the internal data memory through the 16-bit-wide access of the HPI.

The HPI can boot load the CPU as well as access the full range of the ’C6201/C6701 memory. Also, the HPI offers improved performance and can operate without impacting CPU performance.

For more information on the HPI, see the *TMS320C6201/C6701 Peripheral Reference Guide*.

9.6.4 TMS320C6201/C6701 Power-Down Logic

The 'C6201/C6701 supports three power-down modes (idle1, 2, and 3) that can reduce system power requirements significantly. Idle 1 halts the CPU except for the interrupt logic. Idle2 halts the CPU and the peripherals (except for the interrupt logic). Idle3 halts the phase-locked loop (PLL), stopping the clock tree from switching, which effectively halts the entire chip. Idle3 requires a reset to wake up the device, while the other two modes can be restored using an interrupt or reset. For more details on the power-down logic, see the *TMS320C6201/C6701 Peripherals Reference Guide*.

9.6.5 TMS320C6201/C6701 Multichannel Buffered Serial Port (McBSP)

The 'C6201/C6701 includes two McBSPs, supporting multivendor interface protocol (MVIP) and timers to allow easy algorithm integration. The McBSP is based on the standard TMS320C2x/C5x/C54x serial-port interface. 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.

The McBSP provides:

- ❑ Full-duplex communication
- ❑ Double-buffered data registers
- ❑ Direct interface to other devices
- ❑ Clock generation or an internal programmable frequency shift clock
- ❑ Multichannel transmit and receive

9.6.6 TMS320C6201/C6701 Timers

The device has two 32-bit general purpose timers that you can use to:

- ❑ Time events
- ❑ Count events
- ❑ Generate pulses
- ❑ Interrupt the CPU
- ❑ Send synchronization events to the DMA

The timer has two signaling modes and can be clocked by an internal or an external source. The timer has an I/O pin that functions as an input clock, as an output clock, or as a general-purpose I/O pin.

9.7 Tools for TMS320C6x

For information about the code generation and debugging tools available for the ’C6x devices, see Chapter 13, *Code Generation Tools*, and Chapter 14, *System Integration and Debugging Tools*. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

TMS320C8x Devices

TMS320C8x devices—the world's first 64-bit DSPs designed for the multiprocessing environment—include the 'C80 and the 'C82. This chapter describes the devices, lists key features, and provides block diagrams.

Topic **Page**

10.1 TMS320C8x Introduction

The processing power of the ’C8x devices supports any application that requires high-performance digital signal processing. The first member of the ’C8x generation, the ’C80, is a single-chip, parallel processor that can be used for applications such as real-time audio/video processing, high-end data communications, and image processing. The ’C82 is the second member of the ’C8x family that can be used in high-volume applications like desktop video-conferencing, high-speed telecommunications, and three-dimensional graphics.

Since the ’C8x devices are fully programmable, you can blend industry-standard algorithms with proprietary software to optimize system performance and differentiate products in the market. You receive added flexibility, because a ’C8x-based system can be adapted to evolving technologies through software modifications rather than hardware redesigns. This enables you to upgrade as new standards emerge.

10.2 TMS320C8x Key Features

The following are key features of the high-performance TMS320C8x generation.

- Version capable of over 2 billion operations per second (BOPS)
- 50- or 60-MHz performance
- A 32-bit reduced instruction set computer (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
- 50K bytes of on-chip RAM on the ’C80 and 44K bytes 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 Gbytes/s transfer rates on the ’C80
  Up to 2.6 Gbytes/s transfer rates on the ’C82
- Big-endian and little-endian byte-ordering modes
- A 4G-byte memory address space
- A 64-bit transfer controller capable of up to 400 Mbytes/s in on-chip and off-chip memory transfers

  Dynamic sizing of bus width (64, 32, 16, or 8 bits)
  Direct access to 64-bit VRAM/DRAM/SRAM/SDRAM memory
  Direct access to 64-bit extended data out (EDO) DRAM (’C82)
- A video controller that contains dual-frame timers for simultaneous image capture and display (’C80 only)
- Four external interrupts (three edge-triggered and one level-triggered)
- A full-scan design (plus boundary scan), accessed via an IEEE Standard 1149.1 (JTAG)-compliant test port that provides emulation support
- A 3.3-V design
- TI EPIC 0.5 µm CMOS technology
- Efficient packaging

  305-pin ceramic pin grid array (CPGA) or 352 ceramic ball grid array (CBGA) (’C80)

  352-ball grid array (BGA) (’C82)

10.3 TMS320C8x Master Processor (MP)

The master processor (MP) is a 32-bit RISC processor with an integral IEEE-754 floating-point unit. As with other RISC processors, all accesses to memory are performed with load and store instructions, and most integer and logical operations are performed on registers in a single cycle. The floating-point instructions are pipelined; therefore, you can start a single-precision multiply or any floating-point add instruction on each clock cycle. Moreover, the floating-point unit approaches 120 MFLOPS in performance at a 60-MHz internal clock rate.

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

The MP is structured for efficient execution of C code. For example, the MP contains an R0 register, often called a zeroing register, used by C. Also, the MP instruction set is tailored to contain many of the C executables found in compiler technology.

Features of the master processor include:

❑ 32-bit RISC CPU delivering 60 MIPS at 60 MHz

  Targeted for high-level languages

❑ IEEE-754 120-MFLOP floating-point unit

  Parallel multiply, add, and load/store

❑ 31 32-bit registers

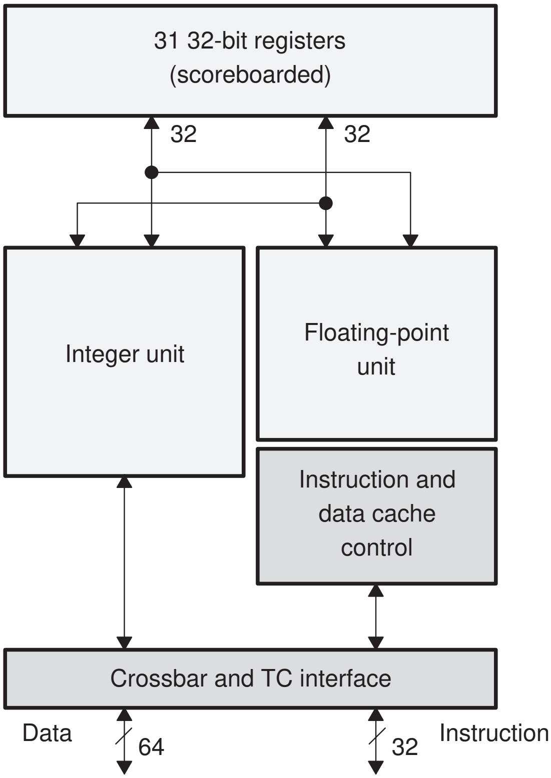
  Single file for integer and floating point
  Loads and FPU results are scoreboarded

❑ Instruction and data cache control

  4K-byte instruction cache
  4K-byte data cache
  2K-byte parameter RAM (’C80), 4K-byte parameter RAM (’C82)

Figure 10–1 shows a block diagram of the ’C8x master processor.

Figure 10–1. TMS320C8x Master Processor (MP)



10.4 TMS320C8x MP Floating-Point Unit

The MP’s floating-point unit is capable of performing IEEE-754 floating-point operations in 32-bit single-precision and 64-bit double-precision floating point. Conversion between different formats is also supported. In addition, the floating-point unit provides vector floating-point operations with the option of performing a parallel load or store to improve program efficiency.

Hardware support for the floating-point unit consists of a full double-precision floating-point add unit and a 32-bit single-precision floating-point multiply unit.

Features of the MP floating-point unit include:

❑ IEEE-754 floating point

Hardware-exception handling

❑ FP add unit with double-precision ALU

1-cycle adds/subs/compares (single and double) and conversions
6-cycle single-and 20-cycle double-precision divide
9-cycle single-and 26-cycle double-precision square root

❑ Floating-point multiply unit performs all multiplies (integer and floating-point), divides, and square roots

1-cycle single-precision multiply
4-cycle double-precision multiply

❑ Pipelined—can start a new instruction every cycle

3-stage pipeline
Register file scoreboard prevents races

❑ Vector FP for 120-MFLOP operation

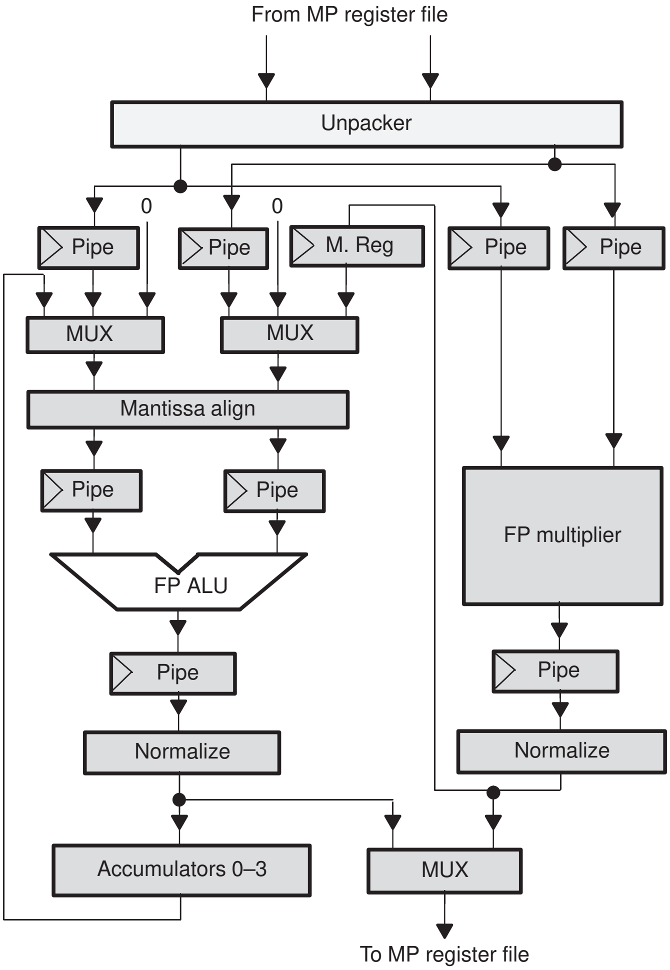
Parallel multiply, add, and 64-bit load (p++) in one cycle
Four double-precision accumulator registers support pipelining
Supports matrix multiplies, DCTs, and FFTs

❑ FP status and interrupt-enable registers

MP’s test-and-branch instructions access FP status

Figure 10–2 shows a block diagram of the ’C8x MP floating-point unit.

Figure 10–2. TMS320C8x MP Floating-Point Unit



10.5 TMS320C8x Parallel Processing (PP) Advanced DSPs

The parallel processing (PPs) advanced digital signal processors provide much of the ’C8x’s performance. The PPs are designed to perform digital signal processing along with bit-field and multiple-pixel manipulation. These processors have advanced features that are not found in any other DSP or general-purpose processor and can perform in excess of ten RISC-like operations in each cycle.

In order to specify the multiple parallel operations that the PPs can perform, a wide instruction word of 64 bits is used. The instruction has fields that independently control the data unit and the two address units. All instructions execute in a minimum of a single cycle.

Each PP has a register file 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. The PP features:

❑ 3-input ALU with mixed arithmetic and Boolean operations

  Can perform masking at the same time as an add or subtract

❑ Flexible data path feeding 3-input ALU

  Fast bit and file processing

❑ Address data paths can be used for general-purpose arithmetic

❑ Byte/halfword multiple arithmetic

  Single instruction stream, multiple data stream (SIMD) processing within each processor

  Better handling of pixels and Z-buffers than in other DSPs or general-purpose processors

❑ Eight primary data registers, d0 to d7 (D registers), that can perform up to seven reads and four writes

  Two multiplier sources, three ALU sources, one multiplier result, one ALU result, and three LD/ST/MOVE

❑ Splittable multiplier for fast pixel path

  Any D register can be used on a multiply-with-parallel-add

❑ Three levels of zero-overhead loops

❑ Conditional operations (for ALU, load/store, and/or register source)

Additional features include:

- ❑ Two address units

  Up to two memory operations per cycle

- ❑ Single-cycle multiplier

  One 16-bit or two 8-bit results per cycle

- ❑ Splittable 3-input ALU

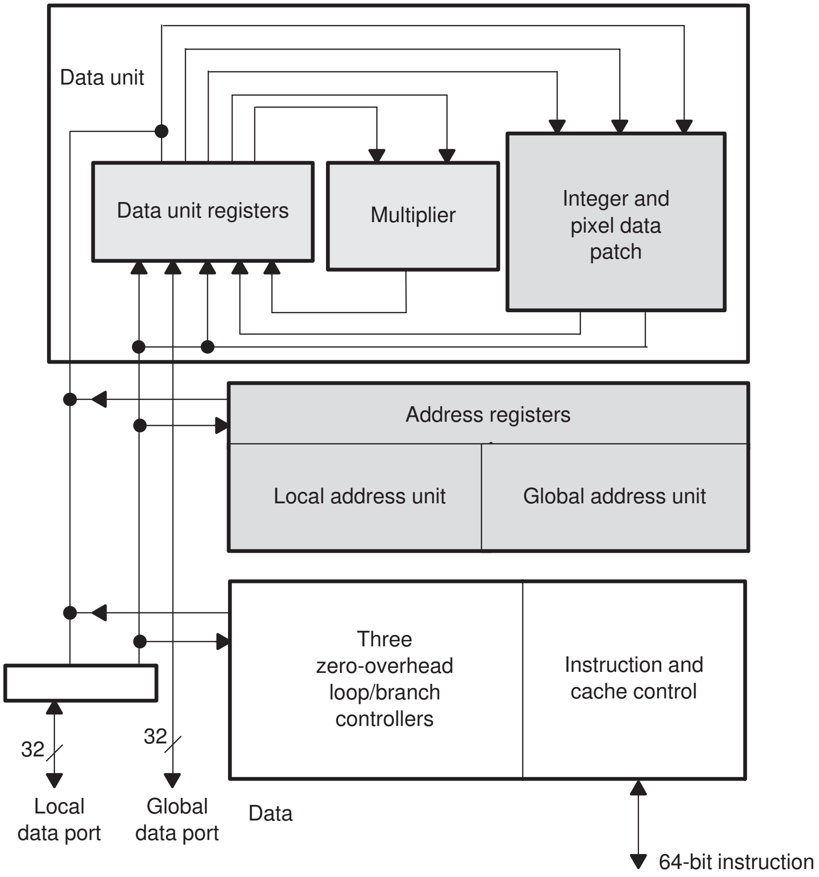
  Multiple operations in each pass

  Up to four 8-bit results per cycle

- ❑ Pixel and bit field hardware

Figure 10–3 shows a block diagram of the ’C8x PP.

Figure 10–3. TMS320C8x Parallel Processing (PP) Advanced DSPs



10.6 TMS320C8x Parallel Processing (PP) Data Unit

The data unit has two data paths; each data path has its own set of hardware that functions independently of the other data path.

The ALU data path includes a barrel rotator, mask generator, 1-bit to n-bit expander, and a 3-input ALU that can combine the mask or expander output with register data to create over 2000 different processing options. The 3-input ALU can perform 512 logical and/or mixed logical and arithmetic operations that support masking or merging and addition/subtraction in a single pass. The ALU can also be split to perform multiple 8-bit or 16-bit operations in parallel.

The PP data unit features are:

❑ 3-input ALU (512 operations)

Mixed arithmetic and Boolean in one cycle (mask and add/sub in one pass)

Mask/merge and field processing

Splittable for multibyte operations

❑ 16-bit by 16-bit multiplier (32-bit results)

Rounding for DCT accuracy
Splittable into two 8-bit- by 8-bit multipliers (16-bit results)

❑ Flexible data path

Barrel rotator
Mask generator
N-to-1 and 1-to-N translations via mf register
Left/rightmost 1 and bit-change

❑ 44 user-visible registers

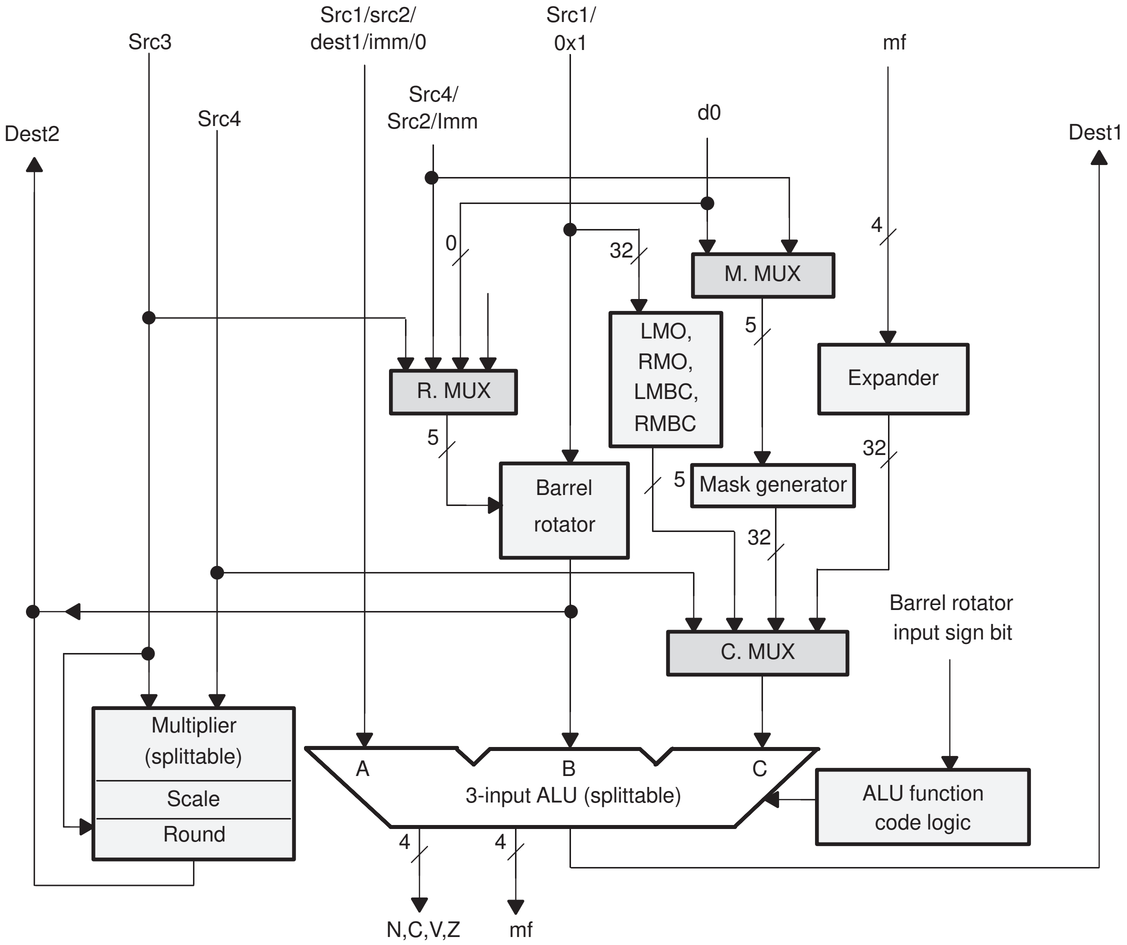
Any register can be operand of ALU

❑ Eight D registers
❑ Conditional operations

Conditional choice of register pair source
Conditional save of result
Add … => instruction and data cache control
2K-byte instruction cache (’C80), 4K-byte instruction cache (’C82)
2K-byte instruction parameter RAM
8K-byte data RAM

Figure 10–4 shows a block diagram of the ’C8x PP data unit.

Figure 10–4. TMS320C8x PP Data Unit



10.7 TMS320C8x Transfer Controller (TC)

The transfer controller (TC) is a combined DMA machine and memory interface that intelligently queues, prioritizes, and services the data requests and cache misses of the MP and the PPs. The transfer controller interfaces directly with the on-chip SRAMs. Through the TC, all of the processors can access the system external to the chip. In addition, data-cache or instruction-cache misses are automatically handled by the TC.

Data transfers are specifically requested by the PPs or the MP in the form of linked-list packet transfers, which are handled by the TC. These requests allow multidimensional blocks of information to be transferred between a source and destination, either of which can be on-chip or off-chip. Packet-oriented data transfers offer compatibility with several local area network standards, such as ATM.

The TC performs:

- ❑ Cache fills and writes
- ❑ Direct loads and stores from/to off-chip memory through DEA request
- ❑ Block movement of data through packet transfers
- ❑ Refresh and SRT (shift register transfer) cycles needed to maintain DRAMs and VRAM capture/display buffer, respectively

Features of the TC include:

- ❑ 400 Mbytes/s external bandwidth
- ❑ Direct RAM, VRAM, SRAM, and SDRAM control, EDO DRAM (’C82)
- ❑ Dynamic bus sizing (64, 32, 16, or 8 bits)
- ❑ Packet transfers controlled autonomously by transfer controller
- ❑ Linear x/y addressing

  Independent source and destination
  Automatic byte alignment

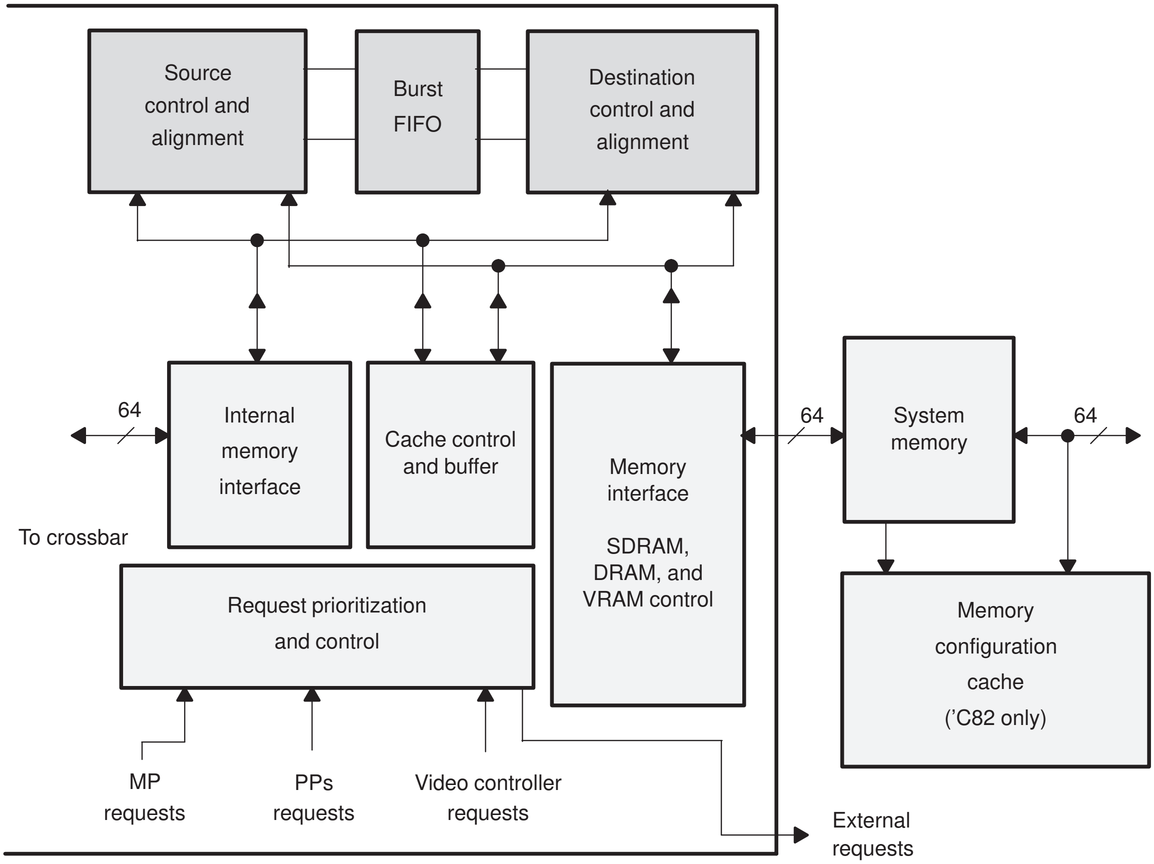
- ❑ Intelligent request

  Queuing and prioritization

The ’C82 TC includes a memory configuration cache that consists 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 and it can be locked into a set high or low priority. The configuration cache reduces the number of pins necessary in the ’C82 and in support chips.

Figure 10–5 shows a block diagram of the ’C8x TC.

Figure 10–5. TMS320C8x Transfer Controller (TC)


Source control and alignment
Burst FIFO
Destination control and alignment
64
Internal memory interface
Cache control and buffer
Memory interface
SDRAM, DRAM, and VRAM control
64
System memory
64
To crossbar
Request prioritization and control
Memory configuration cache ('C82 only)
MP requests
PPs requests
Video controller requests
External requests

10.8 TMS320C80

The ’C80, the first member of the TMS320C8x generation, is a 50-, or 60-MHz, CMOS multiprocessor DSP designed for high-performance processing. The ’C80 architecture combines a floating-point unit and a RISC processor to provide a multiprocessor environment on a single device.

The ’C80 combines four 32-bit parallel processors (PPs) with a 32-bit master RISC processor, a video controller, a transfer controller, and 50K bytes of SRAM on a single chip. Each PP is capable of many parallel operations per cycle. The PPs perform pixel/field processing as well as digital signal processing. The RISC processor has a 120-MFLOPS IEEE-754 floating-point unit.

The ’C80 has a 16- or 20-ns cycle time, executes at over 2 BOPS, and achieves 400M bytes/s of data to off-chip memory.

The ’C80’s on-chip RAM supports many parallel accesses per cycle. A crossbar switch supports up to 4.2G bytes/s transfer rates: 2.4G bytes/s of data and 1.8G bytes/s of instructions. The 32K bytes of RAM can be shared by all processors and the transfer controller. The video controller supports any display or capture resolution. Other features of the ’C80 include:

❑ Transfer controller (supports multidimensional packet transfers)

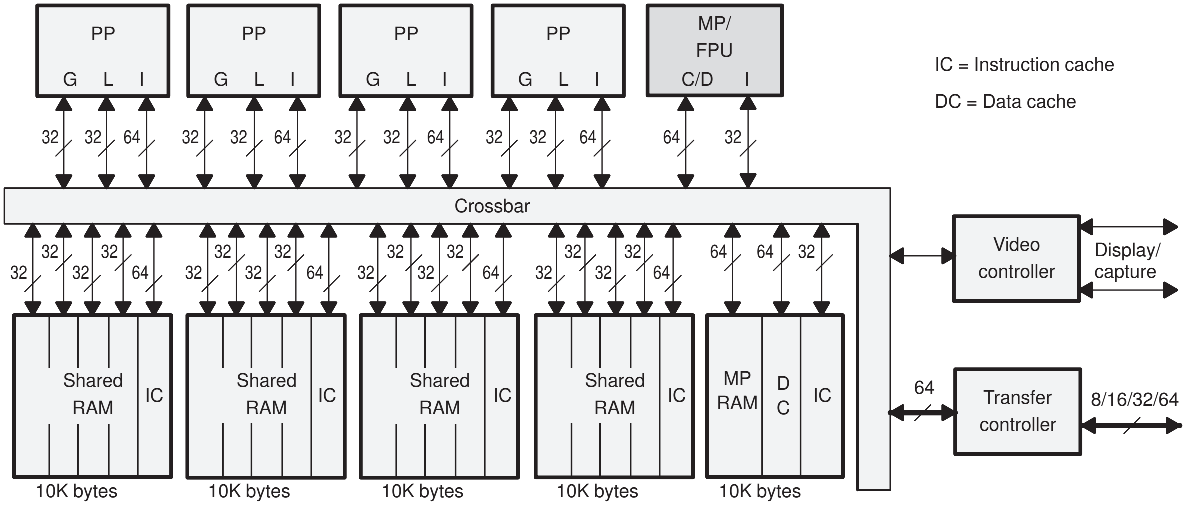
    400 Mbytes/s on- and off-chip memory transfers
    4G-byte memory address space
    Access to 8-, 16-, 32-, or 64-bit SRAM, VRAM, DRAM, SDRAM
    Offloads memory manipulations from the processors

❑ 0.5-µm CMOS technology

❑ Efficient packaging: 305 ceramic pin grid array (CPGA) or 352 ball grid array (BGA) package.

Figure 10–6 shows a block diagram of the ’C80.

Figure 10–6. TMS320C80 Block Diagram



10.9 TMS320C80 Video Controller (VC)

The video controller is included only in the architecture of the ’C80 and is the interface between the device and image-capture display systems.

The ’C80 has two sets of frame timing counters and registers. The video controller keeps track of horizontal and vertical synchronization and blanking timing, as well as supporting a 2-dimensional border region. Each counter has its own asynchronous clock inputs. These synchronization signals can be individually set up as outputs (for display) or inputs (for capture).

The shift-register transfer (SRT) controller has comparators that cause shift-register transfer cycles for VRAMs or cause packet transfers for DRAM-based display memories.

Four main sections make up the video controller:

- Frame timers
- Serial register transfer controller
- Register interface
- Multiplexer

Features of the video controller include:

- Two identical frame timers

  Can be used for display or capture

  Each has an asynchronous clock

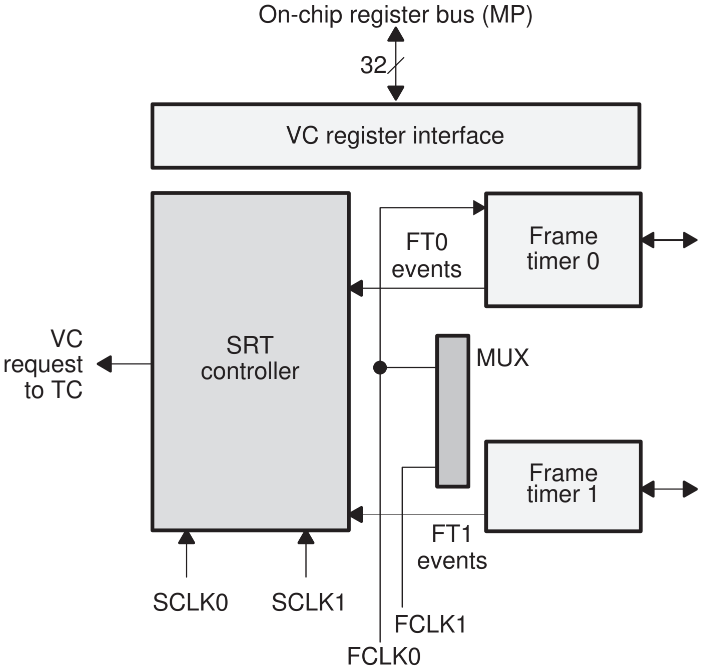
  Generate fully-programmable horizontal, vertical, blank, and border timing

- SRT controller

  Controls two display/capture regions
  Has VRAM shift register transfer control
  Generates timer interrupts to the MP
  Generates timer packet requests to the TC

Figure 10–7 shows a block diagram of the ’C80 VC.

Figure 10–7. TMS320C80 Video Controller (VC)



10.10 TMS320C82

The ’C82 is a 50- or 60-MHz DSP that combines two PPs with 64-bit instructions and 32-bit fixed-point data. Each PP is capable of many parallel operations per cycle. The PPs perform pixel/field processing as well as digital signal processing. The RISC processor has a 100-MFLOPS IEEE-754 floating-point unit. The ’C82 is capable of executing at more than 1.5 BOPS, with a 16-ns cycle time.

The ’C82’s on-chip RAM supports many parallel accesses per cycle. A crossbar switch supports up to 2.6-Gbytes/s transfer rates: 1.6 Gbytes/s of data and 1.0 Gbytes/s of instructions. 32K bytes of RAM can be shared by all processors and the transfer controller. Other features of the ’C82 include:

❑ Transfer controller (supports multidimensional packet transfers)

  400 Mbytes/s on- and off-chip memory transfers

  4G-byte memory address space

  Access to 8-, 16-, 32-, or 64-bit SRAM, DRAM, SDRAM, and EDO DRAM

  Offloads memory manipulations from the processors

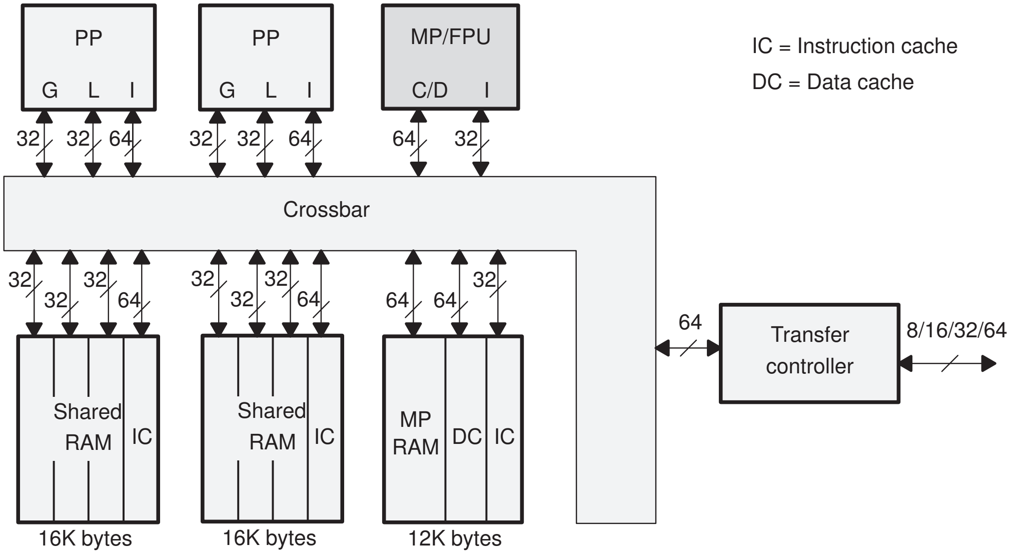
  On-chip memory configuration cache consisting of six 32-bit words to describe the properties of six banks of external memory

❑ 0.5-µm CMOS technology

❑ Efficient packaging: 352 ball grid array (BGA) package

Figure 10–8 shows the TMS320C82 block diagram.

Figure 10–8. TMS320C82 Block Diagram



10.11 Tools for TMS320C8x

For information about the code generation and debugging tools available for the ’C8x devices, see Chapter 13, *Code Generation Tools*, and Chapter 14, *System Integration and Debugging Tools*. Table 2–13 in Chapter 2, *Selection Guide*, provides part numbers and host system information.

Chapter 11

Mixed-Signal Products

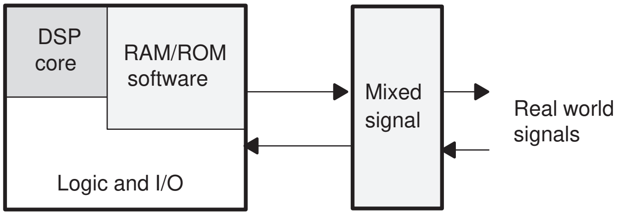
A mixed-signal product is one that provides an interface between the digital and the analog world. Digital signal processing solutions often include a DSP, software, an I/O interface, and mixed-signal products. This chapter describes mixed-signal products that are well suited to providing interfaces for DSP solutions.

Topic **Page**

11.1 DSP Solutions

DSP Solutions (DSPS) are composed of a DSP core, software, system expertise, and a mixed signal product (see Figure 11–1) with peripheral devices such as memory and logic used in embedded systems where the DSP is the primary processor.

Figure 11–1. DSP Solutions



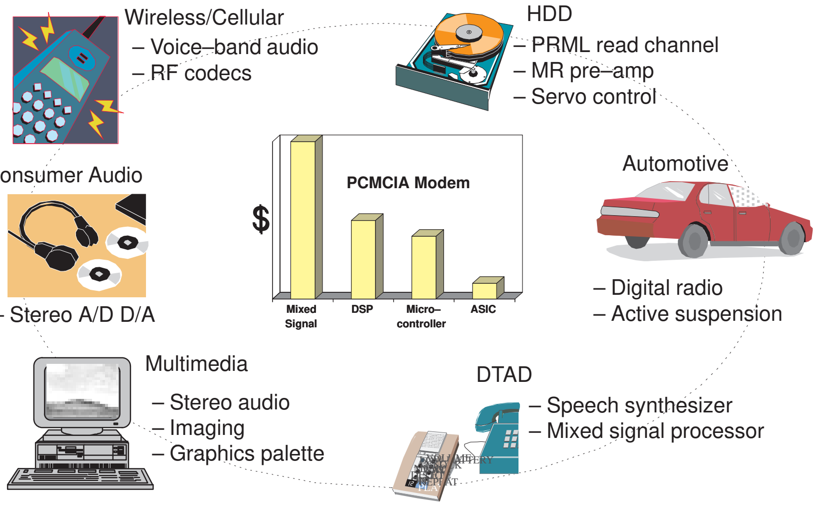
As part of TI's effort to support communications standards worldwide with complete DSP Solutions, TI has developed mixed-signal devices optimized for various uses. Examples of DSP Solutions (see Figure 11–2) include:

❑ Hard disk drives (HDDs)
❑ Digital telephone answering devices (DTADs)
❑ Automotive audio and suspension systems
❑ Multimedia
❑ Consumer audio
❑ Wireless/cellular

One of the best examples is in the wireless/communications field. Originally, cellular telephones were the size and weight of a brick with talk times of 30 minutes. Cellular phones now fit in your pocket, weigh less than a pound, and have talk times of three hours. All of this has been made possible because of DSPS.

The integration of various functions and features within DSP software and the use of an RF codec and the voice-band audio processor have made this progress possible.

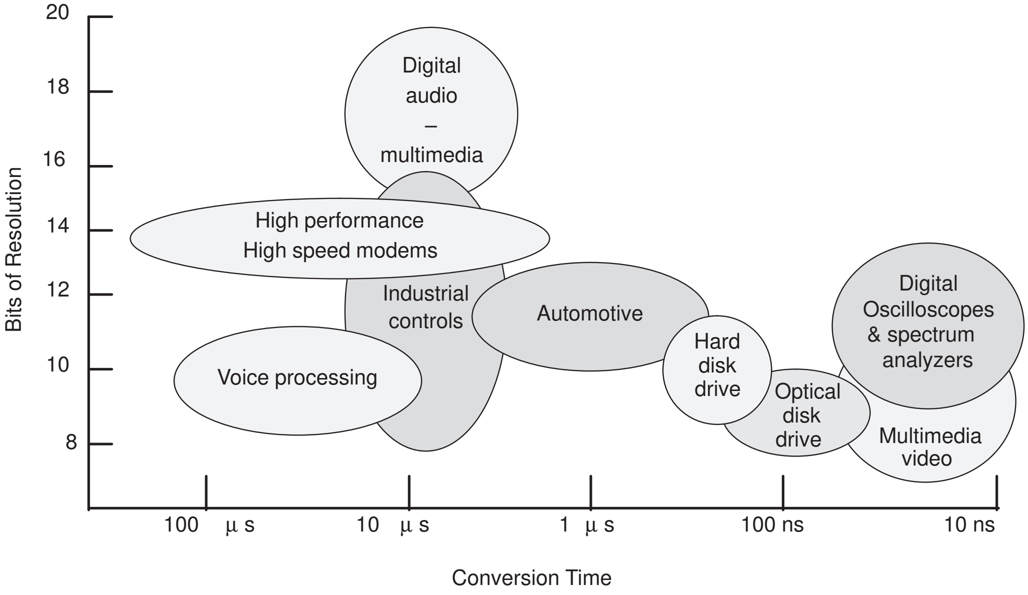
Figure 11–2. Examples of DSP Solutions



Data converters are the classic form of DSP peripherals from the analog point of view. Classic data converters, 10- to 12-bit data converters with a 10-µs conversion time, have been around for several years, supplied by Texas Instruments and other manufacturers. The new equipment, driven by DSPs, is pushing this envelope out in two directions. In the video direction, the need is for much greater speed. Bits of resolution are not as important as very high speed, often with conversion time of 10 ns or less.

On the other end of the spectrum, in audio, the ear is a very discriminating instrument so you need very high resolution and speed is not as critical. For the 10-µs conversion time, we need 18-bit precision conversion. Figure 11–3 shows the progress being made in data converters. Driven by the pervasion of DSP solutions, this progress spans the range of DSP markets.

Figure 11–3. Mixed Signal Products Enable DSP Solutions



11.2 Mixed Signal Products and DSP Solutions

To assist in the selection of analog-to-digital converters (ADCs) and digital-to-analog converters (DACs), two decision trees are provided in this section.

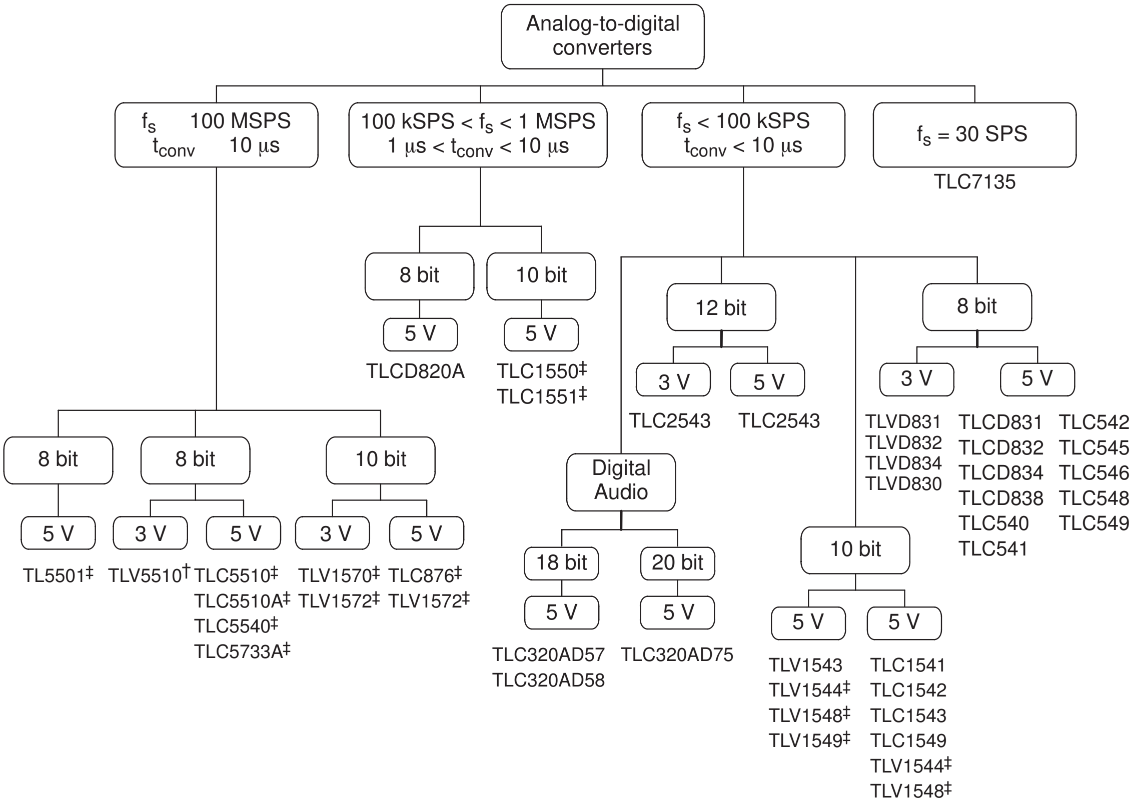
11.2.1 ADC Decision Tree

The choice of ADCs is usually driven by two parameters: speed and resolution. Figure 11–4 is designed to help select the appropriate converter for the desired application.

The first decision is to select the required speed from the three ranges in the ADC decision tree. Next, you select the operating voltage. Finally, you select bits of resolution.

In addition to helping you select an appropriate analog-to-digital converter, Figure 11–4 also illustrates the range of general-purpose ADCs.

Figure 11–4. ADC Decision Tree

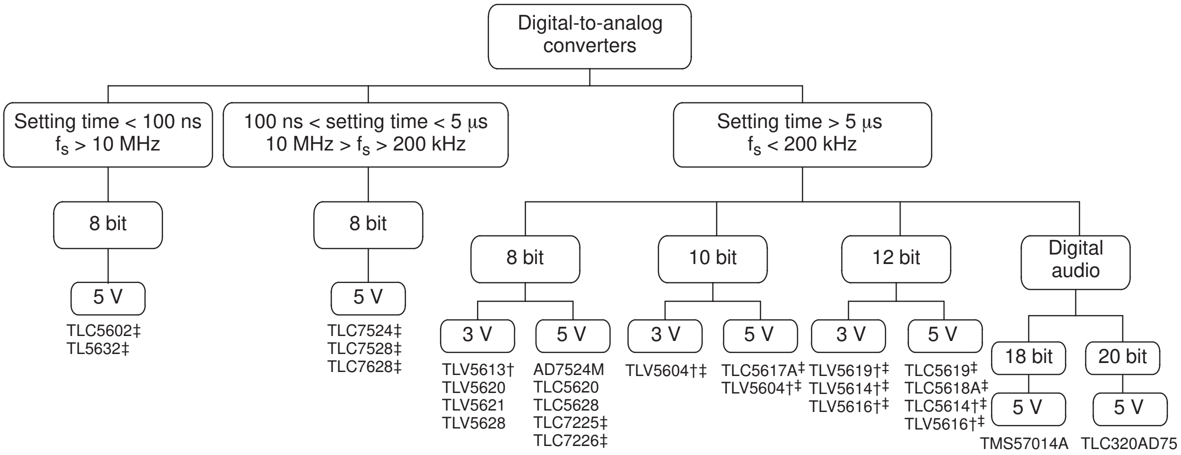


† This part is in the product preview stage of development.
‡ This part is TMS320 compatible.

11.2.2 DAC Decision Tree

Figure 11–5 presents a DAC decision tree with selection based on general applications (general purpose, video graphics, and stereo). The tree is then divided by output type (voltage or current). Although speed is not specifically stated, the fastest devices generally are located in the video graphics path.

Figure 11–5. DAC Decision Tree



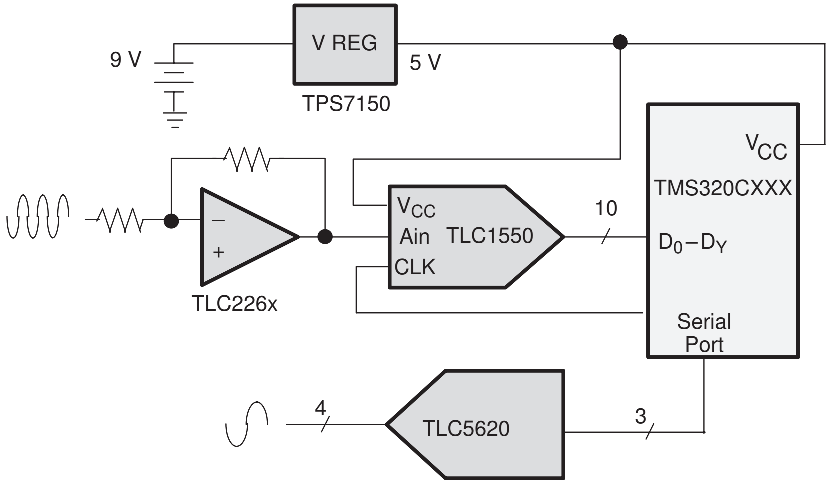
† This part is in the product preview stage of development.
‡ This part is TMS320 compatible.

11.2.3 DSPS: The Total Solution

A DSP Solution for a closed-loop control system can include both an ADC to sense real-world conditions and a DAC to provide proportional feedback or control. Often the requirements for the DAC and ADC are significantly different, requiring separate components for these functions.

Figure 11–6 shows a TLC1550 ADC, which is a 10-bit single-channel converter interfaced through a parallel port to a TMS320Cxxx DSP. The feedback path is from the DSP through a serial port to a TLC5620, which is a quad eight-bit DAC.

Figure 11–6. An Example of a DSP Solution



11.3 TLC1550/51 Analog-to-Digital Converters

The TLC1550 and TLC1551 10-bit, 6-µs ADCs are successive approximation converters that offer a high sample rate (166 ksps) at a low-cost. They feature a parallel interface for speed and a selectable external or internal clock for versatility.

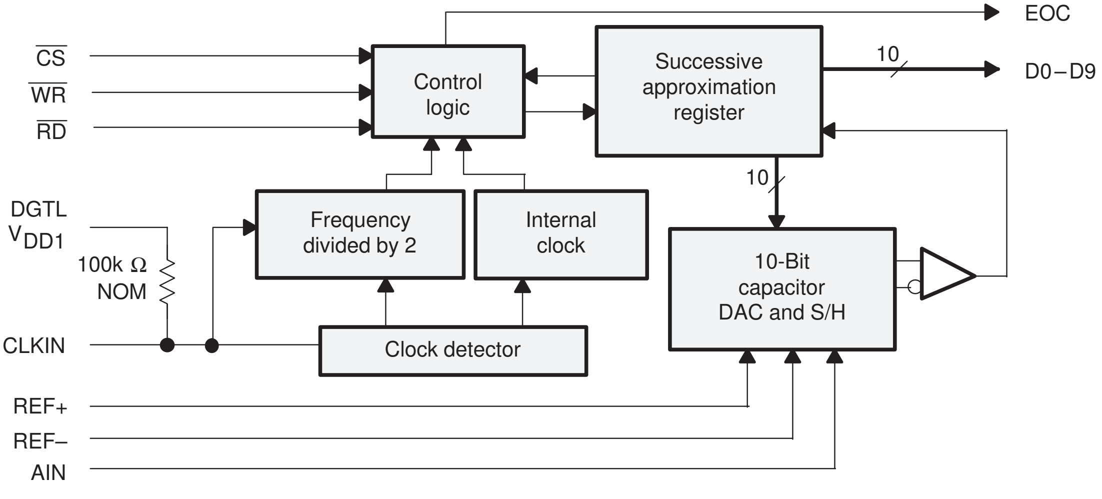
Key features of the TLC1550/51 ADCs include:

- ❑ Single 5-V power supply
- ❑ 3-state outputs
- ❑ Power dissipation of 40 mW maximum
- ❑ Fast parallel processing for DSP and microprocessor interface
- ❑ An external or internal clock
- ❑ DSP/microprocessor interface compatible
- ❑ Conversion time of 6 µs
- ❑ 166-ksps sample rate
- ❑ Total unadjusted error
  - ■ TLC1550, ±0.5 LSB maximum
  - ■ TLC1551, ±1.0 LSB maximum

Applications for these ADCs include vehicle active suspension, data-acquisition systems, DSP front ends, industrial controls, and digital motor control.

Figure 11–7 shows a block diagram of the TLC1550/51 ADC.

Figure 11–7. TLC1550/51 Functional Block Diagram



11.4 TLC5620 Digital-to-Analog Converter

The TLC5620 quad DAC has individual reference inputs for each of the four DACs. This feature enables this circuit to be used as four DACs with separate ranges, as programmable gain blocks, or as a combination of both. The flexibility, simple serial interface, and small (14-pin) package make this circuit well suited to many control-feedback-circuit applications.

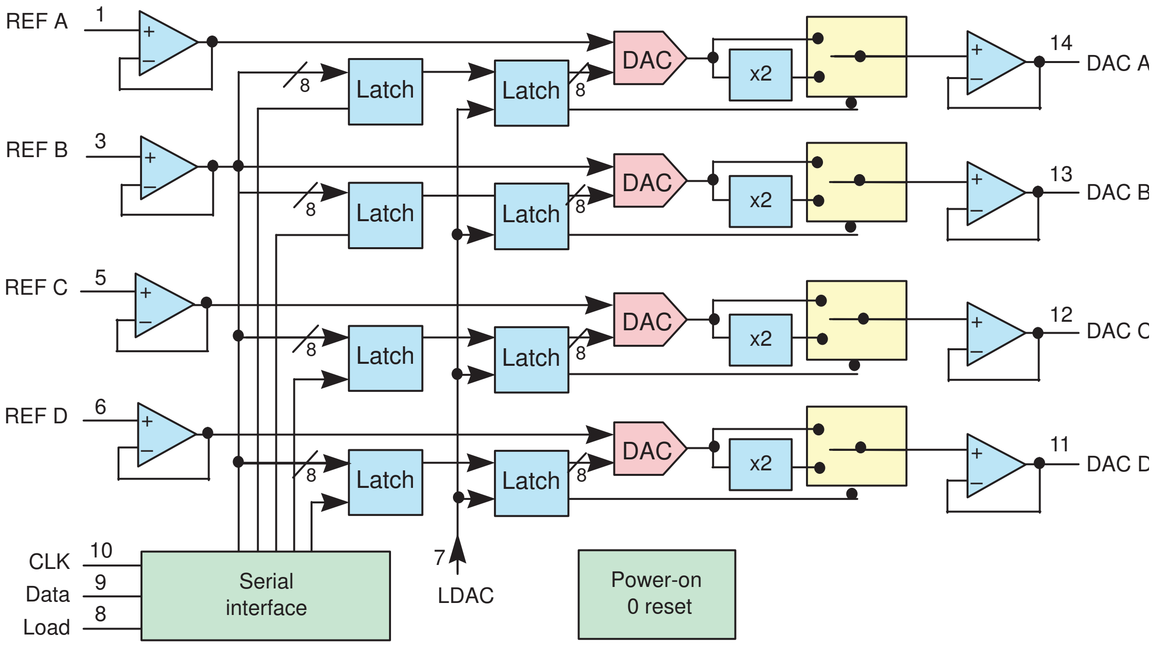
Key features of the TLC5620 DAC include:

- Four 8-bit voltage output DACs
- Individual V_{REF} for each DAC
- Maximum serial input clock rate of 1 MHz
- Buffered reference inputs
- Programmable 1 or 2 times output range
- Double-buffered registers for synchronous updates
- Internal power-on reset
- Low power consumption (20 mW maximum)
- Characterized operating temperatures:
  - TLC5620C 0° to 70°C
  - TLC5620I -40° to 85°C

Applications for the TLC5620 quad DAC include programmable voltage sources, digitally-controlled amplifiers/attenuators, wireless communications, automatic test equipment, process monitoring and control, and signal synthesis.

Figure 11–8 shows a block diagram of the TLC5620 DAC.

Figure 11–8. TLC5620 Block Diagram



11.5 Stereo Audio Converters

The devices listed in this section are stereo audio converters that use the following technologies:

- ❑ Digital-to-analog conversion (DAC)
- ❑ Analog-to-digital conversion (ADC)
- ❑ Analog-to-digital and digital-to-analog conversion (ADA)

11.5.1 TMS57014A Dual 16-/18- Bit Audio DAC

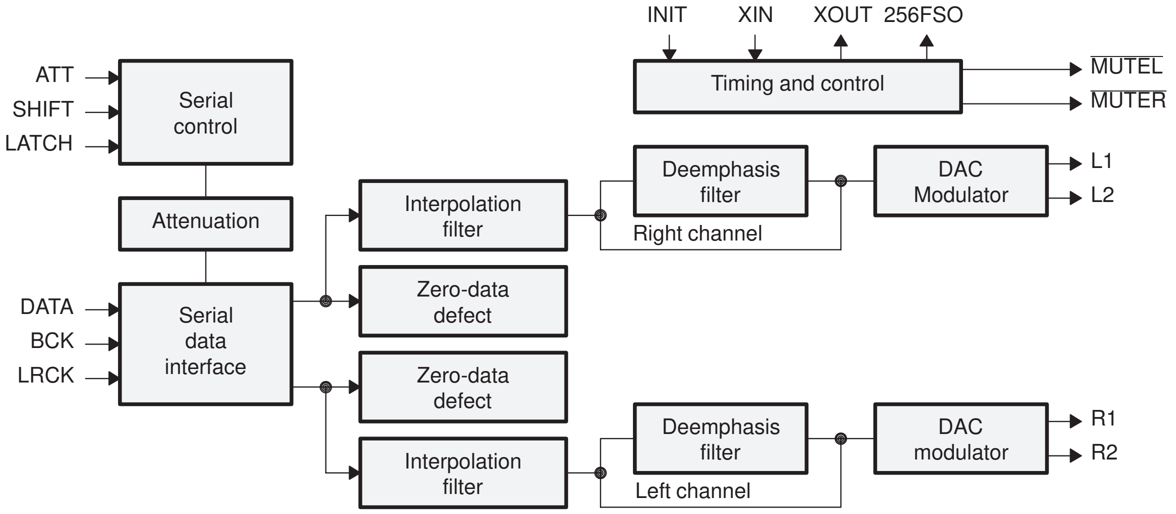
Key features of the TMS57014A include:

- ❑ Sample rates up to 48 kHz
- ❑ De-emphasis filter for 32, 37.8, 44.1, and 48 kHz
- ❑ Digital soft mute to –60 dB
- ❑ Single 5-Vdc supply
- ❑ 18-bit resolution
- ❑ Mute capability with zero data detect flags
- ❑ 2s-complement data format
- ❑ Pulse-width-modulation (PWM) output
- ❑ Serial port interface

Applications of the TMS57014A include digital audio systems, professional quality audio, high-performance audio-speed DAC, waveform generation, and automated test systems.

Figure 11–9 shows a block diagram of the TMS57014A DAC.

Figure 11–9. TMS57014A Dual 16-/18-Bit Audio DAC



11.5.2 TLC320AD57/58 Stereo Sigma-Delta ADCs

The TLC320AD57 and TLC320AD58 are stereo ADCs using the sigma-delta architecture. These converters contain two separate converters with a common control and serial interface.

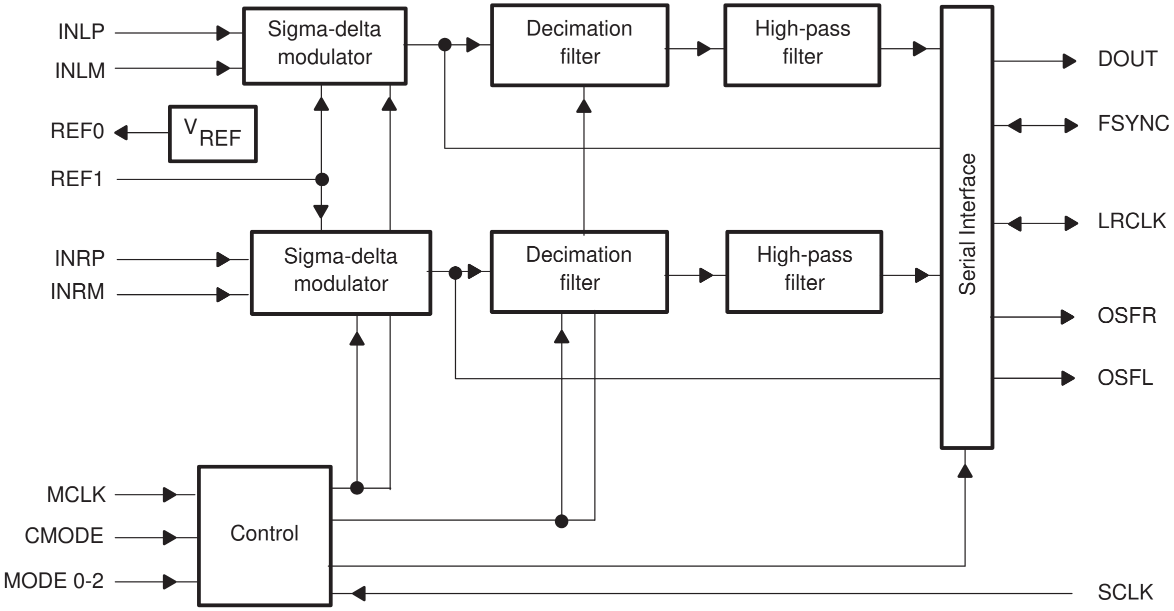
Features of the ’AD57/58 include:

❑ Power supply: Single 5 V
❑ Power-down mode
❑ Serial port interface
❑ 95-dB dynamic range
❑ 93-dB S/N+D
❑ 16-/18-bit resolution
❑ Sample rates to 48 kHz

Applications of the ’AD57/58 include consumer audio, digital radio, industrial process control, multimedia audio, workstations, and DSP analog interface.

Figure 11–10 shows a block diagram of the ’AD57/58 stereo sigma-delta ADC.

Figure 11–10. TLC320AD57/58 Functional Block Diagram



11.5.3 TLC320AD75 High-Performance Stereo ADA

The TLC320AD75 is a high-performance stereo 20-bit ADA converter that uses sigma-delta technology to provide four concurrent 20-bit resolution conversions.

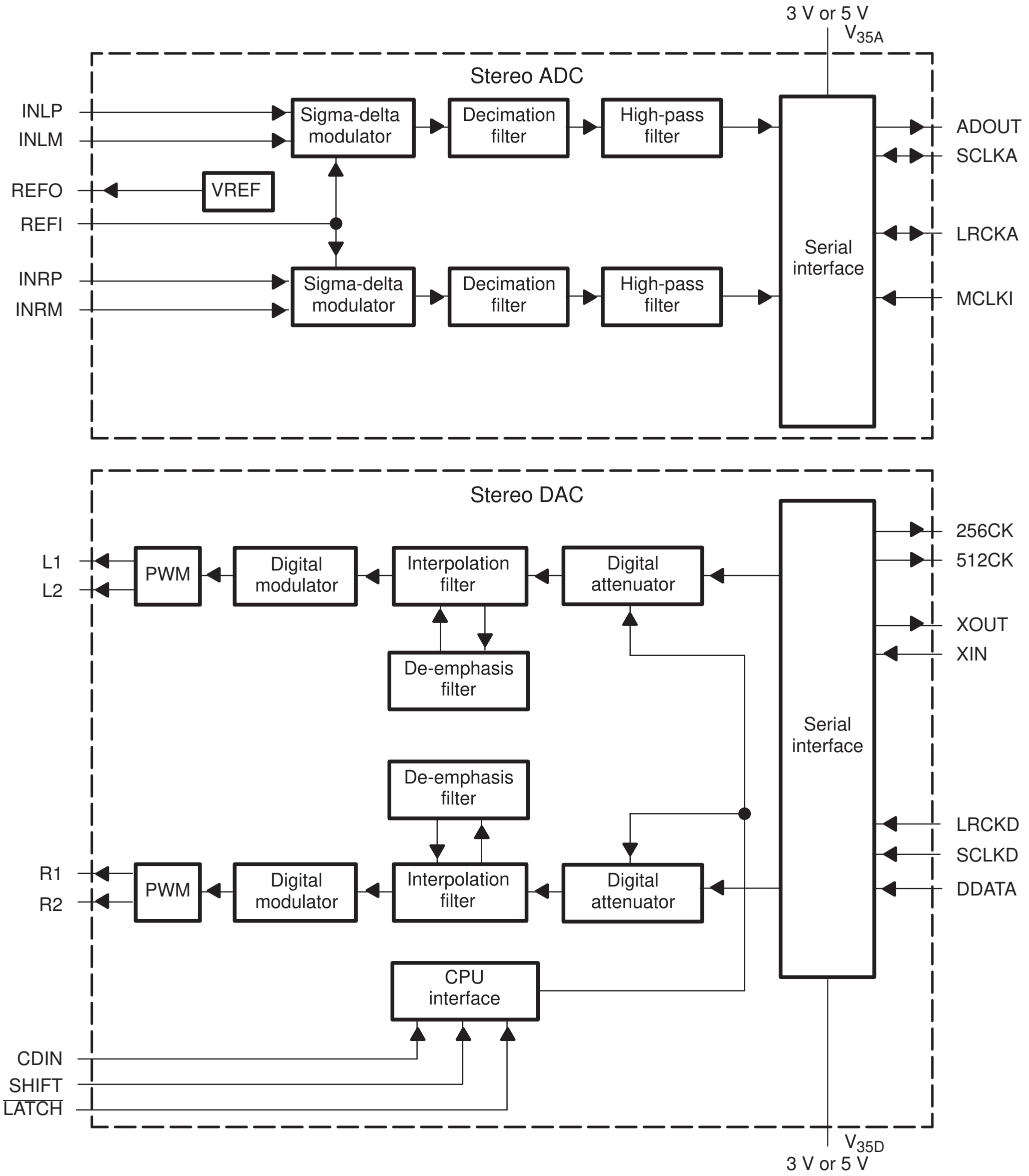
Features of the TLC320AD75 include:

- ❑ Sample rates of up to 48 kHZ
- ❑ Digital de-emphasizing filter
- ❑ Digital tenuation—soft and mute on DAC
- ❑ Signal-to-noise ratio of 100 db (ADC) and 104 db (DAC)
- ❑ Differential architecture
- ❑ Total harmonic distortion and noise—.003% (DAC)

TLC320AD75 applications include consumer audio and high-end business audio.

Figure 11–11 shows a block diagram of the ’AD75 high-performance ADA.

Figure 11–11. TLC320AD75 Functional Block Diagram



11.5.4 TLC320AD80 Stereo Set-Top Box DAC System

The TLC320AD80 is an audio processing subsystem for set-top box applications. This device includes a stereo/audio DAC, analog volume and balance control, analog TV manual decoder, de-emphasis filter, and an analog wideband multiplexer.

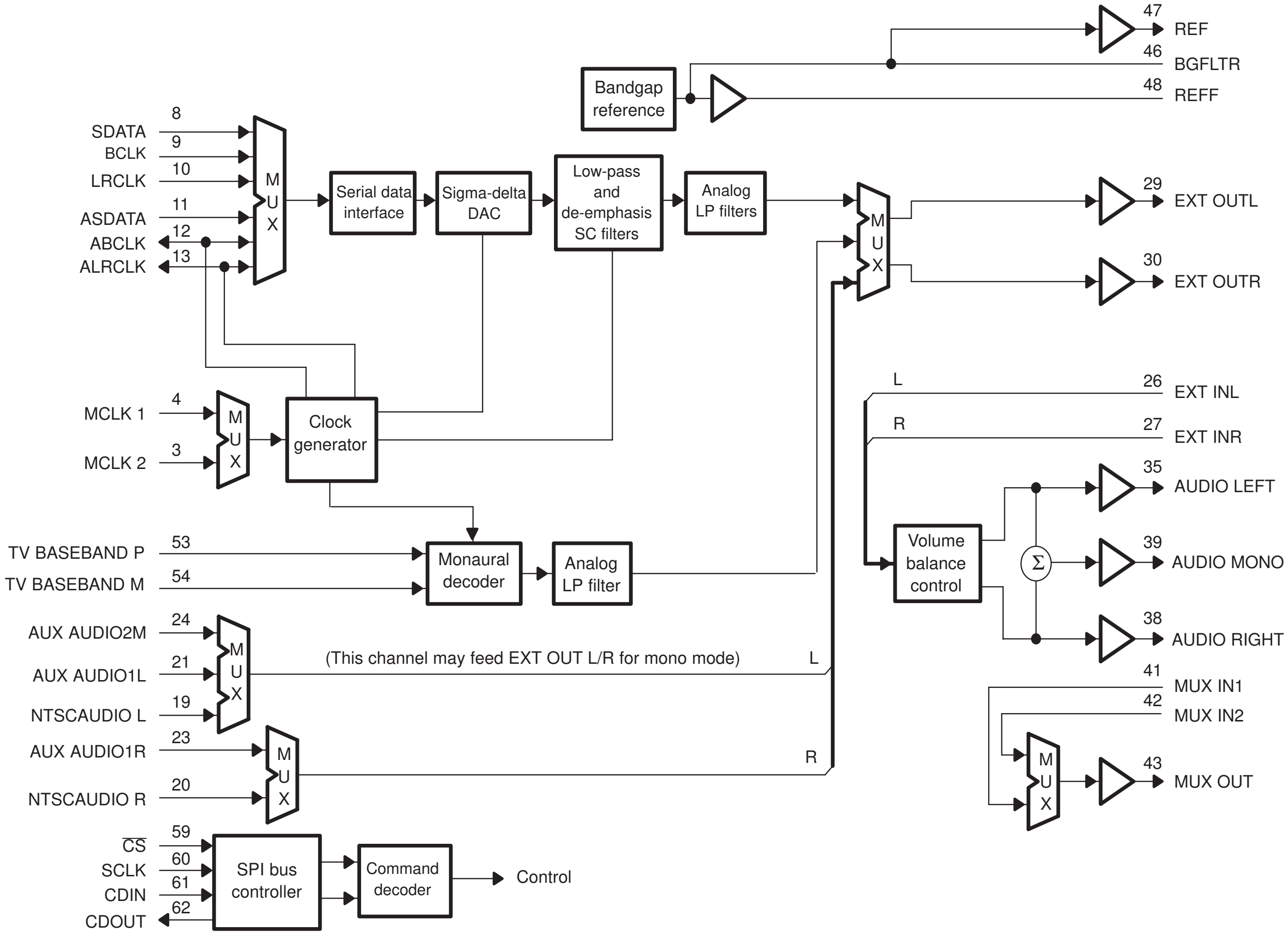
Features of the TLC320AD80 include:

❑ 16-bit sigma-delta stereo/audio DAC
❑ Analog volume and balance control
❑ 50/15 ms de-emphasis filter
❑ Two PCM audioports (16- or 18-bit serial)
❑ SPI compatibility
❑ Power-down mode

TLC320AD80 applications include DBS set-top boxes, high-definition television (HDTV), digital audio broadcast receivers, video laser discs, and video CD.

Figure 11–12 shows a block diagram of the ’AD80 audio processing subsystem.

Figure 11–12. TLC320AD80 Functional Block Diagram



11.6 Analog Interface Circuits (AICs)

AICs are described in this section, with block diagrams and features of each one.

Figure 11–13. Analog Interface Cicuits

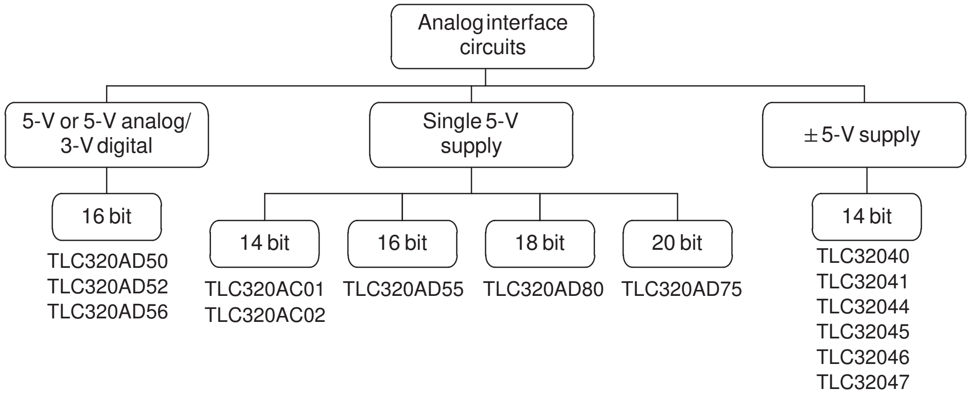


Table 11–1. Analog Interface Circuits (AICs)

Device	Resolution (bits)	Sampling Rate (kHz)	Bandwidth (kHz)	Supply Voltage(s) (V)	P_d (mW) typ	SNR (dB) typ	Conversion Method	Description
TLC320AD75	20	44.1	0.002 – 20	5	400	104	Sigma-delta	Stereo ADA circuit (audio)
TLC320AD80	18	48		5			Sigma-delta	Audio processor subsystem
TLC320AD56	16	22.05	8.8	5/3	100	70	Sigma-delta	Sigma-delta AIC
TLC320AD50	16	22.05	8.8	5/3	175	70	Sigma-delta	Sigma-delta AIC with mstr/slv function
TLC320AD52	16	22.05	8.8	5/3	175	70	Sigma-delta	Sigma-delta AIC with mstr/slv function
TLC320AD55	16	10.3	4	5	150	70	Sigma-delta	Sigma-delta AIC
TLC320AC02	14	25	10.8	5	100	70	Succesive approx.	Single-supply AIC
TLC320AC01	14	25	10.8	5	100	72	Succesive approx.	Single-supply AIC
TLC32047	14	25	0.3 – 11.4	5	375	70	Succesive approx.	Wide-band AIC with (sin x)/x correction
TLC32046	14	25	0.3 – 7.2	5	375	85	Succesive approx.	AIC with (sin x)/x correction

Table 11–1. Analog Interface Circuits (AICs) (Continued)

Device	Resolution (bits)	Sampling Rate (kHz)	Bandwidth (kHz)	Supply Voltage(s) (V)	P_d (mW) typ	SNR (dB) typ	Conversion Method	Description
TLC32045	14	19.2	0.1 – 3.8	5	375	80	Succesive approx.	Voice-band AIC (relaxed TLC32044)
TLC32044	14	19.2	0.1 – 3.8	5	375	80	Succesive approx.	Voice-band AIC
TLC32041	14	19.2	0.3 – 3.6	5	375	89	Succesive approx.	AIC w/o internal reference
TLC32040	14	19.2	0.3 – 3.6	5	375	89	Succesive approx.	AIC

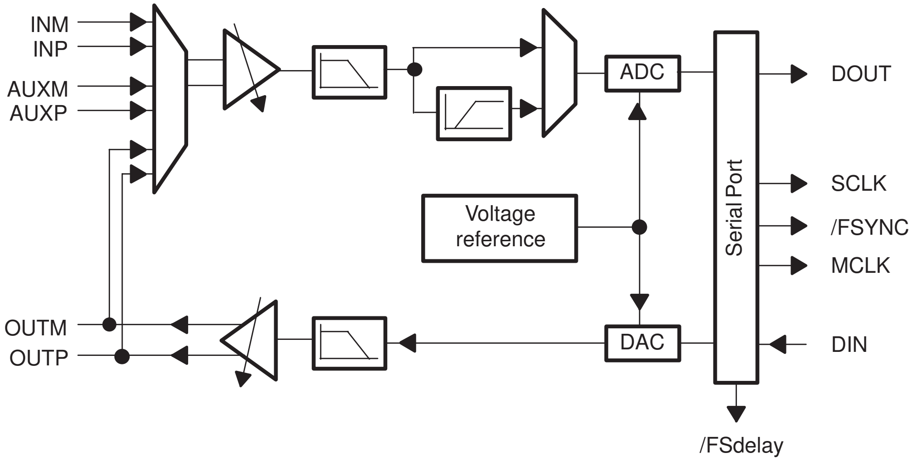
11.6.1 TLC320AC01/02 AICs

The analog-interface circuit (AIC) is a complete analog-interface system. The system consists of an ADC with an input amplifier multiplexer and antialiasing filter, a DAC path with a reconstruction filter and output amplifier, and the necessary serial data interfaces.

The AIC products were originally designed for modem-type applications; however, they are currently being used with DSPs in applications such as speech processing, speech analysis, voice encryption/decryption, modems, and industrial-process controls.

Figure 11–14 shows a block diagram of the TLC320AC01/02 AIC.

Figure 11–14. TLC320AC01/02 Block Diagram



11.6.2 TLC320AD50 Sigma-Delta AIC

The TLC320AD50 is analog interface circuit (AIC) is a versatile 16-bit sigma-delta analog front end for modem and business applications. It provides high resolution conversion using an oversampling technique.

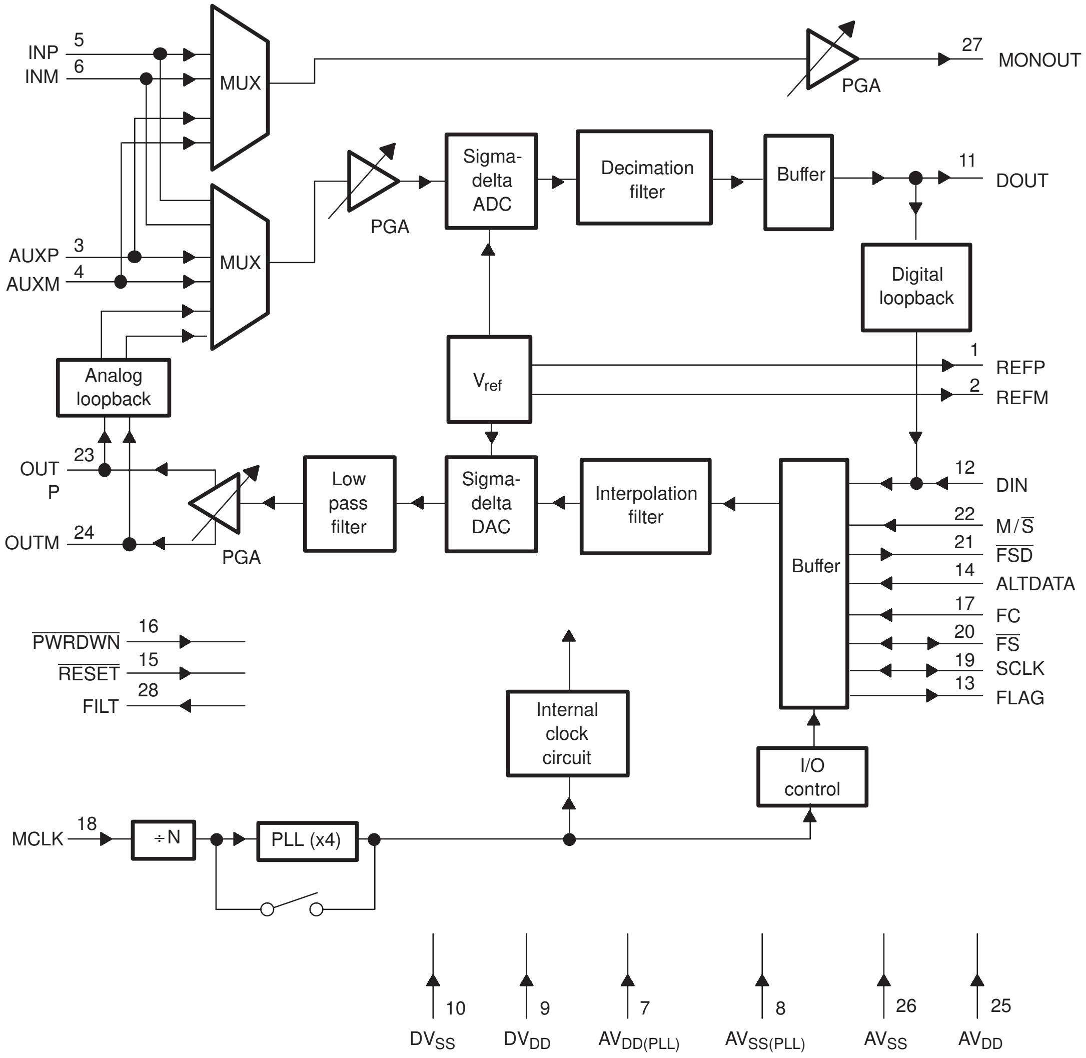
Features of the ’AD50 include:

❑ Single 5 V supply or 5 V analog/3 V digital
❑ Master/slave mode (supports up to 3 slaves)
❑ 16-bit ADC and DAC
❑ Differential output drives (600 ohm load)
❑ Power-down mode
❑ Glueless DSP interface
❑ Gain control
❑ System test mode

’AD50 applications include modems (V.34, DSVD, telephony), personal computer memory card international association (PCMCIA) fax modems, DSP analog interface, noise suppression/cancellation, and industrial process control.

Figure 14–14 shows a block diagram of the TLC320AD50 sigma-delta AIC.

Figure 11–15. TLC320AD50 Functional Block Diagram



11.6.3 TLC320AD55 Sigma-Delta AIC

The TLC320AD55 is a very high performance AIC. A sigma-delta architecture is used to achieve the high performance. This circuit provides the functionality and performance necessary to implement applications such as the V.34 modem.

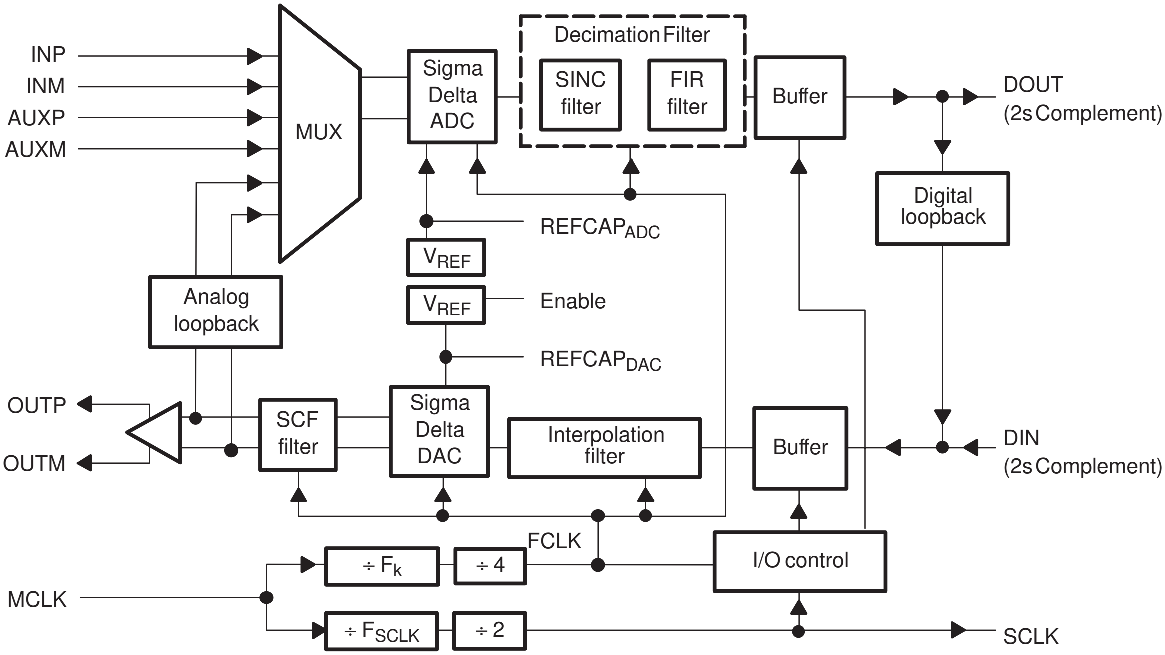
Key features of the TMC320AD55 include:

❑ Power supply: single 5 V
❑ Power-down mode
❑ Serial-port interface
❑ Antialiasing and anti-imaging filters
❑ 89-dB dynamic range
❑ 80-dB signal-to-noise (S/N) ratio
❑ 16-bit resolution
❑ Output gain control

TLC320AD55 AIC applications include V.34 modems, DSP analog interfaces, noise-cancellation techniques, speech processing, and industrial process control. The TLC320AD55 has a 28-pin SOIC (DW) package.

Figure 11–16 shows a block diagram of the TLC320AD55 sigma-delta AIC.

Figure 11–16. TLC320AD55 Block Diagram



11.6.4 TLC320AD56 Sigma-Delta AIC

The TLC320AD56 is a cost-reduced version of the ’AD55. The cost reduction was accomplished by removing the switched capacitor filter in the DAC section, reducing the chip size and cost. Depending on the specific application, an external filter may be required.

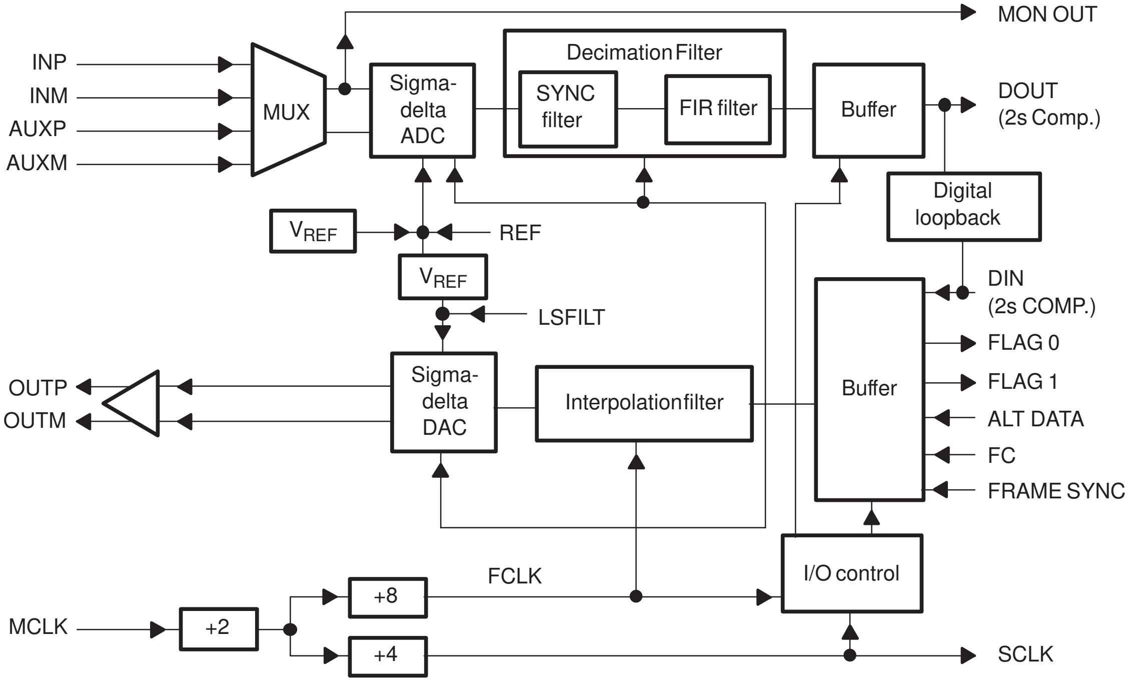
Features of the ’AD56 include:

- ❑ Power supply: single 5-V or 5-V analog/3-V digital
- ❑ Power-down mode to <1 mW
- ❑ Serial-port interface
- ❑ 91-dB dynamic range
- ❑ 88-dB S/N+D (ADC)
- ❑ 85-dB S/N+D (DAC)
- ❑ 16-bit resolution
- ❑ Cost-reduced solution

Applications of the ’AD56 include V.34 modems, DSP analog interfaces, noise-cancellation techniques, speech processing, industrial process control, and business audio. The ’AD56 has 28-pin PLCC and 48-pin SQFP (PCMCIA) packages.

Figure 11–17 shows a block diagram of the TLC320AD56 sigma-delta AIC.

Figure 11–17. TLC320AD56 Functional Block Diagram



Chapter 12

Customizable DSPs (cDSP)

Texas Instruments is the world leader in DSP solutions with offerings that suit a wide range of applications. Customizable DSPs utilize high-performance industry-standard cores with leading edge standard cell and gate-array application-specific integrated-circuit (ASIC) technology to enable a custom solution for your application requirements. With more than five years of experience shipping cDSP technology in volume, TI offers you its TMS320 family core architecture, a proven leader worldwide with more than 20,000 customers of mass market and custom products.

TI's exclusive cDSP technology enables high-performance full-integrated DSP systems on a single chip. The crucial parts of this solution are the TMS320 DSP cores and all peripherals, software and support needed to create the cost-effective, high-performance chip our customers demand. TImeline Technology is the name given to the TI integration technology that is used on customizable DSPs of up to 125-million transistors. This technology can cut development time in half.

The cDSP solution is now used in an array of industries and products, including automotive, cellular telephones, modems, CD-ROMs, and hard disk drives. The task of creating a cDSP device means your design team and TI's design team are committed to work together to meet your design goals.

Topic **Page**

12.1 Attributes

Various combinations of cores, memory blocks, peripherals, and ASIC logic are available through TI's cDSP and ASIC libraries to build a cDSP device configured to your specifications. Cores are optimized to minimize size and power consumption while maximizing processing power and ease of use. Presently, there are three proven TMS320 family cores available: the TMS320C2xLP, T320C52, and T320C54x. In addition to core memory, numerous selections of on-chip memory are available, including RAM, ROM, flash, and DRAM. The cDSP option offers an array of peripherals targeted at the wireless, hard disk drive, multimedia, and automotive markets. The complete ASIC library is available to integrate your custom design functions onto the die of the cDSP.

TI defines customizable DSPs as the ability to integrate on a single chip:

- An industry-standard DSP core ('C2xx, 'C5x, 'C54x…)
- The user's choice of memory (RAM, ROM, flash)
- Custom logic gates
- Analog modules (ADC, DAC)
- Software and hardware macros

For instance, TI supplies about 95 percent of the DSPs found in hard disk drives today, and because a single DSP can replace multi-component designs, HDD manufacturers have been able to cut the cost of drive electronics by more than 30 percent.

Development time for an HDD is just six months in today's competitive environment, followed by a short product life cycle of about 18 months. This makes time-to-market critical to the success of the new products. TI has demonstrated that it is able to move cDSP solutions for HDD designs from samples to volume production in less than three months.

12.2 Benefits

The goal of TI’s cDSP solution is to develop a unique product with enhanced system integration and performance, reduced cost, low power consumption and, most importantly, fast time to market. By consolidating system memory, logic, and peripherals within a single chip, cDSP customers save significant board space and reduce cost. Power consumption is lowered by reducing the system loading from several devices and their interconnection to a single cDSP. In addition, consolidation will increase system performance due to reduced access times and enhanced architecture. To minimize your time to market, the cDSP design flow is accompanied by extensive tools and support every step of the way. From a starter kit that allows you to sample the power of DSP products to the in-depth analysis of an IEEE Standard 1149.1 (JTAG)-compliant scan port, you are empowered with the tools and technical support needed to get your product to market quickly. By defining a cDSP solution to meet your exact system requirements, you can clearly differentiate your product from those of your competitors.

12.3 Key Features

Presently, there are three proven TMS320 family cores available:

DSP Cores Available	ASIC Family	Performance	Power Consumption (3/5 V)
TMS320C2xLP	TxC2000, TxC4000	30 and 40 MIPS	1.1/1.9 mA/MIPS
T320C52	TxC2000	40 and 50 MIPS	1.4/2.0 mA/MIPS
T320C54x	TxC4000, TSC5000	40 and 60 MIPS	1.5/2.5 mA/MIPS

The cDSP option offers many peripherals needed by DSP systems. The peripherals listed below are available or planned for development.

Peripherals:

❑ Analog switch
❑ ASIC PLL (APLL)
❑ Host port interface (HPI)
❑ 16-bit timer
❑ Synchronous serial port
❑ Serial-port interface (SPI)
❑ Keyboard interface
❑ Serial-control interface (SCI)
❑ Pulse-width modulation (PWM)
❑ Universal asynchronous receiver/transmitter (UART)
❑ Real-time clock
❑ Interface controller
❑ LCD controller
❑ Buffered serial port (BSP)
❑ General-purpose chip selects
❑ Maskable interrupts
❑ Synchronous serial port
❑ Asynchronous serial port
❑ Event timer
❑ Watchdog timer
❑ DMA controller

You can specify any of the available memory options listed below to meet your system specifications. All three cores can address up to 64K 16-bit words of program memory, 64K 16-bit words of local data memory, 32K 16-bit words of global data memory, and 64K 16-bit words of I/O ports for a total of 224K words address reach.

Memory type:

- ❑ Single-access RAM
- ❑ ROM
- ❑ Flash
- ❑ DRAM

Because cDSP is built into TI ASIC backplane technology, you benefit from all the features that are part of the associated TI ASIC library, including:

- ❑ 8-bit ADC
- ❑ 10-bit ADC
- ❑ 8-bit DAC
- ❑ ASIC logic
- ❑ General-purpose and specialized I/Os

Chapter 13

Code Generation Tools

Texas Instruments supports designers in the complete development of their applications, from concept to production. This results in fast time to market, design ease, and increased productivity. TI development support products include the code-generation tools discussed in the sections listed below.

Topic **Page**

13.1 TMS320 Optimizing ANSI C Compilers

Speedy development and code maintenance over the life cycle of a product are concerns that all developers share. TI supports DSP developers with a family of optimizing compilers for the TMS320 DSPs. TMS320 optimizing ANSI C compilers translate ANSI standard C language files into highly efficient TMS320 assembly language source files, which then are input to a TMS320 assembler/linker. All TMS320 compilers have been validated for their conformance to the ANSI C specification, using the industry-standard Plum-Hall test suite.

TI offers optimizing ANSI C compilers that support the ’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x, and ’C8x devices, and are complemented by the standard TMS320 programmer’s interface for debugging C and assembly source code. The C compilers produce a rich set of information for the debugger, which allows source-level debugging in C. This enhances productivity and shortens the development cycle for both fixed-point and floating-point applications.

This section discusses five different compilers that have similar capability but different target processors:

❑ The TMS320 ’C2xx/’C5x optimizing ANSI C compiler is a full-featured C compiler that supports the ’C2xx and ’C5x product families.

❑ The ’C54x optimizing ANSI C compiler supports the ’C54x. The ’C2xx/’C5x and the ’C54x compilers are referred to as the fixed-point compilers.

❑ The TMS320 ’C3x/’C4x floating-point optimizing ANSI C compiler is a full-featured C compiler that supports the ’C3x and ’C4x. Throughout this book, this compiler is referred to as the floating-point compiler*.*

❑ The ’C6x optimizing ANSI C compiler offers many global and local code optimizations and supports both the fixed-point ’C62x product and the floating-point ’C67x product.

❑ The ’C80 compiler is actually two compilers, one for the master processor and one for the parallel processor.

The TMS320 optimizing ANSI C compilers feature an ANSI-standard, runtime-support library (a source interlist utility to enable customers to associate assembly language statements with the C code that produced them), and a shell program that facilitates one-step translation from C source to TMS320 object-code files. The object code is then ROM-able, relocatable, and reentrant.

The benefits of ANSI C support are:

❑ **Standardization**. C, although one of the most portable programming languages, has lacked standardization, particularly in the area of common extensions to the language. ANSI provides a standard for these extensions.

❑ **Compatibility**. In general, the ANSI C standard is a superset of the Kernighan and Ritchie (K&R) standard. Most C programs that compile and run under a K&R compiler (including earlier releases of the TMS320 C compilers) should be capable of running under the new optimizing ANSI compiler. The few cases of obscure, obsolete, or questionable program constructions can be easily rewritten for ANSI compliance. ANSI compatibility also enhances portability; that is, existing code written for another processor can be ported into the TMS320 C compiler with little or no additional coding.

❑ **New types**. The new *const* and *volatile* types allow improved optimizations.

❑ **Improved function conventions**. Function prototypes allow improved type checking and enable optimization of calling conventions.

The optimizing ANSI C compilers were designed with three major efficiency goals in mind:

❑ Producing compiled, general-purpose C code that compares favorably to hand-coded assembly language

❑ Providing a simple and accessible programming interface to the C runtime environment so that time-critical DSP algorithms demanding extreme performance can be implemented in assembly language

❑ Establishing a comprehensive, easy-to-use tool set for the development of high-performance DSP applications in C

13.1.1 ’C2xx/’C5x Fixed-Point Compiler

The ’C2xx/’C5x compiler enhances productivity by enabling you to program in C. C code is easier to prototype, debug, and benchmark than assembly language.

The fixed-point compilers translate the widely used ANSI C language directly into highly optimized assembly language for either a ’C2xx or ’C5x device, according to a command line switch. Once an algorithm is coded in C, the TMS320 C compiler generates the appropriate assembly code, which is then assembled and linked by the TMS320 assembler and linker.

The fixed-point C compiler and assembly language tools support modular programming by allowing you to compile and assemble individual modules and then link them together. The fixed-point C compilers also perform both global optimizations and loop optimizations, such as strength reduction.

13.1.2 ’C54x Fixed-Point Compiler

The ’C54x compiler enhances productivity by enabling you to program in C. C code is easier to prototype, debug, and benchmark than assembly language.

The ’C54x compiler translates the widely used ANSI C language directly into highly optimized assembly language for the ’C54x. Once an algorithm is coded in C, the TMS320 C compiler generates the appropriate assembly code, which is then assembled and linked by the TMS320 assembler and linker.

The ’C54x compiler and assembly language tools support modular programming by allowing you to compile and assemble individual modules and then link them together. The ’C54x compiler also performs both global optimizations and loop optimizations, such as strength reduction.

13.1.3 ’C3x/C4x Floating-Point Compiler

The ’C3x/C4x floating-point compiler supports the ’C3x and ’C4x devices and performs both global optimizations and loop optimizations, such as strength reduction. Additionally, it thoroughly analyzes code to optimize the use of memory and register variables. It also searches for vector/matrix operations by mapping memory accesses to those ’C4x addressing modes that were optimized for the vector and matrix operations.

For the ’C40, a parallel-processing, runtime-support library uses the communication ports and DMA controller of the ’C4x. This simplifies the use of this device’s powerful features for multiprocessor applications.

The floating-point compilers have a big memory model with unlimited space for global data, static data, and constants. In the small (default) model, this space is limited to 64K words for faster, more efficient coding/execution.

The floating-point compiler for the ’C30 also has a short multiplication option that generates efficient MPYI instructions. These are 24 × 24-bit multiplication on the ’C3x, yielding a 32-bit resultant for integer multiplications instead of runtime-support calls.

13.1.4 ’C6x C Compiler

The ’C6x compiler supports fixed-and floating-point code development, providing a single tool set for the ’C62x and ’C67x architectures. The highly efficient ’C6x C compiler takes advantage of the silicon’s advanced VLIW architecture to produce code that executes up to three times faster than code produced by other DSP compilers.

The C compiler accepts ANSI C source code and produces efficient ’C6x assembly language source code, by performing a wide variety of optimizations to improve the efficiency of the compiled code. The ’C6x compiler offers a variety of intrinsics to provide even higher performance that that achievable through C alone.

13.1.5 ’C8x Multiprocessing Compilers

The C compilers for the single-chip, multiprocessing ’C8x support both the on-chip parallel-processing DSPs (PPs) and the RISC master processor (MP). This allows programmers to develop sophisticated applications in a high-level language such as C and then optimize their code in assembly. The C compilers allow for easy implementation of data and message passing between tasks (or processors) in parallel-processing systems.

The compiler package includes a 4-level optimization program that improves the execution speed and reduces the size of C programs by simplifying loops, rearranging statements and expressions, and allocating variables into registers.

13.1.6 Key Features of TMS320 Optimizing ANSI C Compilers

Key features of the TMS320 optimizing ANSI C compilers include:

- ❑ Complete and exact conformance with the ANSI C specification
- ❑ Highly efficient code. The compilers incorporate state-of-the-art generic and target-specific optimizations (described in detail in the following subsections).
- ❑ ANSI standard runtime-support library
- ❑ ROM-able, relocatable, and reentrant code
- ❑ The ability to link C programs with assembly language routines, allowing hand coding of time-critical functions in assembly language
- ❑ A full-featured, flexible linker that allows total control over memory allocation, memory configuration, and partial linking and contains features that allow easy runtime relocation of code

- A C shell program that facilitates one-step translation from C source to executable code
- Fast compilation to increase productivity
- Unlimited symbol table space (up to the amount of available host memory)
- Complete and useful diagnostics (error messages)
- An archiver utility that allows you to collect files into a single archive file or library by adding new files or by extracting, deleting, or replacing files. You can use a library of object files as input to the linker.
- Ability to expand inline both runtime-support and user-defined functions
- A utility that builds object libraries from source libraries
- A variety of listing files, including:
  - Assembly-source file, which can optionally include interlisted, C source code as well as register-usage information
  - Preprocessed output file useful for separating preprocessing/parsing (if memory limitations dictate) and for troubleshooting macro definitions
  - Assembly-listing file with line numbers and opcodes

The TMS320 C compilers have passed all Plum-Hall validation suites, which are a series of routines that test the validity and conformability of a C compiler. Plum-Hall validation is the *de facto* standard for validating ANSI C compilers.

13.1.7 TMS320 ANSI C Compiler Optimizations

The efficiency of a C compiler depends on the scope and number of optimizations the C compiler performs, as well as on the application. The TMS320 C compilers perform a wide variety of optimizations to improve the efficiency of the compiled code. The following list and the explanations in Section 13.1.7.1 to Section 13.1.7.3 describe some of the optimizations and highlight particular strengths of the C compilers.

- General-purpose C optimizations
  - Algebraic reordering, symbolic simplification, constant folding
  - Alias disambiguation
  - Data flow optimizations
    - Copy propagation
    - Common subexpression elimination
    - Redundant assignment elimination

  - Branch optimizations/control-flow simplification
  - Loop induction variable optimizations, strength reduction
  - Loop rotation
  - Loop-invariant code motion
  - Inline expansion of function calls

- Fixed-point compiler optimizations
  - Register variables
  - Cost-based register allocation
  - Autoincrement addressing modes
  - Repeat blocks
  - Delayed branches, calls, and returns
  - Arranging of variables on the local frame
  - Elimination of unnecessary LDPK instructions

- Floating-point compiler optimizations
  - Register variables
  - Register tracking/targeting
  - Cost-based register allocation
  - Autoincrement addressing modes
  - Repeat blocks
  - Delayed branch instructions
  - Use of registers for passing function arguments
  - Parallel instructions
  - Conditional instructions
  - Loop unrolling

13.1.7.1 General-Purpose Optimizations

Differences in the same optimizations for fixed-point and floating-point compilers are highlighted in the examples.

- Algebraic reordering, symbolic simplification, constant folding

  For optimal evaluation, the compiler simplifies expressions into equivalent forms requiring fewer instructions or registers. For example, the expression $(a + b) – (c + d)$ requires more instructions and registers to evaluate than the equivalent expression $((a + b) – c) – d$. Operations between constants are folded into single constants. For example, $a = (b + 4) – (c + 1)$ becomes $a = b – c + 3$. See Example 13–1.

- Alias disambiguation

  Programs written in C generally use many pointer variables. Frequently, compilers are unable to determine whether two or more l (lowercase L) values (symbols, pointer references, or structure references) refer to the same memory location. This aliasing of memory locations often prevents the compiler from retaining values in registers, because it cannot be sure that the register and memory continue to hold the same values over time. Alias disambiguation is a technique that determines when two pointer expressions cannot point to the same location, allowing the compiler to freely optimize such expressions.

- ❑ Data flow optimizations

  Collectively, the following three data flow optimizations replace expressions with less costly ones, detect and remove unnecessary assignments, and avoid operations that produce values already computed. The compiler performs these data flow optimizations both locally (within basic blocks) and globally (across entire functions). See Example 13–1 for fixed-point compilers and Example 13–2 and Example 13–3 for floating-point compilers.

  - ■ Copy propagation

    Following an assignment to a variable, the compiler replaces references to the variable with its value. The value could be another variable, a constant, or a common subexpression. This may result in increased opportunities for constant folding, common subexpression elimination, or even total elimination of the variable.

  - ■ Common subexpression elimination

    When the same value is produced by two or more expressions, the compiler computes the value once, saves it, and reuses it.

  - ■ Redundant assignment elimination

    Often, copy propagation and common subexpression elimination optimizations result in unnecessary assignments to variables (variables with no subsequent reference before another assignment or before the end of the function). The compiler removes these dead assignments.

Example 13–1. Data Flow Optimizations for Fixed-Point Compilers

```
simp(int j)
{
     int a = 3;
     int b = (j*a)+(j*2);
     int c = (j<<a);
     int d = (j>>3)+(j<<b);

     call(a,b,c,d);
     ...
}
```

Fixed-point compiler (’C5x option) output is:

```
_simp:
     ...
****************************************************************
*    b = j*5;
****************************************************************
     LARK    AR2,-3+LF1      ;AR2 = &j
     MAR     *0+
     LT      *               ;T = *AR2
     MPYK    5               ;P = T * 5
     ADRK    4-LF1           ;AR2 = &b
     SPL     *               ;*AR2 = P
****************************************************************
*    call(3, b, j << 3, (j >> 3) + (j << b));
****************************************************************
     LT      *               ;T = *AR2
     SBRK    4-LF1           ;AR2 = &j
     LACT    * ,AR1          ;ACC = j<<b
     SACL    * ,AR2          ;save off ACC on TOS (top of stack)
     SSXM                    ;need sign extension for right shift
     LAC     * ,12,AR1       ;high ACC = j>>3
     ADD     * ,15           ;add TOS to high ACC
     SACH    *+,1,AR2        ;stack high ACC
     LAC     * ,3,AR1        ;ACC = j<<3
     SACL    *+,AR2          ;stack ACC
     ADRK    4-LF1           ;AR2 = &b
     LAC     * ,AR1          ;ACC = b
     SACL    *+              ;stack ACC
     CALLD   _call           ;call begins
     LACK    3               ;ACC = 3
     SACL    *+              ;stack ACC
     CALL    _call           ;call occurs
     ...
```

The constant 3, assigned to a, is copy-propagated to all uses of a; a becomes a dead variable and is eliminated. The sum of multiplying j by 3 (the value of a) and by 2 is simplified into b = j*5, which is recognized as a common subexpression. The assignments to c and d are dead and are replaced with their expressions. These optimizations are performed across jumps.

Example 13–2. Data Flow Optimizations for Floating-Point Compilers

```
simp(int j)
{
   int a = 3;
   int b = (j * a) + (j * 2);
   int c = (j << a);
   int d = (j >> 3) + (j << b);

   call(a,b,c,d);
   ...
}
```

Floating-point compiler output is:

```
_simp:
*
* RC   is allocated to user var ’j’
* RS   is allocated to temp var ’T$2’
* RE   is allocated to temp var ’T$1’
*
       ...
       LDI     2,R0            ; (j*a + 2j) == (3j + 2j) == (5j) ==(4j + j)
       LSH     R0,RC,R1        ; R1 = (4j) == (j << 2)
       ADDI    R1,RC,RE        ; b = (4j + j) == 5j
       LDI     3,R1            ; load shift count
       LSH     R1,RC,RS        ; c = (j << a) == (j << 3)
       LSH     RE,RC,R2        ; R2 = (j << b)
       ADDI    RS,R2,R3        ; R3 = (j << b) + (j << a)
       PUSH    R3              ; push R3 (d)
       PUSH    RS              ; push c
       PUSH    RE              ; push b
       PUSH    R1              ; push a (tracked in R1)
       CALL    _call
       ...
```

The constant 3, assigned to a, is copy-propagated into all uses of a. a becomes a dead variable and is removed completely. The sum of multiplying j by 3 (a) and by 2 is simplified into a multiply by 5, which is computed with a shift and add. The expression (j << a) is computed once for assignment to c and then reused for calculating d. These optimizations are also performed across jumps.

Example 13–3. Copy Propagation and Control-Flow Simplification for Floating-Point Compilers

```
fsm()
{
     enum { ALPHA, BETA, GAMMA, OMEGA } state = ALPHA;
     int *input;

     while (state != OMEGA)
           switch (state)
           {
               case ALPHA: state = ( *input++ == 0 ) ? BETA   : GAMMA;   break;
               case BETA : state = ( *input++ == 0 ) ? GAMMA  : ALPHA;   break;
               case GAMMA: state = ( *input++ == 0 ) ? GAMMA  : OMEGA;   break;
           }
}
```

Floating-point compiler output is:

```
_fsm:
*
* AR4      is allocated to user var 'input'
*
      LDI      *AR4++,R0  ; initial state == ALPHA.
      BZ       L4         ;    if input == 0     goto state BETA
      B        L12        ;    else          goto state GAMMA
L9:   LDI      *AR4++,R0  ; state == ALPHA.
      BNZ      L12        ;    if input != 0     goto state GAMMA
L4:   LDI      *AR4++,R0  ; state == BETA.
      BNZ      L9         ;    if input != 0     goto state ALPHA
      LDI      *AR4++,R0  ; state == GAMMA.
      BNZ      EPI0_1     ;    if input != 0     goto state OMEGA
L12:  LDI      *AR4++,R0  ; state == GAMMA.
      BZ       L12        ;    if input == 0     goto state GAMMA
EPI0_1:...                ; state == OMEGA.
     ...
```

The switch statement and the state variable from this simple finite-state machine process are optimized completely away, leaving a streamlined series of conditional branches.

❑ Branch optimizations/control-flow simplification

The compiler analyzes the branching behavior of a program and rearranges the linear sequences of operations (basic blocks) to remove branches or redundant conditions. Unreachable code is deleted, branches to branches are bypassed, and conditional branches over unconditional branches are simplified to a single conditional branch. When the value of a condition can be determined at compile time (through copy propagation or other data flow analysis), a conditional branch can be deleted. Switch case lists are analyzed in the same way as conditional branches and are sometimes eliminated entirely. Some simple control-flow constructs can be reduced to conditional instructions, totally eliminating the need for branches. See Example 13–3 for floating-point compilers.

❑ Loop induction variable optimizations, strength reduction

Loop induction variables are variables whose value within a loop is directly related to the number of executions of the loop. Array indices and control variables of *FOR* loops are very often induction variables. Strength reduction is the process of replacing costly expressions involving induction variables with more efficient expressions. For example, code that indexes into a sequence of array elements is replaced with code that increments a pointer through the array. Loops controlled by incrementing a counter are implemented with block repeat instructions, or with efficient decrement-and-branch instructions. Induction variable analysis and strength reduction together often remove all references to the programmer's loop control variable, allowing it to be eliminated entirely.

❑ Loop rotation

The compiler evaluates loop conditionals at the bottom of loops, saving a costly extra branch out of the loop. In many cases, the initial entry conditional check and the branch are optimized out.

❑ Loop-invariant code motion

This optimization identifies expressions within loops that always compute the same value. The computation is moved in front of the loop, and each occurrence of the expression in the loop is replaced by a reference to the precomputed value.

❑ Inline expansion of function calls

The special keyword *inline* directs the compiler to replace calls to a function with inline code, saving the overhead associated with a function call as well as providing increased opportunities to apply other optimizations. See Example 13–4 and Example 13–5.

Example 13–4. Inline Function Expansion for Fixed-Point Compilers

```
inline int acc(int *p, int n)
{
     int i;
     int sum = 0;

     for (i = 0; i<n; i++)
      sum += p[i];
     return sum;
}
process(int *p)
{
     int sum;

     sum = acc(p, 100)
     ...
}
```

Fixed-point compiler ('C5x option) output is:

```
_process:

LF2  .set    0
     ....
                         ;ARP = AR6
     LARK    AR6,1       ;AR6 = &acc_1_sum
     MAR     *0+
     LACK    0           ;ACC = 0
     SACL    *,AR2       ;acc_1_sum = 0
     LARK    AR2, -3+LF2 ;AR2 = &p
     MAR     *0+
     LAR     AR5,*,AR5   ;AR5 = p
     LACK    99
     SAMM    BRCR        ;loop 100 times
     RPTB    L9–1        ;begin loop
     LAC     *+,AR6      ;ACC = *p+
     ADD     *           ;ACC += acc_1_sum
     SACL    *,AR5       :acc_1_sum = ACC
L9:                      ;end loop
     ....
```

The keyword inline signals the compiler to expand the call to acc in place. The symbol acc_1_sum is created to accumulate the sum.

Example 13–5. Inline Function Expansion for Floating-Point Compilers

```
inline blkcpy (char *to, char *from, int n)
{   if (n > 0 )
        do *to++ = *from++; while (--n !=0);
}
struct s { int a,b,c[10]; } s;
initstr (struct s *ps, char t[12])
{   blkcpy((char *)ps, t, 12);
}
```

Floating-point compiler output is***:***

```
_initstr
* R2   assigned to variable 't'
* AR2  assigned to variable 'blkcpy_1_to'
* AR4  assigned to variable 'blkcpy_1_from'
* BK   assigned to variable 'ps'
* RC assigned to variable 'L$1'

        LDI        BK,AR2        ;blkcpy_1_to = ps
        LDI        R2,AR4        ;blkcpy_1_from = t
        LDI        *AR4++,R0     ;+-------------------------------------
        RPTS       10            ;| expansion of blkcpy:
        STI        R0,*AR2++     ;| copy 12 words
    ||  LDI        *AR4++,R0     ;+-------------------------------------
        STI        R0,*AR2++     ;
    ...
```

The special inline declaration of blkcpy results in the call being replaced with the function's body. The compiler creates temporary variables blkcpy_1_to and blkcpy_1_from, corresponding to the parameters of blkcpy. Often, copy propagation can eliminate assignments to such variables when the argument expressions are not reused after the call.

13.1.7.2 Fixed-Point Compiler Optimizations

The following optimizations are specific to the 'C2xx/'C5x compiler.

❑ Register variables

The compiler maximizes the use of the address registers of the 'C2xx/'C5x DSPs by using them as pointers. This optimization is particularly effective for pointers that arise when array index constructs are turned into loop induction variables.

❑ Cost-based register allocation

The compiler allocates registers to user variables and temporary values according to their type, use, and frequency. Variables used within loops are weighted to have priority over others, and those variables whose uses do not overlap can be allocated to the same register.

❑ Delayed branches, calls, and returns

The ’C5x supports a number of delayed branch, call, and return instructions. Three of these are used by the compiler: branch unconditional (BD), call to a named function (CALLD), and simple return (RETD). These instructions execute in two fewer cycles than their nondelayed counterparts. They execute two instruction words after they enter the instruction stream. Sometimes it is necessary to insert an NOP after a delayed instruction to ensure proper operation. This involves one more word of code than a nondelayed sequence, but it is still one cycle faster. Note that the compiler emits a comment in the instruction sequence where the delayed instruction executes. See Example 13–6.

❑ Arranging variables on the local frame

Local variables are accessed by adjusting AR2 to point to the variable on the frame and then by accessing AR2 indirectly. If variables that are allocated far apart on the local frame must be accessed sequentially, an ADRK or SBRK instruction is required to adjust AR2. If variables that are allocated next to each other must be accessed sequentially, the ADRK or SBRK instruction is not required, because AR2 can be adjusted to point to the next variable by adding a + or a – symbol to the previous indirect access. The compiler takes advantage of this situation by recognizing local variables that are accessed sequentially and allocating those variables next to each other. See Example 13–7.

❑ Autoincrement addressing modes

For pointer expressions of the form **p++*, the compiler uses efficient ’C2xx/’C5x autoincrement addressing modes. In many cases, where code steps through an array in a loop, such as *for (i = 0; i < N; ++i) a[i]...,* the loop optimizations convert the array references to indirect references through autoincremented register variable pointers.

❑ Repeat blocks

For the ’C5x, the compiler supports zero-overhead loops with the RPTB instruction. The compiler can detect loops controlled by counters and generate them via the efficient repeat forms. The iteration count can be either a constant or an expression.

Example 13–6. Fixed-Point Compiler Delayed Branch, Call, and Return Instructions

```
driver()
{
     int i0, i1;
     while (input(&i0) && input(&i1))
          process(i0, i1);
}
```

Fixed-point compiler (’C5x option) output is:

```
_driver:
     POPD   *+
     SAR    AR0,*+            ;save AR0 and return address
     SAR    AR1,*             ;set up local frame
     LARK   AR0,3
     LAR    AR0,*0+
;    <save register vars>
     SAR    AR6,*+            ;save AR6
     SAR    AR7,*+,AR6        ;save AR7
     LARK   AR6,1             ;AR6 = &i0
     LARK   AR7,2             ;AR7 = &i1
     BD     L2                ;begin branch to loop control
     MAR    *0+,AR7
     MAR    *0+

***  B      L2                ;branch to loop control occurs
L1:
     LAC    * ,AR1            ;stack *AR7
     SACL   *+,AR6
     CALLD  _process          ;call occurs
     LAC    * ,AR1
     SACL   *+
***  CALL   _process  OCCURS  ;begin call
     SBRK   2
L2:                           ;loop control
     CALLD  _input            ;begin call
     MAR    *,AR1
     SAR    AR6,*+            ;stack AR6 (&i0)
***  CALL   _input  OCCURS    ;call occurs
     MAR    *-                ;clear stack
     BZ     EPI0_1            ;quit if _input returns 0
     CALLD  _input            ;begin call
     SAR    AR7,*+            ;stack AR7 (&il)
     NOP                      ;necessary, no branches in delay slot
     CALL   _input            ;call occurs
     MAR    *-,AR7            ;clear stack
     BNZ    L1                ;continue if _input returns !0
EPI0_1:
     MAR    *,AR1             ;function epilog
;    <restore register vars>
     MAR    *-
     LAR    AR7,*-            ;restore AR7
     LAR    AR6,*             ;restore AR6
     SBRK   4                 ;clear local frame
     PSHD   *-                ;push return address on hardware stack
     RETD                     ;begin return
     LAR    AR0,*             ;restore AR0
     NOP                      ;necessary, no PSHD in delay slot
***  RET    OCCURS            ;return occurs
```

Example 13–7. Arranging Variables on the Local Frame

```
func()
{
     int a,b,i,j;
     ...
     call(i+a,j+b);
}
```

Fixed-point compiler (’C5x option) output is*:*

```
_func:
     ...
                           ;ARP = AR2,  AR2 points to b
     LAC     *+            ;ACC = b,    AR2 point to j
     ADD     *+,AR1        ;ACC += j,   AR2 points to a
     SACL    *+,AR2        ;stack ACC
     LAC     *+            ;ACC = a,    AR2 points to i
     CALLD   _call         :begin call
     ADD     *,AR1         ;ACC += i
     SACL    *+            ;stack ACC
***  CALL    _call  OCCURS ;call occurs
     SBRK    2             ;clear stack
```

The compiler rearranges the order of the variables on the local frame from int a,b,i,j; to int b,j,a,i; so that the expressions can be computed without unnecessary additional adjustments to AR2, the local variable pointer register.

❑ Elimination of unnecessary LDPK instructions

Whenever a global variable is accessed, the compiler must first ensure that the page pointer has the right value to allow correct access of that variable. If the page pointer does not have the right value, the value must be loaded with an LDPK instruction. To avoid emitting unnecessary LDPK instructions, the compiler performs analysis of global variables declared in a module. This analysis determines where, relative to page boundaries, the global variables are allocated. Note that this analysis does not include variables declared in a different module which are only referenced in the current module. Therefore, when handling successive accesses of global variables declared in the current module, the compiler issues an LDPK instruction only when a variable does not reside in the same page as the last global variable accessed. See Example 13–8.

Example 13–8. Elimination of Unnecessary LDPK Instructions

```
int g1, g2, g3;
extern int e1;

func()
{
     g1 = g2+g3;
     e1 = g2;
}
```

Fixed-point compiler (’C5x option) output is:

```
_func

     ...
     LDPK      _g3            ;set page pointer for this module
     LAC       _g3            ;load g3
     ADD       _g2            ;add g2
     SACL      _g1            ;store g1
     LAC       _g2            ;load g2
     LDPK      _e1            ;set page pointer for external variable
     SACL      _e1            ;store e1
```

Because g1, g2, and g3 are all declared in the local module, the compiler can determine where the variables are in relation to page boundaries and can change the page pointer accordingly — in this case, only once. Note that the page pointer is reset to access the variable that is declared in another module and only referenced in this module.

13.1.7.3 Floating-Point Compiler Optimizations

The following optimizations are specific to the floating-point compiler.

❑ Register variables

The compiler helps maximize the use of registers for storing local variables, parameters, and temporary values. Variables stored in registers can be accessed more efficiently than variables in memory. This optimization is particularly effective for pointers that arise when array index constructs are turned into loop induction variables. See Example 13–9 and Example 13–10.

Example 13–9. Register Variables and Register Tracking/Targeting

```
int gvar;
reg(int i, int j)
{
    gvar = call() & i;
    j  = gvar + i;
}
```

Floating-point compiler output is:

```
_reg:
*
*R4 is allocated to user var 'i'
*R5 is allocated to user var 'j'
*
     ...
     CALL       _call         ;R0 = call()
     AND        R4,R0         ;R0 &= i
     STI        R0,@_gvar     ;gvar = R0
     ADDI       R4,R0,R5      ;tracks gvar in R0,
     ...                      ;targets result into R5 (j)
```

The compiler allocates local variables i and j into registers R4 and R5, as indicated by the comments in the assembly listing. Allocating i to R4 and tracking gvar in R0 allows the sum gvar + i to be computed with a 3-operand instruction, targeting the result directly into j in R5.

❑ Register tracking/targeting

The compiler tracks the contents of registers so that it avoids reloading values if they are used again soon. Variables, constants, and structure references such as (*a.b*) are tracked through both straight-line code and forward branches. The compiler also uses register targeting to compute expressions directly into specific registers when required, as in the case of assigning to register variables or returning values from functions. See Example 13–9.

Example 13–10. Repeat Blocks, Autoincrement Addressing Modes, Parallel Instructions, Strength Reduction, Induction Variable Elimination, Register Variables, and Loop Test Replacement for Floating-Point Compilers

```
float a[10], b[10];
scale(float k)
{
   int i;
   for (i = 0; i < 10; ++i)
      a[i] = b[i] * k;
   ...
```

Floating-point compiler output is:

```
_scale:
   ...
      LDI       @CONST+0,AR4      ; AR4 = &a[0]
      LDI       @CONST+1,AR5      ; AR5 = &b[0]
      MPYF      R4,*AR5++,R0      ; compute first product
      RPTS      8                 ; loop for next 9
      STF       R0,*AR4++         ; store this product...
   || MPYF      R4,*AR5++,R0      ; ...and compute next
      STF       R0,*AR4++         ; store last product
      ...
```

This process shows general and floating-point-specific optimizations working together to generate highly efficient code. Induction variable elimination and loop test replacement allow the compiler to recognize the loop as a simple counting loop and then generate a repeat block. Strength reduction turns the array's references into efficient pointer autoincrements. The compiler unrolls the loop once to separate the first multiply and last store, allowing the body of the loop to be written as a single parallel instruction.

□ Cost-based register allocation

The compiler, when enabled, allocates registers to user variables and compiles temporary values according to their type, use, and frequency. Variables used within loops are weighted to have priority over others, and those variables whose uses do not overlap may be allocated to the same register. Variables with specific requirements are allocated into registers that can accommodate them.

□ Autoincrement addressing modes

For pointer expressions of the form *p++, *p– –, *++p, or *– –p, the compiler uses efficient TMS320 autoincrement addressing modes. In many cases, where code steps through an array in a loop, such as for (i = 0; i < N; ++i) a [i]..., the loop optimizations convert the array's references to indirect references through autoincremented register variable pointers. See Example 13–10.

❑ Repeat blocks

The floating-point compiler supports zero-overhead loops with the RPTS (repeat single) and RPTB (repeat block) instructions. The compiler can detect loops controlled by counters and generate them by using the efficient repeat forms: RPTS for single-instruction loops or RPTB for larger loops. For both forms, the iteration count can be either a constant or an expression. See Example 13–5 and Example 13–10.

Induction variable elimination and loop test replacement allow the compiler to recognize the loop as a simple counting loop and then generate a repeat block. Strength reduction turns the array references into efficient pointer autoincrements.

❑ Delayed branch instructions

The floating-point compiler supports delayed branch instructions that can be inserted three instructions early in an instruction stream, avoiding costly pipeline flushes associated with normal branches. The compiler uses unconditional delayed branches wherever possible and conditional delayed branches for counting loops. See Example 13–11.

Example 13–11. Floating-Point Compiler Delayed Branch Optimizations

```
wait(volatile int *p)
{
       for(;;)
        if (*p & 0x80) *p |= 0xF0;
}
```

Floating-point compiler output is*:*

```
_wait:
L6:
       LDI  *AR4,R0      ; R0 = *p (AR4 is allocated to p)
       TSTB 128,R0       ; test *p & 0x80
       BZ   L6           ; false: loop back
       BD   L6           ; true: loop back (delayed)
       LDI  *AR4,R0      ; R0 = *p
       OR   0f0h,R0      ; R0 = *p | 0xF0
       STI  R0,*AR4      ; *p = R0
***    B    L6           ; branch occurs
```

The unconditional branch at the bottom of this loop is written as a delayed branch, allowing it to execute in one machine cycle.

❑ Use of registers for passing function arguments

The compiler supports a new, optional calling sequence that passes arguments to registers rather than pushing them onto the stack. This can result in significant improvement in performance, especially if calls are important in the application.

❑ Parallel instructions

Several floating-point instructions such as load/load, store/operate, and multiply/add can be paired and executed in parallel. When adjacent instructions match the addressing requirements, the compiler combines them in parallel. Although the code generator performs this optimization, the optimizer greatly increases effectiveness because operands are more likely to be in registers. See Example 13–5 and Example 13–10.

❑ Conditional instructions

The load instructions in the floating-point C compiler can be executed conditionally. For simple assignments such as *a = condition ? expr1 : expr2* or *if* (*condition*) *a = b*, the compiler can use conditional loads to avoid costly branches.

❑ Loop unrolling

When the compiler determines that a short loop is executed a low, constant number of times, it replicates the body of the loop rather than generating the loop; note that low and short are subjective judgments made by the compiler. This avoids any branches or use of the repeat registers. See Example 13–12.

Example 13–12. Loop Unrolling

```
add3(int a[3])
{
   int i, sum = 0;
   for (i = 0; i < 3; ++i) sum += a[i]

   return sum;
}
```

Floating-point compiler output is:

```
_add3:
      ...
      LDI       *-FP(2),AR4   ; AR4 = &a[0]
      LDI       *AR4++,RC     ; sum += a[0]
      ADDI      *AR4++,RC     ; sum += a[1]
      ADDI      *AR4++,RC     ; sum += a[2]
      LDI       RC,R0         ; return sum
      ...
```

The compiler determines that this loop is short enough to unroll, resulting in a simple three-instruction sequence and no branches.

13.1.7.4 ’C6x Compiler Optimizations

The following optimizations are specific to the ’C6x compiler:

- ❑ Software pipelining
- ❑ If conversion/predicated execution
- ❑ Memory address cloning
- ❑ Memory address dependence elimination

The ’C6x compiler also includes an **assembly optimizer**. For high-performance operations, the assembly optimizer extracts the maximum throughput possible from the application code. You can use the assembly optimizer by running the **dynamic profiler** to identify the critical code segments to optimize. You can find further details on the dynamic profiler in Chapter 14, *System Integration and Debugging Tools*.

The assembly optimizer lets you write simple, conventional-looking ’C6x assembly language programs. However, you must be aware that this assembly code ignores:

- ■ parallel instructions
- ■ instruction latencies
- ■ register usage

When the assembly code is input into the assembly optimizer, this tool then schedules the instructions heeding the architectural parallelism. It also considers ’C6x latency requirements, maximizes parallel code, and allocates registers for the unlimited number of available named, virtual registers in order to eliminate manually performing this task.

13.1.7.5 ’C80 Multiprocessing Compiler Optimizations

The following optimizations are specific to the multiprocessing compilers:

- ❑ Register allocator streamlines code development by efficiently managing variable usage of registers

- ❑ Code compactor examines serially-written code to identify instructions that can run in parallel

- ❑ Multitasking executive simplifies the synchronization and execution of multiple DSP tasks running concurrently on the ’C80. Supports ’C8x-to-host processor communications and provides local control of on-chip parallel-processing tasks.

- ❑ Allocates variables to registers

- ❑ Performs control-flow-graph simplification, loop rotation, loop unrolling

- ❑ Simplifies expressions and statements, and functions with return values that are never used

- Expands calls to functions declared inline
- Performs local copy/constant propagation and loop optimizations
- Eliminates local common expressions, global common subexpressions, and global redundant assignments
- Converts array references in loops to incremented pointer form
- Reorders function definitions so attributes of called functions are known when the caller is optimized
- Propagates arguments into function bodies when all call sites pass the same value in the same argument position

13.2 TMS320 Macro Assembler, Linker, and Archiver

The TMS320 macro assemblers and linkers are currently available for all TMS320 devices. The fixed-point assembler and linker support the C2xx/C5x devices; the floating-point assembler and linker support the ’C3x/C4x devices. Assemblers and linkers are also available for both the ’C54x, the ’C6x, and the ’C8x. The assembler/linker is shipped with the C compiler package. For the ’C2xx/’C5x, ’C54x, and ’C3x/C4x, a PC version without the C compiler is available. This section discusses the two assembler and linker toolsets (floating point and fixed point) and highlights their differences.

The TMS320 macro assembler and linker are code-generation tools that convert TMS320 assembly language source files into executable object code. Key features include:

- Macro capabilities and library functions
- Conditional assembly
- Relocatable modules
- Complete error diagnostics
- Symbol table and cross-references

The **assembler** translates assembly language source files into machine language object files. Source files can contain instructions, assembler directives, and macro directives. Assembler directives control various aspects of the assembly process such as the source-listing format, symbol definition, and the way the source code is placed into sections. The assembler has the following features:

- Processes the source statements in a text file to produce a relocatable object file

- Produces a source listing (if requested) and provides control over this listing

- Appends a cross-reference listing to the source listing (if requested)

- Allows segmentation of your code

- Maintains a section program counter (SPC) for each section of object code

- Defines and references global symbols

- Assembles conditional blocks

- Supports macros, allowing the user to define macros either inline with or within a macro library

- ❑ For the TMS320C8x assemblers:
  - ■ Separate assemblers support the MP and PP assembly languages and a linker links assembled code in common object file format (COFF) into common memory.
  - ■ Directives map program and data code on specific processors for fast integration and debugging of parallel-processing code.
  - ■ Support is provided to relocatable modules for maximum code flexibility.

The **linker** combines object files into a single executable object module. As it creates the executable module, it performs relocation operations and resolves external references. The linker accepts COFF object files (created by the assembler) as its input. It can also accept archive library members and modules created by a previous linker run. Linker directives allow you to combine object file sections, bind sections and symbols to specific addresses, and define/redefine global symbols. The linker has these features:

- ❑ Defines a memory model that conforms to the target system's memory
- ❑ Combines object file sections
- ❑ Allocates sections into specific areas within the target system's memory
- ❑ Defines or redefines global symbols to assign them specific values
- ❑ Relocates sections to assign them to final addresses
- ❑ Resolves undefined external references between the input files

The **archiver** makes it possible to collect a group of files into a single archive file. For example, several macros can be collected together into a macro library. The assembler searches through the library and uses the members that are called as macros by the source file. Also, it is possible to use the archiver to collect a group of object files into an object library. The linker includes the members in the library that resolve external references during the link.

Most EPROM programmers do not accept COFF object files as their input. The assembler/linker includes a utility to convert the COFF file into Intel™, Tektronix™, TI-tagged, Motorola-S™, or ASCII hex-object formats.

Chapter 14

System Integration and Debugging Tools

The system integration and debugging tools for the TMS320 family of digital signal processors include the TMS320 debugger's interface (C/assembly source debugger), software simulators, standard evaluation modules (EVMs), and emulators (XDS510™/XDS510WS). These tools are described in the following sections.

Topic **Page**

14.1 TMS320 Debugger’s Interface (C/Assembly Source Debugger)

The TMS320 debugger’s interface brings new levels of power and flexibility to embedded systems development. The C/assembly source debugger is the standard interface for the simulators, evaluation modules, and emulators. The debugger interface is now available for all ’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x,and ’C8x tools.

The debugger is an advanced software interface that runs on a PC or SPARC ™ and works with TI debugging tools such as the unique, scan-based, real-time TMS320 emulators, the XDS510 and XDS510WS. The debugger provides complete control over programs written in C or assembly language.

The debugger improves productivity by enabling you to debug a program in the language in which it is written. Programs can be debugged in C, assembly language, or both. The debugger also has profiling capabilities that show where to focus development time by quickly identifying the hot or time-consuming sections of a program.

The debugger is easy to learn and use. Its menu-driven window and mouse-oriented interface reduces learning time and eliminates the need to memorize complex commands. The debugger’s custom-made displays and flexible command entry let you develop a debugging environment that suits the systems needs (see Figure 14–1). A shortened learning curve and increased productivity reduce the software development cycle, speeding products to market.

Figure 14–1. Debugger's Customized Display

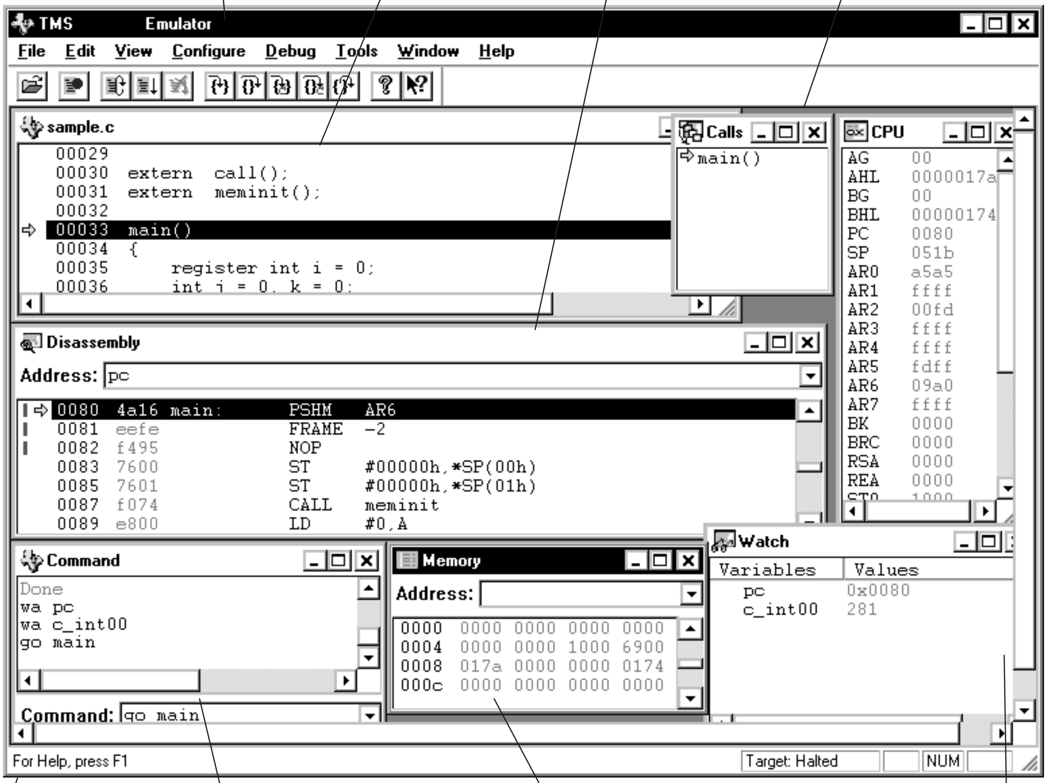
Pulldown menus are available for most tasks

Toolbar icons let you select commands quickly

C source display

Disassembly display

Natural-format data displays



```
00029
00030  extern  call();
00031  extern  meminit();
00032
00033  main()
00034  {
00035      register int i = 0;
00036      int j = 0, k = 0;
```

Calls: main()

Disassembly Address: pc

```
0080  4a16  main:    PSHM   AR6
0081  eefe            FRAME  -2
0082  f495            NOP
0083  7600            ST     #00000h,*SP(00h)
0085  7601            ST     #00000h,*SP(01h)
0087  f074            CALL   meminit
0089  e800            LD     #0,A
```

```
AG    00
AHL   0000017a
BG    00
BHL   00000174
PC    0080
SP    051b
AR0   a5a5
AR1   ffff
AR2   00fd
AR3   ffff
AR4   ffff
AR5   fdff
AR6   09a0
AR7   ffff
BK    0000
BRC   0000
RSA   0000
REA   0000
ST0   1000
```

```
Done
wa pc
wa c_int00
go main
```

Command: go main

```
0000  0000 0000 0000 0000
0004  0000 0000 1000 6900
0008  017a 0000 0000 0174
000c  0000 0000 0000 0000
```

Variables	Values
pc	0x0080
c_int00	281

Context-sensitive status bar

Interactive command entry with command history

Scrolling data displays with on-screen editing

Function call traceback

14.1.1 Debugger Features

Conditional execution and single-stepping (including single-stepping into and over function calls) give you complete control over program execution. A breakpoint can be set or cleared with a click of the mouse or by typing commands. A memory map identifies the portions of target memory that the debugger can access and that can be defined. You can load only the symbol table's portion of an object file to work with systems that have code in ROM. The debugger can execute commands from a batch file, providing an easy method for entering often-used command sequences. Key features include:

- **Multioperation support**. For the 'C2xx, 'C4x, 'C5x, 'C54x, 'C6x, and 'C8x, the C/assembly debugger is enhanced with special parallel-processing capabilities (multiple-processor debug/breakpoint and single-step).
- **Multilevel debugging**. The debugger allows you to debug both C and assembly language codes. While debugging a C program, you can choose to view the C source, the disassembly of the object code created from the C source, or both.
- **Fully configurable, window-oriented interface**. The debugger separates code, data, and commands into manageable information. You can select from several displays or, since the debugger's display is completely configurable, you can create the interface that best suits the application. You can change the display colors, the physical appearance of displayed features (such as window borders), and the window size and position.
- **Flexible command entry**. Commands can be entered by using a mouse, the function keys, or the pull-down menus. The debugger's command history can be used to reenter commands. Symbolic debug is supported, so structure and variable names can be used instead of address or data locations.
- **On-screen editing**. Any data value displayed in any window can be changed easily by pointing (with the mouse) at the value, clicking, and entering the correct value.
- **Continuous update**. The debugger continuously updates information on the screen, highlighting changed values.

- **Comprehensive data display**. You can easily create windows for watching, displaying, and editing the values of variables, arrays, structures, pointers — any kind of data — in their natural format (floating point, integer, character, enumerated, or pointer). Entire linked lists can be displayed.

- **Powerful command set**. The TMS320 debugger supports a powerful command set that makes full use of C expressions. One debugger command performs actions that might require several commands in another system.

- **Compatibility**. The C source debugger runs on IBM PC-ATs and compatible PCs and SPARC workstations.

- **Simplicity**. The debugger's simple setup, default configurations, predefined commands, and inherent flexibility facilitate sophisticated debugging within a short time.

- **Calls stack window.** The calls stack window displays function names in the order that they are called and put on the stack. A function name is removed when popped from the calls stack. This allows you to debug a program that is not executing properly because of a lack of stack space.

- **Memory window.** Memory contents can be displayed and edited to allow you to observe the movement of data and compare expected values to actual ones.

14.1.2 Code Profiler

The code profiling functionality increases the debugging flexibility of the Texas Instruments C source debugger. By using the familiar debugger interface, the profiler shows you where to focus your development time by quickly identifying the time-consuming sections of the program. Code optimizations, such as eliminating bottleneck problems, can dramatically impact execution time. A powerful set of profiling commands simplifies the process of maximizing code efficiency.

Key features of the code profiler include:

- **User-friendly interface**. The TI code profiler shares the same fully configurable, window-oriented, and mouse-controlled interface as the TI C source debugger, so learning to profile is quick and easy.

- **Multilevel profiling**. An assembly window and a C window are displayed, so you can profile C code, assembly code, or both.

- **Powerful command set**. A rich set of commands is available to select and manipulate profile areas on the global, module, function, and explicit levels, so you can efficiently profile even the most complex applications.

- ❑ **Comprehensive statistics.** The profiler provides all the information you need to identify bottlenecks in your code:
  - The number of times each area was entered during the profile session
  - The total execution time of an area, including or excluding the execution time of any subroutines called from within that area
  - The maximum time for one iteration of an area, including or excluding the execution time of any subroutines called from within that area
- ❑ **Versatile display.** The profiler allows you to choose profile areas, the type of statistical data, and the sorting criteria, which ensures an efficient, customized display of statistics. The data also can be accompanied by histograms to show the statistical relationship between profile areas.
- ❑ **Disabled areas.** You can disable portions of a profile area to prevent them from adding to the statistics. This is convenient for removing the timing impact of standard library functions or a fully optimized portion of code.

14.1.3 Dynamic Profiler—'C6x Only

The profiling tool for the 'C6x devices is called the *dynamic profiler.* The profiler creates cycle histograms that are continuously updated as the code runs. It graphically illustrates functions, ranges, and lines in an application that cause performance degradations.

Among the many statistics the profiler can show is:

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

Profiling capability is available in the 'C2xx, 'C3x, 'C4x, 'C5x, 'C54x, 'C6x, and 'C8x C source debuggers and simulators. Profiling is supported in versions that run on Windows™, OS/2™, and SunOS™ operating systems.

14.2 TMS320 Software Simulators

A TMS320 simulator is a software program that simulates the TMS320 microprocessor and microcomputer modes for cost-effective TMS320 software development and program verification in other than real time. With the inexpensive software simulator, you can debug without target hardware. Files can be associated with I/O ports so that specific I/O values can be used during test and debug. Time-critical code, as well as individual portions of the program, can be tested. The clock's counter allows loop timing during code optimization. Breakpoints can be established according to read/write executions (using either program or data memory) or instruction acquisitions.

Each of the TMS320 simulator's software programs simulates TMS320 operation and allows monitoring of the state of the TMS320. Simulation speed is typically on the order of thousands of instructions per second (SPARC) or hundreds of instructions per second (IBM PC). TMS320 simulators are available for the 'C2xx, 'C3x, 'C4x, 'C5x, 'C54x, 'C6x, and 'C8x devices. The 'C8x simulator is shipped with the 'C8x software tool kit for the Sun development platform.

14.2.1 Simulator Key Features

Key features common to the TMS320 software simulators include:

- Execution of user-oriented DSP programs on a host computer
- Modification and inspection of registers
- Data and program memory modification and display:
  - Modification of an entire block at any time
  - Initialization of memory before a program is loaded
- Simulation of peripherals, caches, and pipelined timings
- Extraction of instruction cycle timing for device performance analysis
- Programmable breakpoints on:
  - Instruction acquisition
  - Memory reads and writes (data or program)
  - Data patterns on the data bus or the program bus
  - Error conditions
- Trace on:
  - Accumulator
  - Program counter
  - Auxiliary registers

- Single-stepping of instructions
- Interrupt generation at user-specified intervals
- Error messages for:
  - Illegal opcodes
  - Invalid data entries
- Execution of commands from a journal file
- Use of save states for restarting simulation (’C25)
- Simulation of the entire instruction set for the appropriate device
- Simulation of the peripheral’s key features for the appropriate device (’C2xx, ’C3x, ’C4x)
- Command entry from either menu-driven keystrokes (menu mode) or line mode (’C3x, ’C4x)
- Simulation parameters quickly stored/retrieved from files to facilitate preparation for individual sessions
- Reverse assembly for editing and reassembling source statements
- Memory that can be displayed (at the same time) as
  - Hexadecimal 16-bit values
  - Assembled source code
- Execution modes
  - Single/multiple instruction count
  - Single/multiple cycle count
  - Until condition is met
  - While condition exists
  - For set loop count (’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x)
  - Unrestricted run with halt by keyed input
- Trace execution with display choices
  - Designated expression values
  - Cache memory (’C3x, ’C4x)
  - Instruction pipeline for easy optimization of code (’C2xx, ’C3x, ’C4x, ’C5x, ’C54x)
- Cycle counting
  - Display of the number of clock cycles in a single-step operation or in the run mode
  - Externally generated mode that can be configured with wait states for accurate cycle counting (’C2xx, ’C3x, ’C4x)

The simulators use TMS320 object code produced by the TMS320 macro assembler/linker or ANSI C compiler. Input and output files can be associated with the port addresses of the I/O instructions to simulate I/O devices connected to the processor. Each interrupt flag can be set periodically at a user-defined interval for simulating an interrupt signal. Before program execution is initiated, breakpoints can be defined (a branch to *self* is detected), and the trace mode set (execution is halted).

Once program execution is suspended, the internal registers and both program and data memories can be inspected and/or modified. The trace memory can also be displayed. A record of the simulation session can be maintained in a journal file so it can be reexecuted to regain the same machine state during another simulation session.

14.2.2 TMS320C2xx Simulator

The ’C2xx software simulator uses the TI high-level language (HLL) debugging interface. In addition to the features listed in Section 14.1, it also simulates the serial port and timer peripherals.

The simulator lets you verify and monitor the state of the processor without having to install additional hardware. The ’C2xx software simulator performs:

❑ Modeling of the ’C2xx addressable memory
❑ Modeling of the ’C2xx additional features
❑ Operation of the debugging interface

14.2.3 TMS320C3x Simulator

The ’C3x simulator’s software program simulates the operation of the ’C3x generation of 32-bit, floating-point digital signal processors and aids in the development of effective software. The simulator uses the standard C/assembly source debugger interface, allowing the user to debug code in C, assembly, or both. The ’C3x simulator also simulates external interrupts to the ’C3x device and cache utilization.

The simulator lets you verify and monitor the state of the processor without having to install additional hardware. The ’C3x software simulator performs:

❑ Modeling of the ’C3x addressable memory
❑ Modeling of the ’C3x additional features
❑ Operation of the debugger’s interface

14.2.4 TMS320C4x Simulator

The ’C4x software simulator uses the TI debugger’s interface. This flexible debugging interface lets you view both C language and assembly language simultaneously and can execute single-stepping and software breakpoints on either language for HLL debug (see Section 14.1). The ’C4x simulator also simulates cache utilization.

The simulator lets you verify and monitor the state of the processor without additional hardware. The ’C4x software simulator performs:

❑ Modeling of the ’C4x addressable memory
❑ Modeling of the ’C4x additional features
❑ Operation of the debugger’s interface

14.2.5 TMS320C5x Simulator

The ’C5x software simulator uses the TI HLL debugging interface. In addition to the features listed in Section 14.1, it also simulates the serial port and timer peripherals.

The simulator lets you verify and monitor the state of the processor without having to install additional hardware. The ’C5x software simulator performs:

❑ Modeling of the ’C5x addressable memory
❑ Modeling of the ’C5x additional features
❑ Operation of the debugging interface

14.2.6 TMS320C54x Simulator

The ’C54x software simulator also uses the TI HLL debugging interface and, in addition to the features listed in Section 14.1, simulates the serial port and timer peripherals.

The simulator lets you verify and monitor the state of the processor without having to install additional hardware. The ’C54x software simulator performs:

❑ Modeling of the ’C54x addressable memory
❑ Modeling of the ’C54x additional features
❑ Operation of the debugging interface

14.2.7 TMS320C6x Simulator

The ’C6x simulator software simulates the host processor and memory to perform instruction-and C-level simulation of the TMS320C6x DSP. It uses debugger code produced by the macro assembler/linker or ANSI C compiler, and the standard ’C6x debugger interface. The simulator provides software debug capability for a ’C6x , plus external memory without the DSP hardware. In addition to the simulator features common to all TMS320 software simulators. the ’C6x simulator features include:

- ❑ Execution of user-oriented DSP programs on a host computer
- ❑ Modification and inspection of registers
- ❑ Data and program memory modification and display
  - ■ Modification of an entire block at any time
  - ■ Initialization of memory before a program is loaded
- ❑ Simulation of caches and pipelined timings
- ❑ Extraction of instruction cycle timing for device performance analysis
- ❑ Programmable breakpoints on:
  - ■ Instruction acquisition
  - ■ Error conditions
- ❑ Single stepping of instructions

The simulator offers the following additional features to the TMS320 debugger interface:

- ❑ Can connect memory mapped I/O to a host file to simulate I/O such as synchronous serial port I/O
- ❑ Can simulate external interrupts

14.2.8 TMS320C8x Simulator

The ’C8x simulator software uses the standard TMS320 debugger interface to simulate the operation of the ’C8x master and parallel processors on the host processor rather than targeting an actual ’C8x target system. The simulator can be used to benchmark and to evaluate the ’C8x or to assist in code development when a target system is not yet available. This Sun-based software simulator provides cycle-by-cycle simulation of the ’C80 or ’C82.

Special display features include:

- ❑ An image display that shows a grayscale or color representation of an image stored in memory
- ❑ A data-flow plot that provides cycle-by-cycle visual representation or processor status and memory accesses.

14.3 TMS320 System Debugging and Evaluation Tools

The TMS320 family includes a full range of system debugging tools that can be used as sample target systems in evaluation and application development. Included in this broad line of tools are:

- ❑ DSK—DSP Starter Kits
- ❑ EVM—evaluation modules
- ❑ XDS™—eXtended Development Support emulators

14.3.1 DSP Starter Kit (DSK)

The TI TMS320 DSP Starter Kit (DSK) (see Figure 14–2) is an ideal low-cost tool for first-time users interested in evaluating a DSP platform. Available for the ’C3x, ’C5x, ’C54x, and ’C6x DSPs, the DSK allows you to experiment with and use a DSP for real-time signal processing with the benefits of hardware execution and software support. With the analog-ready interface, you can easily benchmark and test applications such as control systems, audio, and speech processing. The DSK allows you to write and run real-time source code, evaluate that code, and debug your system.

Each DSK comes complete with a TMS320-based board, its own easy-to-use assembler/linker and debugger, and a documentation package. The DSK board contains:

- ❑ An RS-232 serial port (’C5x) or parallel printer port (’C3x) for communicating with your PC

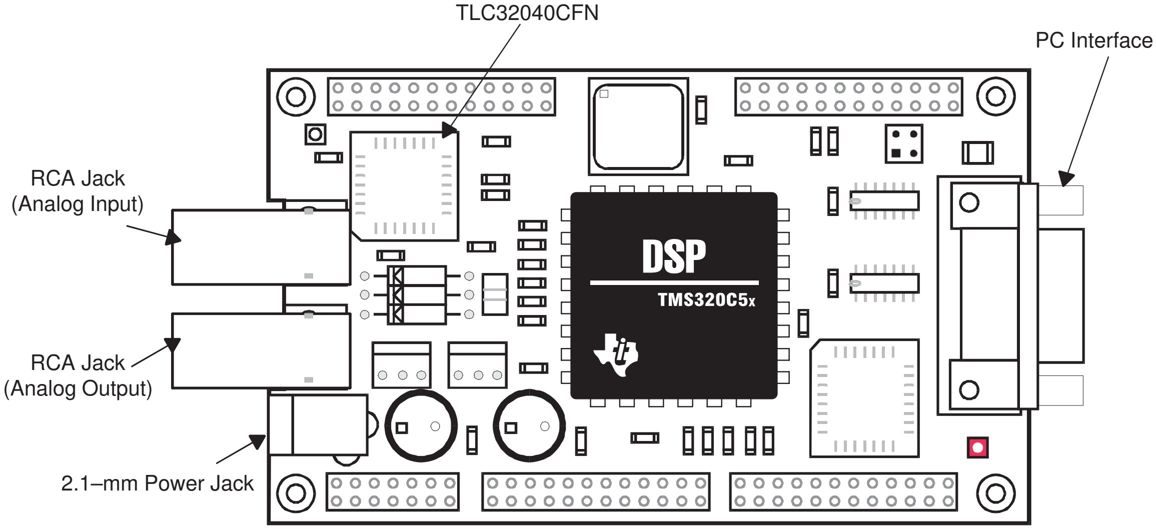
- ❑ A 2.1-mm jack that allows you to attach a simple wall-mounted ac or dc transformer as your power supply (The ’C5x supports ac only.)

- ❑ Two standard RCA jacks (for analog I/O) that provide direct connections to a microphone, speaker, or other analog device

- ❑ An on-board EPROM that allows the DSK to communicate with your PC (’C5x only)

- ❑ 10K of on-chip RAM words on the 40-MHz ’C5x board, or 2K words of on-chip RAM on the 50-MHz ’C3x board

Figure 14–2. DSP Starter Kit (DSK)



The DSK assembler key features include:

- ❑ A simple and easy-to-use interface into which only the most significant features of an assembler have been incorporated
- ❑ Special directives to assemble code at an absolute address during the assembly phase. The linking function is also handled by the assembler. As a result, programs are created quickly and easily.

The DSK debugger key features include:

- ❑ An easy-to-use, window-oriented interface that enables you to download, execute, and debug assembly code
- ❑ A small command set that reduces the number of instructions the user must learn

The TMS320 DSKs run on a PC-AT with MS-DOS or PC-DOS (version 4.01 and later).

14.3.2 Evaluation Modules

TMS320 evaluation modules (EVMs) are low-cost development boards used for device evaluation, benchmarking, and limited system debug. EVMs are PC add-in boards that include the target processor, a small amount of memory, and limited peripherals that allow you to run code in real time and interface to an external system. EVMs are available for the ’C30, ’C50, ’C54x, and ’C6x.

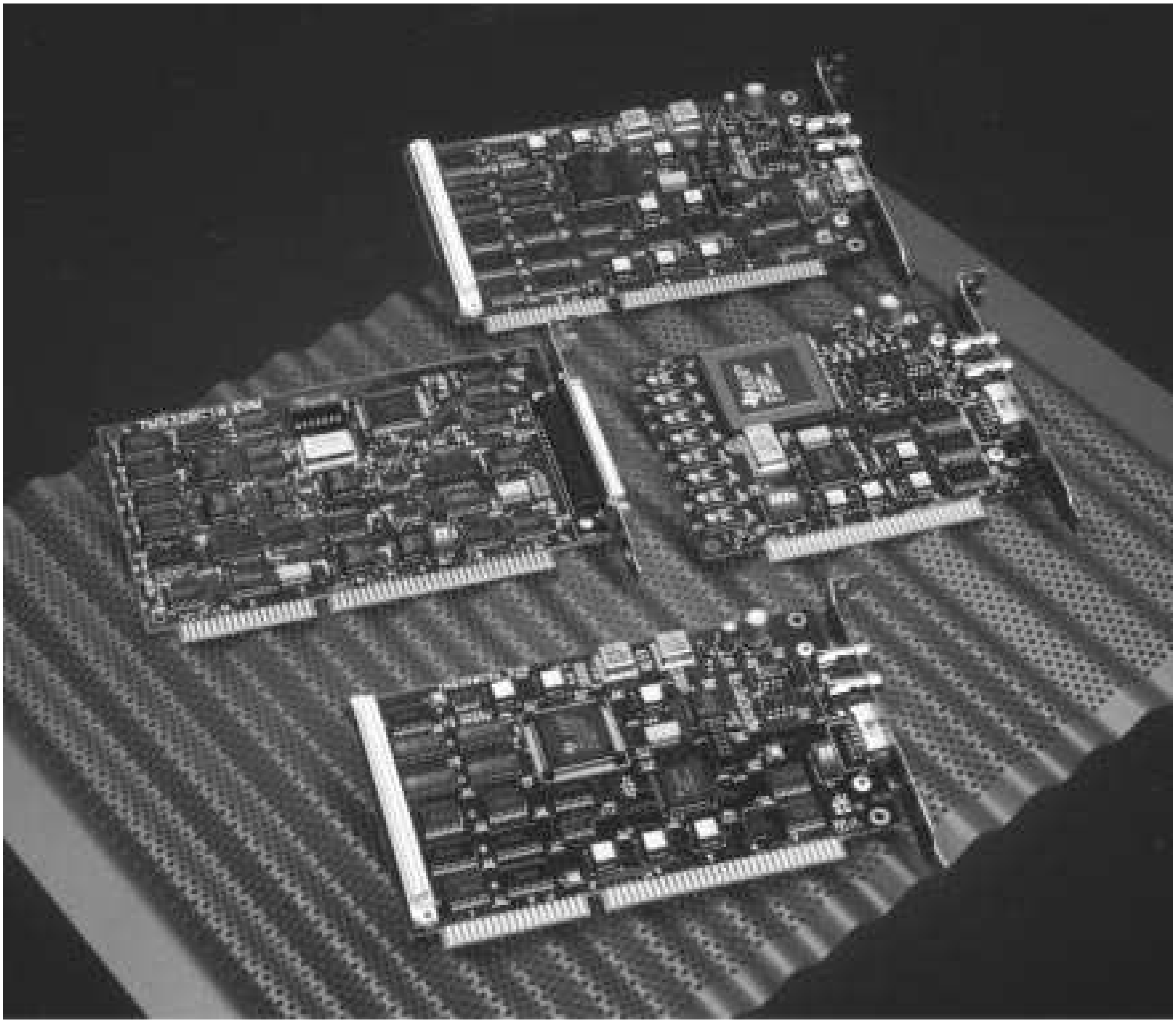
Common TMS320 EVM features include:

- ❑ Modification/display of memory and registers
- ❑ Assembler/linker
- ❑ Software single-step and breakpoint capabilities
- ❑ On-board memory
- ❑ Host upload/download capabilities
- ❑ I/O capability
- ❑ HLL-debug interface

14.3.2.1 TMS320C30 EVM

The ’C3x evaluation module (see Figure 14–3) hosts a ’C30 DSP on board to allow full-speed verification of ’C3x code.

Figure 14–3. TMS320 EVMs



The ’C3x EVM enables you to benchmark and evaluate code in real time while the device is operating at 33 MHz in the rich development environment of the ’C3x assembler/linker and C/assembly source debugger interface. Applications can be benchmarked and tested easily with the analog-ready interface.

The ’C3x EVM comes complete with a PC half-card and software package. The EVM board contains:

- ❑ One ’C30 — a 33-MFLOPS, 32-bit, floating-point DSP
- ❑ 16K-word, zero wait-state SRAM, allowing coding of most algorithms directly on the board

❑ A speaker/microphone-ready analog interface for multimedia, speech, and audio applications development

❑ An external serial-port interface that can be used for connecting multiple EVMs or for extra analog interfacing

❑ A host port for PC communications

❑ Embedded emulation support through the SN74ACT8990 test-bus controller

The system also comes with all of the software required to begin application development on a PC host:

❑ The window-oriented, mouse-controlled interface supports downloading, executing, and debugging of assembly code or C code.

❑ The ’C3x assembler/linker is also included with the EVM. For HLL programming, the optimizing ANSI C and the Ada compilers are offered separately.

14.3.2.2 TMS320C50 EVM

The ’C5x EVM carries a ’C50 DSP on-board to allow full-speed verification of ’C5x code. The ’C50 DSP has 10K on-chip ROM; it is pin compatible with the ’C51 and ’C53 and its peripherals are identical to the peripherals of those devices. Therefore, the ’C5x EVM can be used to evaluate the suitability of the ’C50, ’C51, or ’C53 for a given application.

Because the ’C5x generation is upward source-code compatible with the ’C2xx generation, the ’C5x EVM is used to develop code for ’C2xx devices.

Key features of the ’C5x EVM include:

❑ 50-ns instruction cycle time

❑ 64K-external zero-wait-state SRAM on board

❑ Voice quality analog data acquisition through the TLC32046 analog interface circuit

❑ Standard RCA connector analog input and output for direct connections to microphone and speaker

❑ Embedded emulation support by way of the SN74ACT8990 test-bus controller (TBC)

❑ 16-bit bidirectional PC host-communications port

❑ I/O expansion bus for application use

❑ IBM PC-compatible 16-bit half card, mappable in one of four memory locations

14.3.2.3 TMS320C54x EVM

The ’C54x EVM is a PC/AT plug-in card that lets you evaluate certain characteristics of the ’C54x DSP to see whether the DSP meets your application requirements. You can also create your software to run on board or expand the system in a variety of ways.

The ’C54x EVM carries of ’C541 DSP on board to allow full-speed verification of ’C54x code. The ’C541 has 5K bytes of on-chip program/data RAM, 28K bytes of on-chip ROM, two serial ports, a timer, access to 64k bytes each of external program and data RAM, and an external analog interface for evaluation of the ’C54x family of devices for a given application.

The ’C54x EVM has the following features:

- ’C541 operating at 40 MIPS with 128K words of zero wait-state memory
- Voice-quality analog interface to line I/O or speaker/microphone (user-selectable) through standard RCA connectors
- External serial port
- Parallel I/O-expansion bus
- Two 16-bit bidirectional host-/target-communication channels; one channel contains 64 words of buffering
- Embedded emulation support based on the IEEE 1149.1 standard
- A single 16-bit ISA half-card, mappable to one of four I/O locations

14.3.2.4 TMS320C6x EVM

The TMS320C6x evaluation module (EVM) is a low-cost, general-purpose platform for the development, analysis, and testing of ’C6x digital signal processor (DSP) algorithms and applications. The ’C6x EVM allows you to evaluate the ’C6x DSP and algorithms to determine if your application requirements can be met.

The ’C6x EVM hardware design information and software APIs also provide a reference design that can be used to ease your own ’C6x-based hardware and software development.The ’C6x EVM has a ’C6201 DSP onboard that allows full-speed verification of ’C6x code with the included source debugger.

The ’C6x EVM has the following features:

- ❑ The EVM can be plugged into a peripheral component interconnect (PCI) expansion slot on your computer’s motherboard or be operated standalone on a desktop.
- ❑ the EVM provides a PCI interface, SBSRAM and SDRAM memory, a 16-bit audio codec, and embedded JTAG emulation support.
- ❑ Connectors on the ’C6x EVM provide DSP external memory interface (EMIF) and peripheral signals that enable its functionality to be expanded with custom or third-party daughterboards.
- ❑ The EVM is bundled with TMS320C6x code generation and source debugger tools, Windows 95 and NT drivers, host PC and DSP software APIs, example applications with source code, and various utility applications. The bundled package provides an integrated package that allows you to quickly evaluate the ’C6x DSP’s performance and develop custom applications.
- ❑ The EVM provides a ’C6x hardware reference design that can assist you in the development of your own ’C6x-based products. In addition to providing a reference for interfacing the DSP to various types of memories and peripherals, the design also addresses power, clock, JTAG, and PCI controller interfaces.

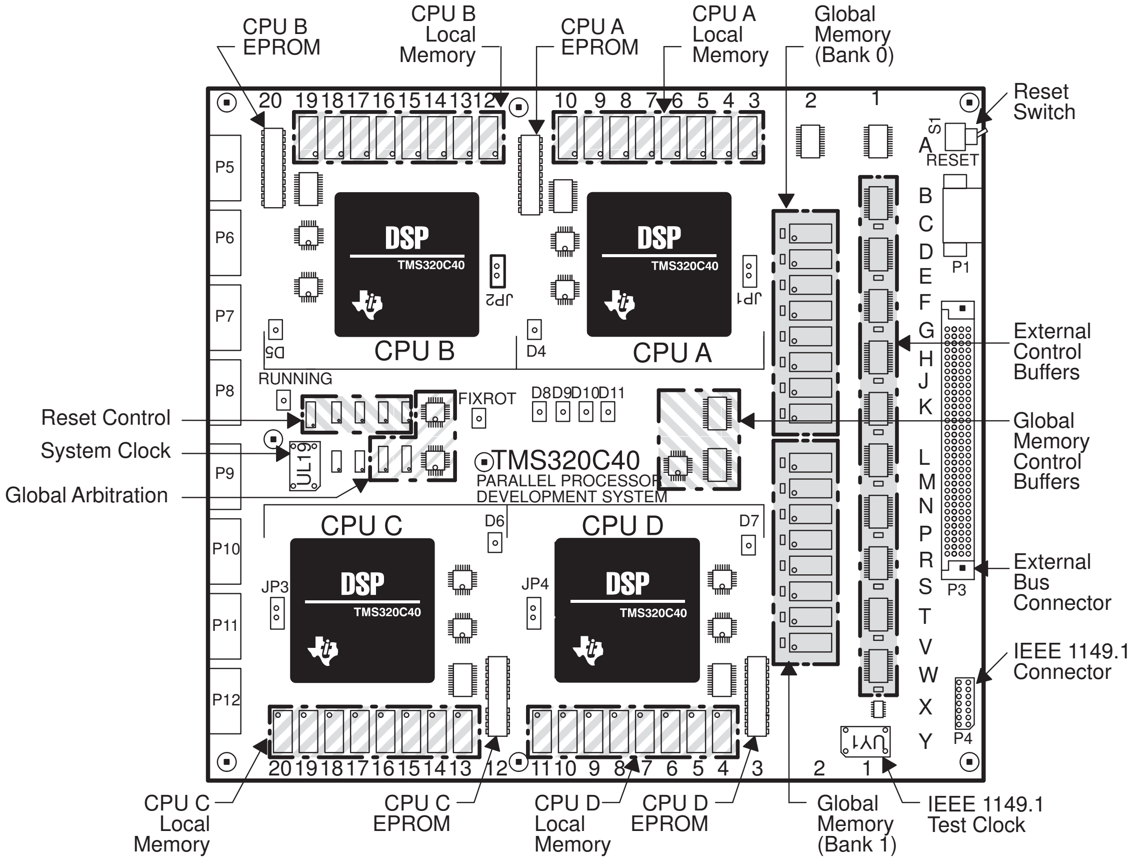
14.3.2.5 TMS320C4x Parallel Processing Development System

The ’C4x parallel processing development system (PPDS) (see Figure 14–4) is the first development board designed exclusively to evaluate and develop parallel-processing, floating-point software applications. You can develop, benchmark, and evaluate code in real time in a rich development environment with the power and speed of the ’C4x PPDS. For a complete system, the XDS510 is necessary.

Key features of the PPDS include:

- ❑ Four on-board ’C40 parallel processors. Each ’C40 is supported by a local bus consisting of:
  - ■ 64K × 32-bit words of zero wait-state SRAM
  - ■ 8K bytes of EPROM
- ❑ 128K × 32-bit words of one wait-state SRAM on a shared global bus
- ❑ An expansion bus connector that provides an external interface to the shared global-memory bus
- ❑ Eight external communication connectors that provide an interface for connecting off-board ’C40s and external peripherals to the PPDS ’C40s.
- ❑ An IEEE Standard 1149.1-compliant (JTAG) test connector that serves as an interface for connecting the XDS510 in-system emulator

Figure 14–4. The TMS320C40 PPDS Board Layout



The PPDS is placed on the desktop and is controlled through the XDS510, available separately. The PPDS is shipped with a dedicated desktop stand and its own 20-A, 50-W power supply. You also need a C compiler.

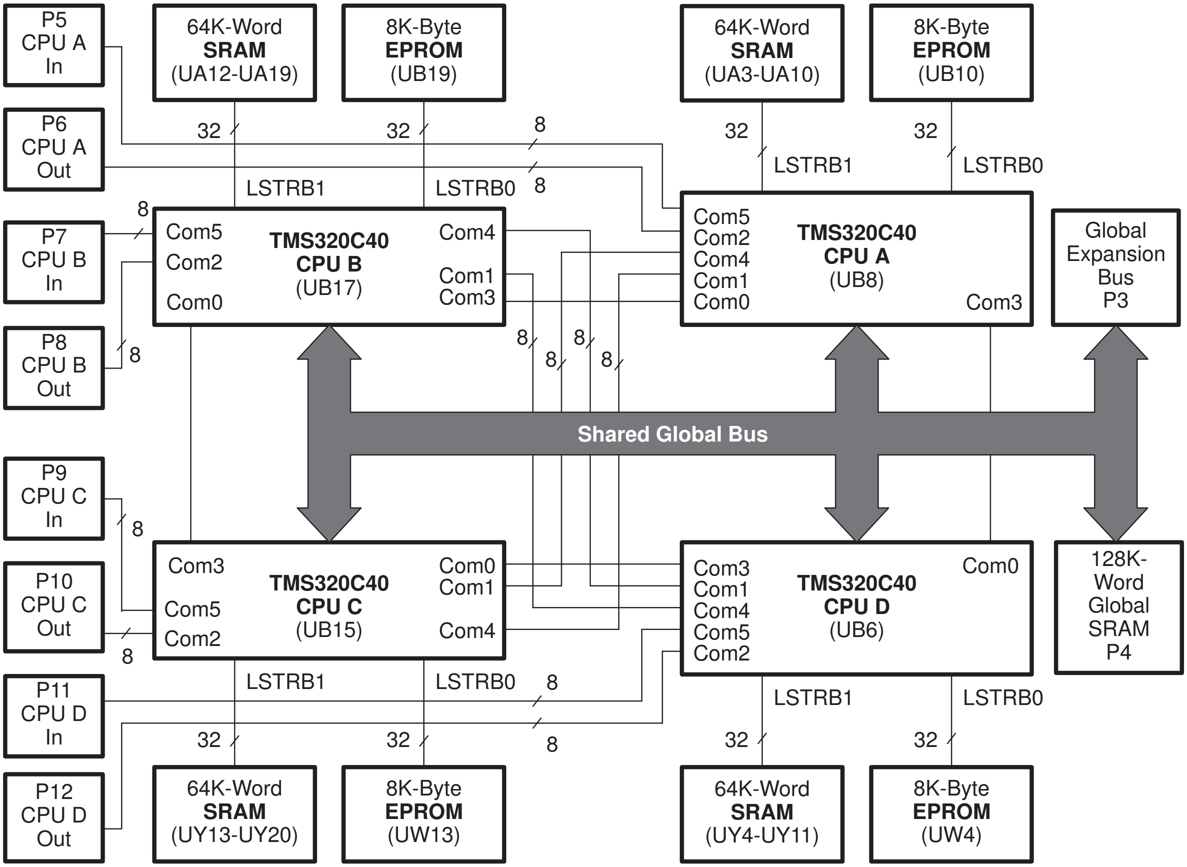
Each ’C40 on the PPDS has direct connections to each of the other ’C40s in the system through the communication ports, allowing you to experiment with various parallel-processing topologies that are best suited for your end application. In addition, each ’C40 also has two communication ports pinned out to external connectors on the left edge of the board, allowing other ’C40-based boards or peripheral boards to be connected to the ’C40s on the PPDS.

The ’C40s are also connected on a shared bus (see Figure 14–5) that has arbitration logic to decide which ’C40 receives access to the shared bus at any given time. The shared bus is brought to a connector, allowing DRAM, data acquisition, and other shared resources to be added to the PPDS.

Even though the PPDS is used with the XDS510, each ’C40 has its own source-level debugging window for code development.

These features give you the flexibility to distribute tasks between multiple processors and to develop, benchmark, and debug multiprocessing algorithms.

Figure 14–5. TMS320C40 PPDS Block Diagram



14.3.3 TMS320 Emulators

The TMS320 extended development systems (XDSs) are powerful, full-speed emulators used for system-level integration and debug. TI provides in-system scan-based emulators (XDS510/XDS510WS).The XDS510/XDS510WS emulator is currently available for the ’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x, and ’C8x DSPs.

14.3.3.1 Scan-Based Emulators (XDS510/XDS510WS)

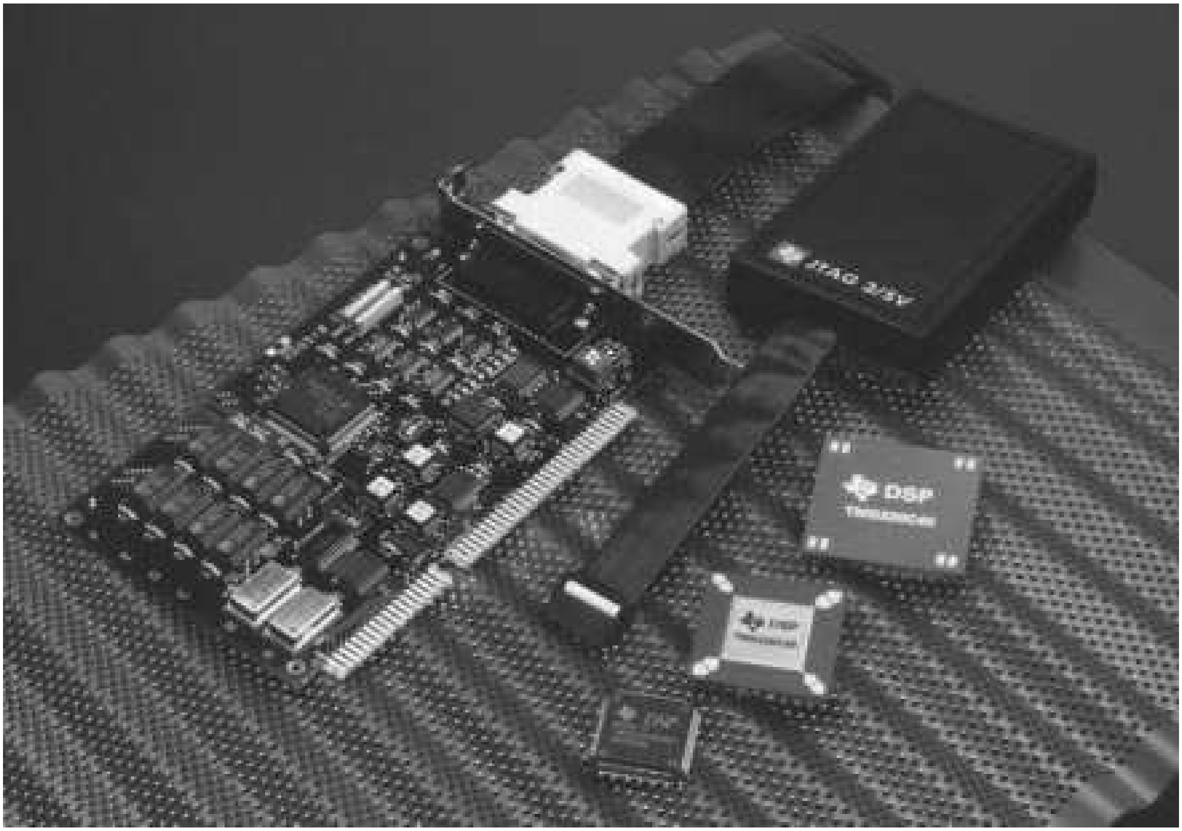
Scan-based emulation is a unique, nonintrusive approach to system emulation, integration, and debug. This approach was conceived and developed by TI to address hardware and software characteristics (reduced internal bus visibility, highly pipelined architectures, fast cycle times, and high-density packaging) that are inherent to sophisticated very-large-scale integration (VLSI) systems.

Scan-based emulation eliminates special bond-out emulation devices, target cable/buffer signal degradation, and the mechanical and reliability problems associated with target connectors and surface-mount packaging. With scan-based emulation, your program can execute in real time from internal or external target memory; no extra wait states are introduced by the emulator at any clock speed.

The TMS320 DSP device’s architecture implements scan-based emulation through internal, shift-register scan paths accessed by a single serial interface. The scan paths provide access to internal device registers and state machines, allowing complete visibility and control. This nonintrusive approach even operates in a production environment where the DSP is soldered into a target system.

The XDS510/XDS510WS emulators are user-friendly, PC- or Sun-based development systems, that have all the features necessary to perform full-speed, scan-based emulation with the ’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x, and ’C8x DSPs. They are the first scan-based emulators that are capable of parallel processing. These emulators make it possible to develop hardware and software and to integrate the hardware and software with the target system. A revolutionary five-wire interface acts as a scan path to every memory and register location in the DSP device (see Figure 14–6). The XDS510WS offers the same functionality for the SPARC workstation.

Figure 14–6. TMS320 XDS510 Scan-Based Emulators



Key features of the XDS510 include:

- ❑ Full-speed execution and monitoring of the device within your target system through a 14-pin target connector (12-pin target to support MPSD emulation on ’C3x)
- ❑ Global run/stop/breakpoint of parallel-processing DSPs
- ❑ HLL debugging interface
- ❑ Software breakpoint/trace and timing with up to 200 software breakpoints
- ❑ Hardware breakpoint/trace on all program and data addresses
- ❑ Single-step execution
- ❑ Interfacing and debugging with C/assembly source debugger
- ❑ Loading/inspection/modification of all registers and memory
- ❑ Benchmarking of execution time of clock cycles

Full-speed emulation and monitoring of the target system is performed serially through a cable that runs from the XDS510 to the target system. The scan path controls the device within the targeted application and provides access to all the registers, as well as to the internal and external memory of the device. Since program execution takes place in the DSP device of the target system, there are no timing differences during emulation. This new emulation technology offers significant advantages over traditional emulators. These advantages include:

- ❑ No cable-length-transmission-line problems
- ❑ Nonintrusive system
- ❑ No loading problems on signals
- ❑ No artificial-memory limitations
- ❑ Common-screen interface for easy usage
- ❑ Easy installation
- ❑ In-system emulation
- ❑ No variance from the device's data sheet specifications

Software breakpoints allow program execution to be halted at a specified instruction address. When a given breakpoint is reached, the program stops execution. At this point, the status of the registers and of the CPU is available. Their contents are visible in the appropriate windows; to view the contents of other memory locations, only one command is required.

Software trace lets you view the state of the device when a breakpoint is reached. This information can be saved in a file for future analysis. Software timing allows you to track the clock cycles between breakpoints for benchmarking of time-critical code.

Single-step execution gives you the capability to step through the program one instruction at a time. After each instruction, the status of the registers and CPU is displayed. This provides greater flexibility during software debugging and helps reduce the development time.

Object code can be downloaded to any valid memory location (program or data) via the scan path interface. Downloading a 1K-byte object program typically takes 100 ms. In addition, by inspecting and modifying the registers while single-stepping through a program, you can examine and modify program code or parameters.

The XDS510 is supported by the TMS320 standard debugger's interface for fast, easy debugging of C and assembly source code.

The emulator’s configurability gives your system flexibility. You can configure both memory and screen color. The address range, memory type, and access type assigned to each location can be configured also. The memory map, which may include EPROM, SRAM, DRAM, SDRAM, and on-chip memory and peripherals, can be configured to reflect the actual peripheral environment of the target system, including wait states and access privileges.

The ’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x, and ’C8x XDS510 emulator packages include:

❑ XDS510 emulator PC board

❑ IEEE Standard 1149.1 (JTAG[†]) or MPSD target cable

❑ ’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x, or ’C8x user interface software (sold separately)

All XDS510 systems use the IEEE 1149.1 target cable, except the ’C3x XDS510 which uses the MPSD cable.

The XDS510 emulator operates on a PC-AT system and requires one 16-bit slot.

[†] IEEE Std 1149.1-1990, IEEE Standard Test Access Port and Boundary-Seam Architecture

14.4 Code Composer—An Integrated Development Environment (IDE)

Texas Instrument's Code Composer™ is an alternative solution to the standard TMS320 debugger's interface described in Section 14.1, TMS320 Debugger's Interface (C/Assembly Source Debugger). Code Composer, a fully integrated development environment (IDE) for TI DSPs, offers, for the first time, features and productivity gains found only in mainstream software development environments (such as Microsoft's Visual C++™). Code Composer also includes features specifically tuned for the DSP software designers and is currently available for the 'C2xx, 'C3x, 'C4x, 'C5x, and 'C54x. Nearly all of Code Composer's main features are new to the DSP market and unite the environment functionality of high-and low-level DSP debuggers, signal probes and scopes (as found in block diagram tools), and graphical profiling, all in a tightly integrated MS Windows™ (and Windows 95) application.

14.4.1 Primary Features

Code Composer includes all the major features provided by the HLL debugger (see Section 14.1). The following advanced features are also included in the Code Composer environment:

- ❑ **A fully integrated environment using TI's compiler (IDE)*.*** Code Composer integrates a project management system, built-in editor, and full debugging and profiling capabilities in a single Windows environment.

- ❑ **Project management for C and DSP assembly files.** The project management system keeps track of all files and their dependencies. This allows Code Composer to save you compile time by recompiling only those files that have changed since the last compile.

- ❑ **Tightly integrated editor tuned for writing C and DSP assembly code.** The built-in editor of Code Composer supports dynamic syntax highlighting for both C and assembly files. Syntax highlighting makes your code easier to read and can help you spot critical syntax errors very easily.

- ❑ **Background compiling while editing and debugging.** There is no need to shell-out to a DOS environment to execute your compiler/assembler tools. Code Composer automatically launches these tools in its environment. Errors are highlighted in Code Composer's build window. You can double click on errors to go directly to the point where the error occurred.

- ❑ **Multiprocessor support under native MS Windows with floating PDM.** Code Composer supports full multiprocessing in Windows 3.1 and Windows 95. The parallel debug manager (PDM) allows you to broadcast commands to all (or the selected group) of processors.

- ❑ **Graphic window scope probes to watch signals at any algorithm point.** Graphical display windows allow the user to view signals in time domain or frequency domain. For frequency domain graphs, the FFT is performed on the host; this allows you to view the spectrum of the interested signal without any modification to its DSP code. Graphical displays can also be connected to a probe point. A probe point (when set at a particular location in the algorithm) specifies when the graphical display window should be updated. This allows you to take a snapshot of the signal when execution of the code reaches that point.

- ❑ **File probes to extract or inject signals/data at any algorithm point via files.** Instead of reading signals in real time, Code Composer allows you to stream signals from/to your PC. This allows you to simulate your algorithm (on the DSP target) with known samples.

- ❑ **Graphical profiling.** Code Composer's profiling capabilities are integrated within its environment.

- ❑ **Execution of user's DOS program in the background ("system" command).** You can execute any DOS programs from within Code Composer and have the output piped to Code Composer's output window. This allows you to integrate your own applications to Code Composer.

- ❑ **State-of-the-art watch window.** Code Composer's watch window allows you to enter any C expression or any variable of interest. Structures, arrays, and pointers can easily be recursively expanded or collapsed. This allows you to drill down complex structures.

- ❑ **Algebraic disassembly window.** The disassembly window gives you the option to view the disassembled opcodes in algebraic C format making the disassembled code much easier to read.

- ❑ **Help on the target DSP.** On-line help on the DSP instruction and registers means that you don't have to carry your User's Guide everywhere.

- ❑ **User extensible.** The GO DSP Corporation's Extension Language (GEL) allows you to add your own menu items to Code Composer's menu bar.

14.4.2 Using Code Composer as a Complete Development Environment

Code Composer tightly integrates Texas Instruments compiler, assembler, and linker tools into its development environment. With Code Composer, users can launch TI's tools from the menu bar and see the compiler output, piped directly to a window, as it happens. Error messages are highlighted, and double clicking on the error message opens the source file and positions the cursor on the line in question. DOS-based TI tools are smoothly multitasked in the background under Windows so users can easily edit source files, debug programs, and compile, all at the same time. Code Composer keeps track of all files and file dependencies (C and assembly source) in a project. The user can choose to compile individual files, to build all files in the project, or to build the project incrementally. Easy-to-use dialog boxes are available for compiler, assembler, and linker options.

The Code Composer watch window allows you to easily "drill down" through complex structures. Variables such as arrays, structures, and pointers can be expanded and collapsed recursively by simply placing the cursor on the variable of interest and pressing the ENTER key. In addition, variables added to the watch window can be edited by simply double clicking on the desired variable. Any C-expression as well as a GEL function can be added to the watch window. By adding a GEL function to the watch window, the GEL function is executed at every breakpoint. From within the called GEL function, more complex tasks can then be performed and the results piped to any output window.

Probe points allow you to observe signals or to inject or extract data at a certain point in the algorithm. Probe points can be connected to any instruction point and memory area. When a designated point in the algorithm is reached, the connected signal probe captures data from the target DSP and displays it appropriately. If a file is connected to the designated point, data is streamed between a specified memory area and a file. Once the operation is complete, execution continues. This feature allows the developer to take snapshots of the target memory and inject or extract data via files at particular points in their algorithm. Using the animation feature, the developer can observe signals and execution to any detail by using real signals from the PC's disk, all with no change to the source code.

Chapter 15

TMS320 Technical Support

In addition to development tools, Texas Instruments provides extensive technical support to assist customers during product design. This support is detailed in the following sections.

Topic **Page**

15.1 DSP World Wide Web Site

TI DSP maintains a site on the World Wide Web where you can find technical information about TI digital signal processing solutions. The data presented covers TI DSPs, linear and mixed-signal devices, development tools, and products of TMS320 third-party developers, including both hardware and software. Also available are technical resources including the On-Line DSP Lab ™ and the 320 Hotline On-Line ™ . The On-Line DSP Lab allows you to “test drive” DSP design tools for free, right from your PC. Currently, the lab features the ’C3x and the ’C54x software tools and evaluation module (EVM) with debugger. The 320 Hotline On-Line is a searchable database that allows you to research various topics and retrieve technical answers 24 hours a day.

Technical documentation including user’s guides, application reports, Designer’s Notebook pages, data sheets, reference guides, and publications are also available.

The world wide web site address is **http://www.ti.com/dsps**.

15.2 Technical Documentation

A wide variety of technical literature is available to assist you through the design cycle. These documents include product and preview bulletins, data sheets, user's and reference guides, over 2500 pages of application notes, and textbooks offered by Prentice-Hall, John Wiley & Sons, and Computer Science Press. The latest product and documentation updates are given in the TMS320 quarterly newsletter, *Details on Signal Processing,* the TMS320 DSP Bulletin Board Service (BBS), and on the Internet in the DSP Solutions world wide web home page. To inquire about available TMS320 literature, call the TI Literature Response Center at:

(800) 477-8924

The DSP Solutions world wide web site (http://www.ti.com/dsps) contains electronic versions of most of the TMS320 technical documentation, including all data sheets and application reports currently available.

The following list describes the general contents of each major category of technical documentation available through the TI Literature Response Center.

❑ **Product bulletins** and product briefs give an overview of the devices and of development support within the TMS320 family, presenting capabilities, diagrams, and hardware and software applications.

❑ **User's guides** for TMS320 processors provide detailed information regarding the architecture of the device, its operation, assembly language instructions, and hardware and software applications.

❑ **Data sheets** include features, electrical specifications, block diagrams, timing characteristics, and mechanical data for each device.

❑ **Application reports** describe the theory and implementation of selected TMS320 applications, including algorithms, code, and block, schematic, or logic diagrams. Currently, there are over 2500 pages of application reports to support the TMS320 family.

❑ **The TMS320 newsletter,** *Details on Signal Processing*, which is published quarterly, updates TMS320 customers on product information and industry trends. To receive a free subscription to this newsletter in the United States, call (800) 477–8924, x 3543.

❑ **Technology brochures** provide an overview of various implementations of DSP technology in applications such as wireless communications.

15.3 DSP Application Reports

There are many reports to assist customers in designing TMS320 DSP applications. Some are available through the TMS320 DSP BBS; others are published in the TMS320 application books. Table 15–1 lists the application reports available.

You can view most of these application reports electronically at the DSP Solutions www site (http://www.ti.com/dsp).

Source code, which can be used to reduce design time and to bring TMS320-based products to market faster, is available through the BBS. Refer to Section 15.8, TMS320 DSP Bulletin Board Service, for more information.

Note:

Please contact the TI Product Information Center at (800) 477–8924 to request this literature or see the DSP Solutions www site.

Table 15–1. Application Reports

Application	Topic	Location/ Literature No.	Device
Data Communications	*Viterbi Implementation on TMS320C5x for V.32 Modems* (Mansoor Chishtie)	SPRA033	TMS320C5x
	Digital Line Echo Canceller Implementation on TMS320C5x DSP (Kevin McCoy and Mansoor Chishtie)	SPRA033	TMS320C5x
Digital Cellular	*Cellular Phone: A Functional Analysis* (B.I. Pawate and Mansoor Chishtie)	SPRA033	TMS320C5x
	IS-54 Simulation Package (John Crockett/Steve Popik/Elliot Hoole)	SPRA033	TMS320C5x
	U.S. Digital Cellular Error Correction Coding Algorithm Implementation on TMS320C5x (Mansoor Chishtie)	SPRA033	TMS320C5x
	A Performance Study of Two TMS320C53-Based Viterbi Algorithms for U.S. Digital Cellular Radio (Mansoor Chishtie)	SPRA033	TMS320C5x
	A TMS320C53-Based Advanced FEC Scheme for U.S.D.C. Radio (Mansoor Chishtie)	SPRA033	TMS320C5x
	IS-54 Digital Cellular Modem Implementation on TMS320C5x (Balaji Srinivasan)	SPRA033	TMS320C5x
	Mobitex Modem Implementation Using the TMS320C6x (Etienne Resweber)	SPRA033	TMS320C5x
	Equalization Concepts (David Smalley)	SPRA033	TMS320C5x
	C5x-Based Equalizer Implementation for IS-54 (Elliot Hoole)	SPRA033	TMS320C5x
Digital Control	*TMS320C31 Embedded Control*	SPRU083	TMS320C31

† Refer to *DFT/FFT and Convolution Algorithms* by C. S. Burrus and T. W. Parks, published by John Wiley & Sons.

Table 15–1. Application Reports (Continued)

Application	Topic	Location/ Literature No.	Device
DSP Interface	*Interfacing the TMS320C3x to the TLC3204x Analog Interface Chip*	SPRA021	TMS320C3x
	A Low-Cost TMS320C30 Host Interface	SPRA021	TMS320C30
	Engine Knock Detection Using Spectral Analysis Techniques With a TMS320 DSP	SPRA039	TMS320C30
	Interfacing Memory to the TMS320C32 DSP	SPRA040	TMS320C32
	A DSP-Based PCMCIA Card Design (Raj Chirayil)	SPRA033	TMS320C5x
	Interfacing the TMS320C8x into SDRAM	SPRA055	TMS320C8x
	Interfacing the TMS320C8x into DRAM	SPRA056	TMS320C8x
Image/Graphics	*TMS320C3x DSPs Supercharge 3-D Graphics*	SPRA024	TMS320C3x
	DSP-Based Handprinted Character Recognition (Alan Josephson)	SPRA033	TMS320C5x
	TMS320C8x Fundamental Graphic Algorithms	SPRA069	TMS320C8x
DSP Routines	*Digital Filter Design Programs (FIR/IIR)*	BBS	TMS320
	Minimizing Quantization Effects Using TMS320 Digital Signal Processor Family	SPRA035	TMS320
	EDN Magazine 1st-Generation (TMS320C1x) Benchmarks	BBS	TMS320C1x
	TMS320C10 FFT Routines	BBS, JW†	TMS320C10
	TMS320C14 Examples (from *TMS320C14 User's Guide*)	BBS	TMS320C14
	Self Test	SPRA012	TMS320C1x/C2x
	EDN Magazine 2nd-Generation (TMS320C2x) Benchmarks	BBS	TMS320C2x
	TMS320C2x-Based Adaptive Line Enhancer Design	SPRA012	TMS320C2x
	'C25 Complex 256-Point FFT	BBS	TMS320C25
	'C25 Real 256-Point FFT	BBS	TMS320C25
	TMS320C25 Examples (from *TMS320C25 User's Guide*)	BBS	TMS320C25
	Preliminary Documentation/Source Listing for C26 Boot ROM	BBS, SPRA017	TMS320C26
	An 8x8 Discrete Cosine Transform Implementation on the TMS320C25 or the TMS320C30	SPRA017	TMS320C25/C30
	Implementation of Adaptive Filters With the TMS320C25 or the TMS320C30	BBS	TMS320C25/C30
	EDN Magazine 3rd-Generation (TMS320C3x) Benchmarks	BBS	TMS320C3x
	'C30 Serial Port Example Program	BBS	TMS320C30

† Refer to *DFT/FFT and Convolution Algorithms* by C. S. Burrus and T. W. Parks, published by John Wiley & Sons.

Table 15–1. Application Reports (Continued)

Application	Topic	Location/ Literature No.	Device
DSP Routines	*C Callable Functions to Initialize TMS320C30 Cache and Wait State Control*	BBS	TMS320C30
	Doublelength Floating-Point Arithmetic on the TMS320C30	BBS	TMS320C30
	Example Programs for Operation of TMS320C30 Serial Ports	BBS	TMS320C30
	An Implementation of FFT, DCT, and Other Transforms on the TMS320C30	BBS	TMS320C30
	TMS320C30 Echo Cancellation Program	BBS	TMS320C30
	TMS320C30 Examples (from *TMS320C30 User's Guide*)	BBS	TMS320C30
	TMS320C30 Matrix Multiply Benchmark Report	BBS	TMS320C30
	TMS320C30 Utility Programs	SPRA031	TMS320C30
	A Parallel Approach for Solving Matrix Multiplication on the TMS320C4x DSP	BBS	TMS320C4x
	Implementing Circular Buffers With Bit-Reversed Addressing	BBS	TMS3206201
	Fast Fourier Transform Algorithms of Real–Valued Sequences Within the TMS320 Family	BBS	TMS320C6201
	Sensorless Control With Kalman Filter on TMS320 Fixed-Point DSP	BBS	TMS320C6201
Miscellaneous	*Implementation of the Kaish Circuit Lockout System With the TMS320 Family*	SPRA022	TMS320
	TMS320C8x Transform 3 Command	SPRA069	TMS320C8x
	TMS320C8x Transform 4 Command	SPRA069	TMS320C8x
	Fill Draw Colored Trapezoid Command	SPRA069	TMS320C8x
	TMS320C8x Draw Colored Line Commands	SPRA069	TMS320C8x
	TMS320C80 PP Integer and Floating-Point Math	SPRA069	TMS320C80
Speech Coding/ Recognition	*Firmware-Programmable μC Aids Speech Recognition*	SPRA012	TMS320
	A TMS320C30-Based LPC Vocoder	SPRA021	TMS320C30
	Calculation of TMS320C5x Power Dissipation Application Report	SPRA030	TMS320C5x
	Theory and Implementation of the Digital Cellular Standard Voice Coder: VSELP on the TMS320C5x (Jason Macres)	SPRA033	TMS320C5x
	Implementation of Speaker-Independent Speech Recognition on TMS320C2x/C5x (Raj Pawate and Peter Robinson)	SPRA033	TMS320C5x
	Automated Dialing of Cellular Telephones Using Speech Recognition	SPRA033	TMS320C5x
	Acoustic Echo Cancellation Algorithms and Implementation on TMS320C8x	SPRA063	TMS320C8x

† Refer to *DFT/FFT and Convolution Algorithms* by C. S. Burrus and T. W. Parks, published by John Wiley & Sons.

Table 15–1. Application Reports (Continued)

Application	Topic	Location/ Literature No.	Device
Telecommunications	*Modified Goertzel Algorithm in DTMF Detection*	SPRA066	TMS320C8x
Tools	*C-Coding Tips for Application Specific Processors*	SPRA021	TMS320C30
	TMS320C30 Evaluation Module Overview	SPRA021	TMS320C30
	How TMS320 Tools Interact with the TMS320C32's Enhanced Memory Interface	SPRA048	TMS320C32
	Accessing Status and Control Fields and I/O Ports in the TMS320CXX HLL Debugger	BBS	TMS320C6201

† Refer to *DFT/FFT and Convolution Algorithms* by C. S. Burrus and T. W. Parks, published by John Wiley & Sons.

15.4 TMS320 DSP Designer’s Notebook Pages

The Designer’s Notebook Pages (DNPs) are short application notes written by TI’s engineering teams, customers, and TMS320 third parties. These application notes provide helpful tips for designing and programming with the TMS320 DSPs. The DNPs are available to download from the TMS320 BBS at (281) 274-2323, or from the DSP Solutions www site at http://www.ti.com/dsps, or from the Internet FTP site at ftp.ti.com. Table 15–2 lists the Designer’s Notebook topics currently available. Also available is the TMS320 DSP Designer’s Notebook, Volume 1 (SPRT125), which contains DNP numbers 1–60.

Table 15–2. Currently Available Designer’s Notebook Pages for TMS320 DSPs

No.	Topic
1	’C3x Block Repeat
2	Avoiding False Interrupts on the ’C3x
3	Bit-Reversed Addressing Without Data Alignment on the ’C3x
4	Optimizing Control Algorithms on ’C5x
5	TMS320C30 Addressing up to 68 Gigawords
6	’C5x EVM Provides for Audio Processing
7	Circular Buffering in Second Generation DSPs
8	Bit-Reversed Addressing in C on the ’C3x
9	Sharing Header Files in C and Assembly
10	Initializing the Fixed-Point EVM’s AIC
11	TMS320C25 Logical Shifts in Parallel With ALU Operations
12	TMS320C40 Boot Loader Selection
13	Reducing System Power Requirements
14	Interfacing the TMS320C31 to A/D and D/A Devices
15	Efficient Coding on the TMS320C5x
16	TMS320C40 DMS Memory Transfer Timing
17	Designing with TMS320C40 Comm Ports: Part 1
18	Creating a Delay Buffer on a TMS320C2x EVM
19	Dual-Access Into Single-Access RAM on a ’C5x Device
20	A Simple Way to Terminate Unused TMS320C40 Comm Ports
21	TMS320C5x Interrupts
22	Fast Logarithms on a Floating-Point Device
23	Switching from Bootloader to MP Mode With the TMS320C31

Table 15–2. Currently Available Designer's Notebook Pages for TMS320 DSPs (Continued)

No.	Topic
24	TMS320C5x Interrupt Response Time
25	TMS320C2x/'C5x EVM AIC Initialization and Configuration
26	A Novel Way of Using TMS320C40 Cache
27	Hardware UART for TMS320C3x
28	Using VRAMs and DSPs for System Performance
29	Using the RBIT on the TMS320E25
30	Addressing Peripherals as Data Structures in C
31	Interrupts in C on the TMS320C3x
32	TMS320C40 Emulator Tips
33	Floating-Point C Compiler: Tips and Tricks — Part 1
34	Guidelines for Decoupling Capacitors on DSP Designs
35	TMS320C5x Interrupts and the Pipeline
36	Improved Context Save/Restore Performance and Interrupt Latency for ISRs written in C
37	Serial ROM Boot
38	Mastering the 'C4x DMA
39	Bootload of C Code for the TMS320C5x
40	How to Convert a HEX30 Output File Into a Linkable Assembly File
41	Supporting External DMA Activity to Internal RAM for TMS320C5x Devices With the PZ Package
42	Binary Search Algorithm on the TMS320C5x
43	Random Number Generation on a TMS320C5x
44	Using a TMS320C80 Serial Port as an Asynchronous RS-232 Port
45	Fast TMS320C5x External Memory Interface
46	TMS320C5x Memory Paging (Expanding its Address Reach)
47	TMS320C5x Clock Modes
48	TMS320C5x Wait States
49	Clocking Options on the TMS320C5x
50	TMS320C5x DSK Analog I/O
51	Bootloading a 'C4x Network—Part 1: Direct Connect System
52	Emulator Processor Access Timeout
53	Extending Fixed-Point Math Dynamic Range With Minimum Cycles

Table 15–2. Currently Available Designer's Notebook Pages for TMS320 DSPs (Continued)

No.	Topic
54	Accessing TMS320C5x Memory-Mapped Registers in CC5XREGS.H
55	C Routines for Setting Up the AIC on the TMS320C5x EVM
56	How Can Comb Filters be Used to Synthesize Musical Instruments on a TMS320 DSP?
57	Initializing the TMS320C5x DSK Board
58	Debugging a Full-Duplex UART on the TMS320C3x
59	Designing Macros for the TMS320C5x
60	Accessing States and Control Fields and I/O Ports in the TMS320Cxx HLL Debugger
61	Multipass Linking
62	Linking C Data Objects Separate From the .bss Section
63	Shared Memory Interface With a TMS320C5x DSP
64	Accessing TMS320C54x Memory-Mapped Registers in C—C54XREGS.H
65	Interfacing External Memory to the TMS320C5x DSK
66	Interfacing a TMS320C2xx or 'C5x DSP to a TLC548 8-Bit A/D Convertor
67	Interfacing a TMS320C2x, TMS320C2xx, or TMS320C5x DSP to an 8-Bit Boot EPROM
68	Using the Circular Buffers on the TMS320C5x
69	Viewing TMS320C8x Register Bit Fields and Memory-Mapped Registers in the HLL Debugger
70	Parity Generation on the TMS320C54x
71	μ-Law Compression on the TMS320C54x
72	Interfacing Two Analog Interface Circuits to One TMS320C5x Serial Port
73	Writing TMS320C8x PP Code Under the Multitasking Executive
74	Reading a 16-Bit Bus With the TMS320C5x Serial Port
75	Interfacing a TMS320C3x DSP to the TLC320AD58C 18-Bit Stereo A/D Converter
76	Interfacing a 20-MSPS TLC5510 Flash A/D Converter to TMS320C2xx and TMS320C5x Fixed-Point DSPs
77	IDLE2 Instruction on a TMS320C51 DSP When Using a Divide-by-One Clock Option

15.5 University Textbooks

Numerous TMS320 textbooks have been published to support digital signal processing research and education. These textbooks (listed below according to publisher) are designed to aid in the understanding of DSP applications and implementations using the TMS320 family.

❑ **Prentice-Hall**
College Technical and Reference Division
Upper Saddle River, NJ 07458
(201) 236-7000
(800) 223-1360
(800) 947-7700

- ***Digital Signal Processing Applications With the TMS320 Family***, volume 1 (K.S. Lin, editor), is a reference guide for developing applications. The guide consists of application reports, published articles, and technical reports covering a wide range of DSP applications. Source code for the application algorithms is given in the text as well as on two floppy disks (included). ISBN #013212-4661, U.S. $89.00
- ***Digital Signal Processing Applications With the TMS320 Family***, volume 2 (P. Papamichalis, editor), contains additional TMS320C1x and TMS320C2x applications. Applications reports include DSP interface and algorithm debug techniques, as well as data communications, telecommunications, and digital control applications for the TMS320 family. Source code for application algorithms is given in the text as well as on floppy disks (included). ISBN #013212-9523, U.S. $69.00
- ***Digital Signal Processing Applications With the TMS320 Family***, volume 3 (P. Papamichalis, editor), focuses primarily on ’C3x applications, such as implementation of Fast Fourier Transforms (FFT), DCT, and other transforms, and on a wide range of floating-point algorithms. Source code for application algorithms is given in the text as well as on floppy disks (included). ISBN # 013212-9604, U.S. $78.00

❑ **John Wiley & Sons**
Professional Reference and Trade Group
605 Third Avenue
New York, NY 10158-0012
(212) 850-6000

- ***DFT/FFT and Convolution Algorithms*** (C.S. Burrus and T.W. Parks) completely covers the theory and computation of Discrete Fourier Transforms (DFT). The three main approaches to FFTs (Cooley-Tukey, prime-factor, and Winograd) are also described in detail. TMS320 coding examples are included. ISBN #0471-819328, U.S. $64.95

- ***Digital Filter Design*** (T.W. Parks and C. S. Burrus) is a comprehensive guide to digital filter design methodologies, covering basic theory as well as working programs. Properties, design, approximations, and implementation of FIR and IIR filters are discussed in detail. Design examples using TMS320 DSPs are included. ISBN #0471-828963, U.S. $75.95

- ***Theory and Design of Adaptive Filters*** (J.R. Treichler, C.R. Johnson, Jr., and M.G. Larimore) introduces the fundamental concepts, design techniques, and application guidelines of adaptive filters. This text discusses the analysis and design of the three basic classes of adaptive filters: FIR, IIR, and adaptive property restorative filters. Several TMS320 design examples are presented. ISBN #0471-832200, U.S. $108.00

- ***Digital Signal Processing With the TMS320C25*** (R. Chassaing and D. Horning) describes the architecture and instruction set of the TMS320C25. Theoretical discussion is followed by practical examples supported by projects and applications. A disk containing all the programs used and a filter design package is included. ISBN #0471-510661, U.S. $74.95

- ***Digital Signal Processing With C and the TMS320C30*** (Ralph Chassaing) describes the architecture and instruction set of the TMS320C30. Programming examples using both C and TMS320C30 are included throughout the text. A disk of programming examples is included. ISBN #0471-577774, U.S. $59.95

- ***A Simple Approach to Digital Signal Processing*** (Craig Marven and Gillian Ewers) takes the reader step-by-step through the most basic signal processing concepts to more complex functions and devices, including sampling, filtering, frequency transforms, data compression, and even DSP design decisions. ISBN #0471-152439, U.S. $39.95

- ***Active Noise Control Systems — Algorithms and DSP Implementations*** (Sen M. Kuo) emphasizes the practical aspects of active noise control (ANC) systems using a signal processing and DSP implementation perspective. The principles of adaptive signal processing are combined with experimental results and practical issues including the application of these structures and algorithms using C and assembly programs on the TMS320C25 and TMS320C30. ISBN #0-471-13424-4, U.S. $79.95

❑ **Computer Science Press**
41 Madison Avenue
New York, NY 10010
(212) 349-8263

Digital Signal Processing (Richard Haddad and Thomas Parsons) is a comprehensive guide to digital control theory and the design and implementation of digital control systems. This book is a collection of more than 40 application reports and papers from well-known industry experts. Many reports explore the advantages of implementing control algorithms with digital rather than analog techniques. Specific applications discussed include computer peripherals, motion control/robotics, power electronics, and automotive. ISBN #0710-782065, U.S. $59.95

❑ **Plenum Publishing**
233 Spring Street
New York, NY 10013-1578
(800) 221-9369
(212) 620-8000
(212) 807-1047 (Fax)

Communication System Design Using DSP Algorithms (Steven A. Tretter) is a comprehensive set of experiments used to explore digital signal processing and communication systems theoretical concepts by implementing them on actual hardware (TMS320C30) in real time. ISBN # 0-306-45032-1, U.S. $55.00

15.6 Technical Articles Bibliography

This section lists key articles and papers that have been published about the TMS320 DSPs. Readers who are interested should be able to locate these articles/papers at their local public or university library.

For a complete listing of articles and papers, refer to the bibliographies in *Digital Signal Processing Applications With the TMS320 Family,* volume 1, volume 2, and volume 3 (literature numbers SPRA012, SPRA016, and SPRA017, respectively) and in *Digital Control Applications With the TMS320 Family* (SPRA019).

❑ **cDSP Tools**

- "DSP Design Takes Top-Down Approach," Andy Fritsch and Kim Asal, *EE Times DSP Series Part III,* July 17, 1995.
- "The Growing Spectrum of Custom DSPs," Gene Frantz and Kun Lin, *EE Times DSP Series, Part II,* April 18, 1994.

❑ **Control**

- "Real-Time Control," Gregg Bennett, *Appliance Manufacturer,* May 1995.
- "DSPs Advance Low-Cost 'Green' Control," Gregg Bennett, *EE Times DSP Series Part II,* April 17, 1995.
- "A Greener World Through DSP Controllers," Panos Papamichalis, *DSP & Multimedia Technology,* September 1994.

❑ **DSP Technology**

- "Application Guide with DSP Leading-Edge Technology," Y. Nishikori, M. Hattori, T. Fukuhara, R. Tanaka, M. Shimoda, I. Kudo, A. Yanagitani, H. Miyaguchi, and others, *Electronics Engineering,* November 1995.
- "Function-Focused Chipsets: Up the DSP Integration Core," Panos Papamichalis, *DSP & Multimedia Technology,* March/April 1995.
- "On-Chip Multiprocessing Melds DSPs," Karl Guttag and Doug Deao, *EE Times DSP Series Part III,* July 18, 1994.

❑ **DSP Tools/Development Support**

- "Easing JTAG Testing of Parallel-Processor Projects," Tony Coomes, Andy Fritsch, and Reid Tatge, *Asian Electronics Engineer,* Manilla, Philippines, November 1995.
- "DSP Design Takes Top-Down Approach," Andy Fritsch and Kim Asal, *EE Times DSP Series Part III,* July 17, 1995.

- “The Digital Signal Processor Development Environment,” Greg Peake, *Embedded System Engineering,* United Kingdom, February 1995.
- “Third-Party Support Drives DSP Development for Uninitiated and Experts Alike,” Panos Papamichalis, *DSP & Multimedia Technology,* December 1994/January 1995.

❑ DSP Solutions Applications

- “Digital Signal Processing Solutions Target Vertical Application Markets,” Ron Wages, *ECN,* September 1995.

❑ General-Purpose DSP

- “Fixed or Floating? A Point Question in DSPs,” Jim Larimer and Daniel Chen, *EDN,* August 3, 1995.
- “Toward an Era of Economical DSPs,” John Cooper, *EE Times DSP Series Part I,* January 23, 1995.
- “The Wide World of DSPs,” Jim Larimer, *Design News,* June 27, 1994.

❑ Graphics/Imaging

- “High-Tech Copiers To Improve Images and Reduce Paperwork,” Karl Guttag, *Document Management,* July/August 1995.
- “A Single-Chip Multiprocessor DSP for Image Processing–TMS320C80,” Dr. Ing. Dung Tu, *Industrie Elektronik,* Germany, March 1995.

❑ Hard Disk Drive

- “Digital Signal Processors Boost Drive Performance,” Tim Adcock, *Data Storage,* September/October 1995.

❑ Military

- “Beware of BAT: DSPs Add Brilliance to New Weapons Systems,” Panos Papamichalis, *DSP & Multimedia Technology,* October 1994.

❑ Multimedia

- “DSPs Do Best on Multimedia App,” Doug Rasor, *Asian Computer World,* October 9–16, 1995.
- “Host-Enabled Multimedia: Brought to You by DSP Solutions,” Panos Papamichalis,” *DSP & Multimedia Technology,* September/October 1995.
- “Choose DSPs for PC Signal Processing,” Panos Papamichalis, *DSP & Multimedia Technology,* January/February 1995.

❑ **Power Dissipation**

- “Telecom Future Driven by Reduced Milliwatts per DSP Function,” Panos Papamichalis, *DSP & Multimedia Technology,* May/June 1995.
- “Approaching the No-Power Barrier,” Jon Bradley and Gene Frantz, *Electronic Design,* January 9, 1995.

❑ **Speech/Voice**

- “Speech Recognition,” P.K. Rajasekaran and Mike McMahan, *Wireless Design & Development,* May 1995.
- “DSPs: Speech Recognition Technology Enables,” Gene Frantz and Gregg Bennett, *I&CS,* May 1995.
- “DSP and Speech Recognition, An Origin of the Species,” Panos Papamichalis, *DSP & Multimedia Technology,* July 1994.

❑ **Telecommunications**

- “GSM: Standard Strategien, und Systemchips,” Edgar Auslander, *Electronik Praxis,* Germany, October 6, 1995.
- “Developing Nations Take Shine to Wireless,” Russell MacDonald, Kara Schmidt, and Kim Higden, *EE Times,* October 2, 1995.
- “Integration Shrinks Digital Cellular Telephone Designs,” Fred Cohen and Mike McMahan, *Wireless System Design,* November 1994.
- “Telecom Future Driven by Reduced Milliwatts per DSP Function,” Panos Papamichalis, *DSP & Multimedia Technology,* May/June 1995.

15.7 TMS320 Newsletter

The TMS320 newsletter, *Details on Signal Processing*, is published quarterly to update TMS320 customers on product information and industry trends. It covers TMS320 DSP solutions, documentation, new third-party hardware and software support, new applications, technical information about TMS320 products, development tool updates, worldwide contacts for support, design workshops, seminars, conferences, and the TMS320 university program.

To be added to the mailing list, in the United States, write to:

Texas Instruments Incorporated
Attn: DSPS Newsletter Staff
P.O. Box 1443, M/S 722
Houston, Texas 77251-1443

or call (800) 477–8924, x 3543.

15.8 TMS320 DSP Bulletin Board Service

The TMS320 DSP BBS is a telephone-line computer bulletin board that provides access to information about the TMS320 family. The BBS is an excellent means of communicating specification updates for current or new DSP application reports as they become available. It also serves as a means to trade programs with other TMS320 users. In addition, ROM-code algorithms may be submitted by secure electronic transfer via the TMS320 BBS.

The BBS contains TMS320 source code from the more than 2000 pages of application reports written to date. These programs include macro definitions, FFT algorithms, filter programs, ADPCM algorithms, echo cancellation, graphics, control, companding routines, and sine-wave generators.

You can access the BBS with a computer and modem. The modem must be able to communicate at a data rate of either 1200, 2400, 9600, 14,400, or 28,800 bps. A character length of eight bits is required, with one stop bit and no parity. The telephone number of the bulletin board is (281) 274-2323 for U.S.A. or for Europe. There is a 90-minute access limit per day, and it is open 24 hours.

To log onto the BBS, first connect your modem to the telephone line (or if you are using an external modem, connect the modem between the telephone and the computer), and then dial **(281) 274-2323**. Once the call has been established, the BBS responds by displaying a welcome message and asking for your first name, last name, and password (defined when you first access the BBS).

The first time you access the BBS, you have very limited capabilities. You must answer completely the questionnaire that is automatically displayed when your password is defined before the system operator (SYS OP) can upgrade your ID to higher access capabilities. Once the questionnaire has been completed, your ID will be upgraded for nonrestricted access to the BBS, which includes downloading of TMS320 application programs, updates, and device and development tool information. Additional help on any menu can be obtained by using the **?** (question mark) command.

To transfer a file from the BBS to your system, enter the **F** (Files Area) command from the main menu. This allows access to all available BBS files. To get a list and a brief description of the different files available, select the **L** (List file) command. To download a file, choose the **D** command. Select the default protocol by selecting *Your Setting* on the main menu. The BBS supports the most popular protocols, including ASCII, XMODEM, XMODEM-CRC, YMODEM, and ZMODEM. The BBS then asks for a filename. Enter the filename you are interested in. If you use the MS-DOS wildcard characters ***** and **?** and you are using the YMODEM or ZMODEM protocols, you can download several files sequentially. Refer to the MS-DOS manual for the details on wildcard characters. With the proper protocol, the BBS waits for you to start the file transfer.

To log off the BBS, enter the **G** (Goodbye) command. The BBS updates the log-in data and waits for the next BBS customer.

See page viii for other worldwide BBSs available.

15.9 TMS320 DSP ftp Site

The TMS320 ftp site is an Internet mirror of the BBS located at ftp.ti.com. In this site, you can find most of the information available on the BBS.

To access the ftp site, fpt to ftp.ti.com using anonymous login. The path is /mirrors/ tms320 bbs. If you have a browser such as Netscape or Internet Explorer, you can type ftp:/ftp.ti.com/mirrors/tms320bbs to access the ftp site. This site can also be accessed from the DSP solutions www site at http://www.ti.com/dsps.

15.10 TMS320 DSP Technical Hotline

Texas Instruments maintains a hotline that answers TMS320 technical questions. It responds to specific questions regarding TMS320 device problems, development tools, third-party support, consultants, documentation, upgrades, and new products. The hotline is open Monday from 9:30 a.m. to 6:00 p.m. Central Standard Time, and Tuesday–Friday from 8:30 a.m. to 6:00 p.m. Central Standard Time.

To obtain the most complete information regarding your product, first consult your product documentation then go to the DSP Solutions WWW at:

http://www.ti.com/dsps

This site includes the 320 Hotline On-line™, a service that can help answer your questions.

If your question is not answered there, gather all the information that applies to your problem. With your information, manuals, and products close at hand, you can then call the TMS320 DSP Technical Hotline.

When you report your problem, please provide the hotline personnel with the following information:

Name:

Company:

Phone number:

Email address:

Product:

Software version:

Host platform: (PC/SPARC/HP/etc.)

OS type and version:

Target platform:

Problem title: (Provide a one-line description of the problem.)

Problem type: (production stop/critical/noncritical/nuisance)

Problem description: (Provide a detailed description of the problem.)

Attachments: (C source code, assembly code, linker command file, etc.)

You can submit your information via facsimile machine, or you can submit information via electronic mail.

Telephone, fax numbers, and e-mail addresses are given on page viii.

Questions on pricing, delivery, and availability should be directed to the nearest TI field sales office or to the TI Semiconductor Product Information Center (PIC).

Note:

For worldwide TMS320 DSP Hotline support, see page viii.

Chapter 16

TMS320 Third-Party Support

TI offers a worldwide network of hundreds of third parties and consultants who support the TMS320 DSP family.

Topic **Page**

16.1 TMS320 Third-Party Program

TI has the industry’s strongest DSP third-party program and is committed to meeting customers’ ever-increasing demands for increased product capability, cost-effectiveness, ease of use and faster time-to-market. Working as a team, TI and its third parties are striving to make this a reality for customers using DSP. The Texas Instruments DSP Third Party Program consists of a collection of high-growth, high-technology companies committed to providing DSP products and services that include hardware development support, software development support, software algorithms and libraries and consulting services. The network boasts hundreds of members delivering thousands of products supporting TMS320 DSP solutions.

Current third-party products include:

Hardware Development Support

- ❑ Development boards
  - Data acquisition
  - Development hardware
  - Multiprocessing OEM boards
  - Plug-in boards / add-on cards
  - TIM modules
  - Standalone boards
- ❑ Logic analyzers
- ❑ Emulators
- ❑ Device programmers

Software Development Support

- ❑ Debuggers
- ❑ Simulators
- ❑ Simulation models
- ❑ Software development utilities
  - Filters
  - Signal analyzers
  - C-code generators
  - DSP BIOS

Algorithms and Libraries

- Audio algorithms
- Image algorithms
- Integrated development environment / application specific systems
- Operating systems
- Runtime support libraries
- Speech recognition / synthesis algorithms
- Telecommunication algorithims
- Vocoders

Consultants

- Contract engineering design resources
- Turnkey designs
- Hardware and software integration with multiple module/vendors
- Training
- Research and development

16.2 Where to Find a Third-Party Solution

Access URL **http://www.ti-dsp.com/** for a complete list of third-party products and services. If you are looking for up-to-date information, consult our easy-to-use product tables, or you can search the database by:

- ❑ Company name
- ❑ Product category
- ❑ Keyword search
- ❑ Product name
- ❑ Devices supported

Each listing contains comprehensive information on third-party products and network companies. Periodically, the above-mentioned on-line resource is consolidated into the Texas Instruments *Third-Party Application Software and Development Guide* (literature number SPRC013A) on CD-ROM. .In addition, the CD contains a comprehensive collection of TI DSP related information including technical data sheets, training documents, application notes and TI's DSP newsletter archives.

To expand our network, we observe emerging, innovative, high-tech companies with products and services that could complement TI's DSPS products. For comprehensive information on the program and related events, consult the TI DSP Third-Party Program homepage at: **http://www.ti.com/sc/docs/dsps/develop/3party.htm**.

Chapter 17

TMS320 Seminars and Workshops

Texas Instruments offers a wide array of up-to-date technical product seminars and design workshops through its Technical Training Organization (TTO) to assist designers in developing the skills needed to implement their ideas quickly, produce a quality product, and shorten time to market. Applications assistance is also offered through local Customer Design Centers.

The DSP design workshops give design engineers hands-on experience using the latest TMS320 products, development tools, and design techniques. These workshops go beyond the standard lecture format. The exercises and lab experiments start with the basics and move quickly into hands-on exercises. In these workshops, the student learns by doing, not just listening or observing. The workshops are designed to help customers shorten the design cycle, control development costs, and solve design challenges.

Topic **Page**

17.1 TTO Services

The TTO offers 3- or 4-day DSP design workshops to assist users in the development of TMS320-based designs. These workshops are held at a TTO location or at a customer-selected site. Six different DSP design workshops are available, each covering a different generation of the TMS320 DSP family (see Table 17–1).

The main objective of each workshop is to demonstrate hardware and software techniques for implementing current DSP algorithms using a TMS320 digital signal processor. Exercises provide hands-on experience with the development tools needed for a quick start in designing with the TMS320 family.

Table 17–1. Texas Instruments Technical Workshops

Workshop Titles	Length, days
TMS320C2xx Family Digital Signal Processor Workshop	4.0
TMS320C3x Family Digital Signal Processor Workshop	3.5
TMS320C4x Family Digital Signal Processor Workshop	3.5
TMS320C5x Family Digital Signal Processor Workshop	4.0
TMS320C54x Family Digital Signal Processor Workshop	4.0
TMS320C6x Family Digital Signal Processor Workshop	4.0
TMS320C8x Family Digital Signal Processor Workshop	4.0

17.1.1 Introduction to Technical Training

Success today means meeting competitive standards of quality, variety, customization, convenience, and timeliness. No longer is productivity the single measure of success. At Texas Instruments, your success in the design process is foremost on our minds.

Technological advancement and stiff competition are constantly driving the need to shorten the cycle time from design concept to market availability. State-of-the-art skills and practical application techniques are required in order to implement ideas quickly with a high degree of quality and value.

To help you meet these challenges, TI offers you the latest training and consultation on advanced technologies. Lectures led by skilled instructors and hands-on lab exercises using the latest TI development tools accelerate your learning experience. Real-world examples help you apply TI’s advanced technology to your system. You will discover more productive ways of gaining that competitive edge.

TI workshops help you successfully:

- ❑ Solve design challenges
- ❑ Shorten the design cycle
- ❑ Control development costs

17.1.2 Registration and General Information

TI encourages early registration at least four weeks in advance. To register for a workshop within the United States, or for schedule information on courses in North America, call the TTO Central Registration office at (972) 644–5580 or (972) 917–3894 or a TI-authorized distributor. Workshops in Europe and Asia are offered through local Customer Workshops; for more information, contact the nearest Customer Design Center (see Section 17.3). For information on courses outside the United States, contact the nearest Customer Design Center.

Workshop enrollment is normally limited to 12 per class. All classes begin at 8:30 a.m.; full-day classes end at 5:00 p.m., and half-day classes end between 12:00 and 2:00 p.m. Tuition fees charged for our courses include study materials, use of facilities, personal copies of product documentation, design aids, continental breakfast, and lunch. Cancellation within 14 days before a workshop's start date incurs a $500 cancellation fee.

Additional Information:

- ❑ If you have specific dietary constraints, please contact your training location.

- ❑ Smoking is not permitted in Texas Instruments facilities.

- ❑ The suggested attire is business casual.

17.1.3 TI Technical Training Locations

- ❑ **Boston**
  400-1 Totten Pond Road
  Waltham, MA 02154
  (781) 895–9185

- ❑ **Dallas**
  7839 Churchill Way
  Dallas, TX 75251
  (972) 917–3894

❑ **Irvine**
1920 Main Street, Suite 900
Irvine, CA 92714
(408) 383–2363

❑ **San Jose**
2825 North First Street, Suite 200
San Jose, CA 95134
(408) 383–2363

17.1.4 TI Technical Training Design Workshops

Texas Instruments Design Workshops provide the background and experience necessary for you to complete a TI microprocessor-based design. These workshops are valuable to engineers in any size company or project. They are of benefit in any of the following design stages:

❑ System design
❑ Processor evaluation / selection
❑ Hardware / software design

17.1.4.1 Who Should Attend

These workshops are targeted to any engineer with a basic knowledge of microprocessor design. Familiarity with basic assembly coding and embedded system design are helpful.

17.1.4.2 What You Will Learn

Design workshops focus on the architecture and instruction set. The courses cover the following information:

❑ System-level considerations

❑ Techniques for optimizing assembly and C code

❑ Coding algorithms to take advantage of the processor's architecture, buses, and special hardware features

❑ Writing a program from start to finish

❑ Hardware interface issues

17.1.4.3 Hands-On Learning

Using PC-based development tools, you will spend about 40 percent of the time writing code and solving problems. Numerous labs provide practice in such areas as using various addressing modes, implementing DSP algorithms, and working with peripherals.

17.1.5 Fixed-Point DSP Workshops

The fixed-point DSP workshops are designed for individuals using the ’C2xx, ’C5x, and ’C54x family processors. Workshops are similar in structure and style, and topics cover the features and application issues for the included generations.

17.1.5.1 TMS320C2xx Workshop

This workshop covers the currently available ’C2xx products, which include the ’C203, ’C209, and the ’C2xLP core architecture (for use in developing cDSP designs). The material can be customized depending on your needs.

Topics covered include:

- ❑ Architectural overview
- ❑ Assembly language tools
- ❑ Data addressing modes
- ❑ Basic programming techniques
- ❑ Advanced programming techniques
- ❑ Numerical issues
- ❑ DSP fundamentals
- ❑ Logical operations
- ❑ Interrupts
- ❑ Logic/memory interfacing, special peripherals
- ❑ cDSP/ASIC design considerations
- ❑ Using the C compiler
- ❑ Design support

17.1.5.2 TMS320C5x Workshop

This workshop covers the ’C5x CPU architecture which is common between all the ’C5x device variations. The specific features of each device are compared and discussed.

Topics covered include:

- ❑ Architectural overview
- ❑ Assembly language tools
- ❑ Data addressing modes
- ❑ Basic programming techniques
- ❑ Advanced programming techniques
- ❑ Numerical issues
- ❑ DSP fundamentals
- ❑ Logical operations
- ❑ Interrupts
- ❑ Hardware interfacing
- ❑ Serial ports and multiprocessor features
- ❑ Using the C compiler

17.1.5.3 TMS320C54x Workshop

Along with the ’C54x architecture and programming, you will learn to use the special hardware on this device, such as the Viterbi accelerator and buffered serial port. A study is made of the special (hardware supported) instructions useful for implementing standard telecom algorithms.

Topics covered include:

- ❑ Introduction to the ’C54x architecture
- ❑ Assembly language tools
- ❑ Data addressing modes
- ❑ Basic programming techniques
- ❑ Advanced programming techniques and pipeline issues
- ❑ Advanced mathematics
- ❑ Numerical issues
- ❑ DSP fundamentals
- ❑ Advanced DSP and specialized DSP hardware
- ❑ Interrupts
- ❑ Hardware interface
- ❑ Serial ports and special interface features

17.1.6 Floating-Point DSP Workshops

The floating-point DSP workshops are designed for individuals using the ’C3x and ’C4x family processors. Workshops are similar in structure and style, and topics cover the features and application issues for the included generations.

17.1.6.1 TMS320C3x Workshop

The ’C3x represents the first generation of TI’s 32-bit floating-point devices.

Topics covered include:

❑ Introduction to the ’C3x architecture
❑ Assembly language tools
❑ Data memory addressing
❑ CPU operations and floating point
❑ Enhanced CPU operations
❑ Pipeline and cache
❑ Direct memory access
❑ Interrupts
❑ Special addressing modes
❑ Memory interface
❑ Timers and serial ports
❑ Using the C compiler

17.1.6.2 TMS320C4x Workshop

The second family in TI’s floating-point DSP lineup, the ’C4x workshop expands on the basic 32-bit DSP architecture by covering the advanced communication ports and DMA coprocessor.

Topics covered include:

❑ Introduction to the ’C4x architecture
❑ Assembly language tools
❑ Data memory addressing
❑ Basic CPU operations
❑ Floating-point and parallel instructions
❑ Pipeline and cache
❑ Interrupts
❑ Special addressing modes
❑ Memory interface
❑ Bootloader
❑ Communication ports
❑ DMA coprocessor
❑ Using the C compiler
❑ Parallel application development
❑ Design support

17.1.7 TMS320C6x Workshop

This workshop is tailored for hardware or software design engineers who plan on designing with the ’C6x DSPs. Experience with digital systems and basic knowledge of assembly language programming is required. Basic knowledge of the C programming language is helpful, although not required.

Topics covered include:

- ’C6x architecture and instruction set
- Use of PC-based development tools
- Optimization of assembly code
- Interrupts and traps
- Peripherals and input/output techniques
- ’C6x hardware
- System design

Following is an outline of the TMS320C6x workshop:

- Architectural overview
- Introduction to pipeline
- Getting started with C
- Getting started with Assembly
- Using the Linker
- System initialization
- Accessing data and program control
- Assembly optimization
- Using the assembly optimizer
- Optimizing the performance of C
- Logical and bit–field operations
- Numerical issues
- Interrupts
- Memory I/F
- Internal program memory and cache
- DMA and boot loader
- Serial port
- HPIF and system considerations

17.1.8 TMS320C8x Workshop

This workshop focuses on designing with the ’C80 multimedia video processor. The workshop covers device architecture, how the on-chip CPUs and resources function and interact, and programming with the parallel processors (PPs) and master processor. You will also learn how to use the development tools, including C compilers, assemblers, source-level debuggers, and the parallel debugging environment.

Topics covered include:

- ’C80 architecture
- Functional block interaction
- Programming the PPs
- Programming the executive and master processor, calling library routines, and partitioning tasks
- Using development tools

Following is an outline of the TMS320C6x workshop:

- Introduction to ’C80 systems
- Introduction to C source debugger
- Crossbar and on-board memory organization
- Parallel processors (PP)
- Transfer controller (TC)
- Master processor (MP)
- ’C80 software architecture
- Video controller (VC)
- System design

17.1.9 Course and Workshop Information

For further information on courses and schedules in North America, call the Product Information Center at (972) 644-5580 or go to the TI semiconductor home page on the Internet: www.ti.com/sc/docs/pic/home.htm. Workshops in Europe, Asia, and other locations outside the United States are offered through local Customer Design Centers; for more information, contact the nearest Customer Design Center (see Section 17.3).

17.2 Design Services

The TI technical staff can offer applications assistance with customer designs through local Customer Design Centers (see Section 17.3). Services include:

- Design assistance
- Simulation
- Emulation

Each Customer Design Center uses up-to-date development systems, including workstations and personal computers, plus demonstration, test, and evaluation equipment. TI staff designers use fully equipped laboratories to provide efficient design assistance.

The first step in a successful design is an explanation of the project's parameter: production requirements, design function(s), and price. The results of these discussions allow TI and a customer to explore:

- Design/cost trade-offs
- Product implementation options

After the various trade-offs/options are selected and approved, Texas Instruments can provide further assistance in the design of a customer's product, sharing a mutual goal of bringing a successful product to market as quickly as possible.

17.3 Customer Design Center Locations

Table 17–2 and Table 17–3 give the worldwide locations of the TI Customer Design Centers.

Table 17–2. Customer Design Center North American Locations

North American Locations	
ATLANTA Texas Instruments 5515 Spalding Drive Norcross, GA 30092 (404) 662–7950	**NORTHERN CALIFORNIA** Texas Instruments 2825 North First Street, Suite 200 San Jose, CA. 95134 (408) 383–2363
BOSTON Texas Instruments 400-1 Totten Pond Raod Waltham, MA 02154–1263 (781) 895–9196	**SOUTHERN CALIFORNIA** Texas Instruments 1920 Main St., Suite 900 Irvine, CA 92714 (714) 660–8140
CHICAGO Texas Instruments 515 W. Algonquin Road Arlington Heights, IL 60005 (708) 640–2909	**OTTAWA** Texas Instruments Canada, Ltd 301 Moodie Drive, Suite 102 Nepean, Ontario Canada, K2H 9C4 (613) 726–1970
DALLAS Texas Instruments 7839 Churchill Way Park Central V, MS 3984 Dallas, TX 75251 (972) 917–3881	**MEXICO CITY** Texas Instruments de Mexico Alfonso Reyes 115 Col. Hipodromo Condesa Mexico, D.F., Mexico 06170 (52) (5) 515–6081 (52) (5) 515–6249
INDIANAPOLIS Texas Instruments 550 Congressional Blvd., Suite 100 Carmel, IN 46032 (317) 573–6400	

Table 17–3. Customer Design Center International Locations

International Locations	
AUSTRALIA Texas Instruments Australia Ltd. 6–10 Talavera Road, North Ryde New South Wales, Australia 2113 (61) (2) 8789000	**JAPAN** (Tokyo) Texas Instruments Japan Ltd Ms Shibaura Building 9F 4–13–23 Shibaura Minato-Ku, Tokyo, JAPAN 108 (81) (3) 3769–8700
BRAZIL Texas Instruments Electronicos Rua Paez Leme, 524–7 Andar 05424 Sao Paulo, Brazil (55) (11) 815–6166	**JAPAN** (Osaka) Texas Instruments Asia LTD Osaka Branch Nissho-Iwai Bldg 5F 2–5–8 Imabashi Chuou-Ku Osaka, Japan 541 (81) (6) 204–1881
FEDERAL REPUBLIC OF GERMANY Texas Instruments Deutschland GMBH Kirchhorster Strasse 2 3000 Hannover 51, FR Germany (49) (511) 648021	**KOREA** Texas Instruments Korea Ltd. 28th Floor, Trade Tower 159 Samsung-Dong Kangnam-Ku, Seoul Trade Center P.O. Box 45 Seoul, Korea 135–729 (82) (2) 5512800
FEDERAL REPUBLIC OF GERMANY Texas Instruments Deutschland GMBH Haggertystrasse 1 8050 Freising, FR Germany (49) (8161) 80–0	**SINGAPORE** Texas Instruments Singapore (Pte) Ltd. Asia Pacific Division 101 Thomson Road #23–01 United Square Singapore 1130 (65) 2519818
FRANCE (Paris) Texas Instruments France 8–10 Avenue Morane Saulnier Boîte Postale 67 Velizy Villcoublay Cedex, France (33) (13) 0701001	**SWEDEN** Texas Instruments International Trade Corporation Box 30 S–164 93 Kista Isafjordsgatan 7, Sweden (8) 752–5800

Table 17–3. Customer Design Center International Locations (Continued)

International Locations	
HONG KONG Texas Instruments Hong Kong Ltd. 8th Floor, World Shipping Centre 7 Canton Road Kowloon, Hong Kong (852) 7351223	**TAIWAN** Texas Instruments Taiwan Ltd. Taipei Branch 10 Floor, Bank Tower 205 Tung Hua N. Road Taipei, Taiwan 105 Republic of China (886) (2) 7139311
ITALY (Milan) Texas Instruments Italia S.P.A. Centro Direzionale Colleoni Palazzo Perseo Via Paracelso, North 12 20041 Agrate Brianza, MI, Italy (39) (39) 63221	**UNITED KINGDOM** Texas Instruments Ltd. Regional Technology Center Manton Lane Bedford, England MK41 7PA (44) (234) 270111

Table 17–4. European Customer Training Infolines

Country	**Contact**	**Telephone Number**
France/Israel/Turkey/S. Africa	Sylviane Huguet	+33 1 30 70 11 57
Nordic	Marja Kinos	+46 87 52 58 36
United Kingdom	Claudia Dolente	+44 16 04 66 31 10
Benelux	Anne Marie Van Lint	+32 2 745 55 30
Italy/Spain/Portugal	Vanda Tomasi	+39 39 68 4 22 19
Central Europe	Martina Luther Beatrix Szeleczky	+49 81 61 80 44 86 +36 1 3 19 28 14

Chapter 18

TMS320 University Program

Texas Instruments believes it is important to train future engineers and encourages universities to do advanced research in the area of digital signal processing. TI has established the TMS320 University Program to make its resources available to universities and to assist in the incorporation of the TMS320 family information into engineering research and course curricula. The program provides considerable cost savings and expert technical assistance. Hundreds of universities are already taking advantage of this program.

Topic **Page**

18.1 Introduction to the TMS320 University Program

Through the TMS320 University Program, TI offers discounts on development tools, software donations, expert engineering assistance for university research, and third-party contacts. TMS320 documentation and technical hotline assistance can be found on the DSP Solutions World Wide Web (WWW) home page at:

http://www.ti.com/dsps

In addition, numerous TMS320 textbooks, including lab manuals and user's guides, are published to support digital signal processing research and education at the graduate and undergraduate levels (refer to Section 15.5, *University Textbooks*, for a description of the TMS320 textbooks).

Note:

Texas Instruments reserves the right to make changes at any time in its TMS320 University Program policies.

18.2 Development Tools Available to Universities

TMS320 DSPs including the ’C1x, ’C2x, ’C2xx, ’C3x, ’C4x, ’C5x, ’C54x, ’C6x, and ’C8x and associated development tools are available to universities at a discount. For purchase, contact any of TI’s authorized distributors.

The code-generation tools include the TMS320 macro assembler, linkers, and C compiler packages (includes assembler/linker).

The system integration and debug tools include XDS™, emulators, simulators, EVMs, DSP Starter Kits (DSKs), and adapters.

The DSP Teaching Kit (TMDS320015X) includes the following in one box:

❑ Five lectures on the basics of DSP

❑ ’C3x or ’C5x DSK card

❑ Power supply

❑ Connectors and cables

❑ Demonstration program, including source code

❑ Overhead transparencies

❑ Test material

❑ Student notes

❑ Lecturer’s Guide

❑ Supporting textbook: *”A Simple Approach to DSP”* by Craig Marven and Gillian Ewers (Wiley)

18.3 DSP Lab Workstation Recommendations

The DSP lab workstation, designed for use at the undergraduate and first-year graduate levels, allows students to practice the theory learned in an associated DSP course. Students can design DSP systems using practical examples and perform real-time DSP simulations. A DSP lab usually consists of four or five workstations. Workstations typically include the following:

- ❑ **’C2x Lab Workstation**
  - ■ PC or Sun/SPARC
  - ■ ’C2x simulator
  - ■ ’C2x EVM (or ’C2x DSK)
  - ■ ’C2x/’C5x optimizing ANSI C compiler package
  - ■ ’C2x/’C5x assembler/linker
- ❑ **’C2xx Lab Workstation**
  - ■ PC or Sun/SPARC
  - ■ ’C2xx simulator
  - ■ ’C5x EVM (or ’C5x DSK)
  - ■ ’C2xx optimizing ANSI C compiler/assembler/linker package
- ❑ **’C3x Lab Workstation**
  - ■ PC or Sun/SPARC
  - ■ ’C3x simulator
  - ■ ’C3x EVM (or ’C3x DSK)
  - ■ ’C3x/’C4x optimizing ANSI C compiler package
  - ■ ’C3x/’C4x assembler/linker
- ❑ **’C4x Lab Workstation**
  - ■ PC or Sun/SPARC
  - ■ ’C4x simulator
  - ■ ’C3x/’C4x optimizing ANSI C compiler
  - ■ ’C3x/’C4x assembler/linker
- ❑ **’C5x Lab Workstation**
  - ■ PC or Sun/SPARC
  - ■ ’C5x simulator
  - ■ ’C5x EVM (or ’C5x DSK)
  - ■ ’C2x/’C5x optimizing ANSI C compiler package
  - ■ ’C2x/’C5x assembler/linker

- ❑ **’C54x Lab Workstation**
  - ■ PC or Sun / SPARC
  - ■ ’C54x simulator
  - ■ ’C54x EVM
  - ■ ’C54x optimizing ANSI C compiler/assembler/linker package
- ❑ **’C6x Lab Workstation**
  - ■ PC or Sun / SPARC
  - ■ ’C6x simulator
  - ■ ’C6x EVM
  - ■ ’C6x optimizing ANSI C compiler/assembler/linker package
- ❑ **’C8x Lab Workstation**
  - ■ PC
  - ■ ’C8x PC software toolset

The following textbooks are recommended for the indicated workstation environments:

- ❑ **’C2x Workstation**
  - ■ *TMS320C2x User’s Guide* from Texas Instruments (literature number SPRU014C) (Prentice-Hall)
  - ■ *Digital Signal Processing With C and the TMS320C25,* by Chassaing and Horning (John Wiley & Sons)
- ❑ **’C2xx Workstation**

  TMS320C2xx User’s Guide from Texas Instruments (literature number SPRU127A) (Prentice-Hall)
- ❑ **’C3x Workstation**
  - ■ *TMS320C3x User’s Guide* from Texas Instruments (literature number SPRU031D)
  - ■ *Digital Signal Processing Applications With the TMS320C30 Evaluation Module* from Texas Instruments (literature number SPRA021)
  - ■ *Communication System Design Using DSP Algorithms,* by Steven A. Tretter (Plenum Publishing)
- ❑ **’C4x Workstation**

  TMS320C4x User’s Guide from Texas Instruments (literature number SPRU063A)

- ❑ **’C5x Workstation**

  TMS320C5x User’s Guide from Texas Instruments (literature number SPRU056B)

- ❑ **’C54x Workstation**

  TMS320C54x User’s Guide from Texas Instruments (literature number SPRU131B)Additional TMS320 DSP textbooks are available (see Section15.5,*UniversityTextbooks*, for a complete listing).

- ❑ **’C6x Workstation**

  TMS320C62xx Programmer’s Guide from Texas Instruments (literature number SPRU196A)

  TMS320C62xx Peripherals Reference Guide from Texas Instruments (literature number SPRU190A)

  TMS320C62xx CPU and Instruction Set Reference Guide from Texas Instruments (literature number SPRU189B)

  TMS320C6x Software Tools Getting Started Guide from Texas Instruments (literature number SPRU185A)

18.4 DSP Research Workstations

A DSP research lab workstation is created by adding the following equipment to the DSP lab workstation:

❑ **’C2x Research Workstation**

- ■ TMS320C2x XDS/22 emulator
- ■ DSP software library

❑ **’C2xx Research Workstation**

- ■ XDS510™ emulator (PC or Sun/SPARC)
- ■ ’C2xx HLL Debugger (PC or Sun/SPARC)

❑ **’C3x Research Workstation**

- ■ XDS510 emulator (PC or Sun/SPARC)
- ■ ’C3x HLL Debugger (PC or Sun/SPARC)

❑ **’C4x Research Workstation**

- ■ XDS510 emulator (PC or Sun/SPARC)
- ■ ’C4x HLL Debugger (PC or Sun/SPARC)
- ■ ’C4x Parallel Processing Development System (PPDS)

❑ **’C5x Research Workstation**

- ■ XDS510 emulator (PC or Sun/SPARC)
- ■ ’C5x HLL Debugger (PC or Sun/SPARC)

❑ **’C54x Research Workstation**

- ■ XDS510 emulator (PC or Sun/SPARC)
- ■ ’C54x HLL Debugger (PC or Sun/SPARC)

❑ **’C6x Research Workstation**

- ■ XDS510 emulator (PC or Sun/SPARC)
- ■ ’C6x HLL Debugger (PC or Sun/SPARC)

❑ **’C8x Research Workstation**

- ■ XDS510 emulator (Sun/SPARC)
- ■ ’C8x HLL Debugger (PC or Sun/SPARC)
- ■ ’C8x Software Toolkit (Sun/SPARC)

After the DSP lab workstation or DSP research workstation is set up, Texas Instruments provides continued support to the university in the form of suggestions for DSP projects, up-to-date documentation, TMS320 WWW site and BBS, and a hotline. Third-party companies offer special workstation packages and development tools that support the TMS320 digital signal processors.

For more information about the TMS320 University Program and associated pricing, write or e-mail to:

North America
Texas Instruments Incorporated
P.O. Box 1443, M/S 722
Houston, TX 77251–1443
Attention: University Program
e-mail: univ@msg.ti.com
Fax: 281-274-2279
North American University Program Web Site:
http://www.ti.com/sc/ university
Product Information:

TI Product Information Center USA	800-336-5236
TMS320 BBS	281-274-2323
TMS320 Hotline e-mail address	dsph@ti.com
TMS320 Internet BBS	ftp.ti.com
Software Registration/Upgrades	214-638-0333

Asia/Pacific
Texas Instruments Asia Ltd.
24F, Tun Hua S. Rd
Taipei 106, Taiwan ROC
Attn.: University Program
email: tiasia@ti.com
Fax: 886-2-377-5624
Product Information:

International	866-2-23786800	
Domestic	Local Accesss Code	TI Number
Australia	1-800-881-011	800-800-1450
China	10811	800-800-1450
Hong Kong	800-96-1111	800-800-1450
Indonesia	001-801-10	800-800-1450
Korea	080-551-2804	
Malaysia	1-800-800-011	800-800-1450
New Zealand	000-911	800-800-1450
Philippines	105-11	800-800-1450
Singapore	800-01111-111	800-800-1450
Taiwan	080-006800	
Thailand	0019-991-1111	800-800-1450

Japan

Texas Instruments Japan Ltd.
MS-Shibaura Bldg.
4–13–23
Minato-ku,
Tokyo 108-0023 Japan
Attn.: University Program
e-mail: jup@ti.com
Fax: 813-3457-7344

Product Information

Product Information Center in Japan	0120-81-0026
or call 03-3457-0972-or (Intl).	813-3457-0972
Product Information Center in Japan (Fax)	0120-81-0036
or call 03-3457-1259or (intl.)	813-3457-1259

Texas Instruments Japan Homepage:
http://www.tij.co.jp/dsps

India

Texas Instruments (India) Ltd.
DSP Product Development Centre (India),
Golf View Homes
Wind Tunnel Road
Murugeshpalya
Bangalore – 560 017, India
Attention: Sanjeev
Das Mohapatra Manager, University Program
e-mail: sanjeev@india.ti.com
Telephone: 91-80-5269451 Ext 198
91-80-5099198
Fax: 91-80-5298519

Product Information:

Literature Response Center	852-2-956-7288
Literature Response Center Fax	852-2-956-2200
Taiwan DSP BBS	886-2-376-2592
Korea DSP Hotline	82-2-551-2804
Korea DSP Hotline Fax	82-2-5512828
Korea DSP BBS	82-2-551-2914
Hong Kong DSP Hotline	852-2956-7268
Hong Kong DSP Hotline Fax	852-2956-1002
Singapore DSP Hotline Fax	65-390-7179

Mexico

Texas Instruments Mexico
Xola 613, Mod.1-2, col. del Valle,
031000 Mexico DF, Mexico
Attention: University Program
e-mail: s-martini@ti.com
Fax: 525-6399226

Product Information

TI Product Information Center USA	800-336-5236
TMS320 BBS	281-274-2323
TMS320 Hotline	dsph@ti.com
TMS320 Internet BBS	ftp.ti.com
Software Registration/Upgrades	214-638-0333

Europe, Middle East and Africa

Texas Instruments Europe
Product information and multi–lingual support from:
The European Product Information Centre (EPIC)
Attention: The University Programme
8–10 avenue Morane Saulnier–BP 67
78141 Velizy-Villacoublay
Cedex France
e-mail epic@ti.com

Telephone:	(English)	3-130-70-1165
	(French)	33-130-70-64
	(Italian)	33-130-70-1167
	(German)	33-130-70-1168
Fax:		33-130-70-1032
BBS:		33-130-70-199

European University Programme Web Site:
http://www.ti.com/europe/docs/univ/docs/main.htm

Factory Repair and Exchange Instructions

The Microprocessor Development Systems' Factory Repair Center in Houston, Texas (and other locations worldwide), offers warranty repair or exchange at no charge (except shipping) and nonwarranty repair at standard labor and material rates for all current products. You can receive expedited service on exchanges at an additional cost.

Topic Page

A.1 Normal Warranty Exchange or Repair

TI repairs or exchanges products free of charge, provided:

- ❑ You notify Texas Instruments of the problem within ninety (90) days of shipment from Texas Instruments or an authorized distributor.
- ❑ Factory Repair Center's inspection shows that the problem(s) was not caused by accident, alteration, improper installation, improper testing, misuse, neglect, or unauthorized repair.
- ❑ Texas Instruments does not accept responsibility for customer–installed changes, including, but not limited to, customer–generated software in programmable devices. Texas Instruments also reserves the right to refuse to repair and the right to return, at the customer's expense, any product that cannot be tested to its specifications because of the customer's modifications.

Note:

If TI accepts your unit for repair and if you ask for return of the same serial-numbered unit, TI will repair that specific unit. If you do not ask for return of the specific serial-numbered unit, TI reserves the option to repair your returned unit or to exchange it for an equivalent unit.

Exchanged products will be replaced with refurbished units that meet TI workmanship standards for refurbished products.

A.2 Nonwarranty Exchange or Repair

Nonwarranty factory repair or exchange is available on all current and repairable Microprocessor Development Systems products. TI accepts your product as repairable if the cost of repair does not exceed the cost of replacement. Otherwise, you will be notified that the product cannot be classified as repairable and the product will be promptly returned to you.

A.3 System Updates

The standard TI practice is to update each Microprocessor Development Systems product that is returned for repair to the current revision of the make and model. Extra charges may apply to some updates.

A.4 Shipping Instructions

For any factory repair or upgrade, follow these instructions:

1) If you reside within the United States or Canada, contact the Factory Repair Center at (281) 274–2285 and ask for a *Return Material Authorization* (RMA) number.

   If you reside outside of the United States or Canada, contact the nearest RTC or local sales office for instructions.

Note:

TI cannot be responsible for any product returned without prior authorization.

2) Fill out the *Factory Repair and Exchange Questionnaire* card that came with the warranty papers. Provide the following information:

   - The RMA number; note that Texas Instruments will not accept your equipment without this number
   - Your name, contact name (if applicable), and telephone number
   - Purchase order number (if applicable)
   - Proof of date of purchase (required for warranty repair)
   - Model number
   - Serial number
   - *Ship To* information, including address, amount of insurance, and shipping method; note that Texas Instruments ships by UPS or its equivalent and insures for the minimal amount, unless you specify otherwise
   - *Bill To* address
   - If desired, request for return of same serial-numbered unit
   - Description of symptoms—please be as detailed as possible

3) Make a copy of the waybill and the *Factory Repair and Exchange Questionnaire* card for your records in case tracing of your shipment becomes necessary.

4) Pack the unit carefully and securely, preferably with the packing material from its original shipping box. If the original packing material is not available, be sure to use an antistatic packing material where needed to prevent electrostatic discharge damage to board assemblies, components, and target cables. Before sealing, enclose the original copies of the waybill and the *Factory Repair and Exchange Questionnaire.*

5) Return your product (freight prepaid) to the appropriate Factory Repair Center. Within the United States and Canada, send the unit to:

   Texas Instruments Incorporated
   Microprocessor Development Systems
   Factory Repair Center, M/S 730
   12203 Southwest Freeway
   Stafford, Texas 77477

   Outside of the United States and Canada, your local contact will provide shipping instructions.

A.5 Charges and Method of Payment

Most products are repaired on a fixed-price repair basis, provided that the returned product is repairable. Fixed repair prices do not include the cost of repairing items damaged through accident, alteration, improper installation, misuse, neglect, or unauthorized repair. Certain factory upgrades may also incur an extra charge.

You will be notified if the cost of repair exceeds the standard fixed-price rate. You may request return of the same serial-numbered product or may exchange it for a refurbished product.

You may request expedited exchange service at an extra cost, subject to product inventory. The requested product will be shipped within one (1) working day from receipt of the returned product. Product is returned (F.O.B.) to the customer by the Factory Repair Center. Transportation and insurance charges are added to the customer’s invoice.

Appendix B

Program License Agreement

Software programs included with TI Microprocessor Development Systems products are distributed subject to the terms of the license agreement included with the program package, unless a separate written agreement is executed.

A typical TI program license agreement is reproduced on the following page for your reference (note that some Microprocessor Development Systems products include third-party software programs distributed under license by TI, under the terms of the specific agreement packaged with them).

The license terms give you the right to use the program on a single-host computer system. You may move these programs from machine to machine, providing that you do not violate the copyright by making unauthorized copies and/or installing the program on more than one host computer at a time.

Programs on floppy-disk media are typically limited to single-user computer systems. Use of these programs on multiuser host systems requires the payment of additional fees. All other programs may be used on either single- or multiple-user systems, including those with remote log-on capability.

Installation of a program on a server for transmission over a network requires that a network-extension license be obtained (payment of additional fees required) for the program in question.

Some software may be identified as runtime libraries or application software in the user documentation. The terms of the license normally allow you to modify this software and otherwise derive programs from it. When this software is supplied in source-code format, the source-code versions are subject to the terms of the agreement, but the object-code versions are not.

PROGRAM LICENSE AGREEMENT

This document is displayed for you to read prior to using the software and documentation. By using the software and documentation, you agree to abide by the following provisions. If you choose not to agree with these provisions, promptly return the unopened package to the place you obtained it for a refund.

1) **License**—Texas Instruments Incorporated (“TI”), grants you a license to use the software program and documentation in this package (“Licensed Materials”) on a single-user computer in the case of software on floppy disc media or a single computer in the case of all other software. You may not use the Licensed Materials on more than one computer at the same time or otherwise network the Licensed Materials. Use of the Licensed Materials on a network requires payment of additional fees.

2) **Restrictions**—You may not reverse-assemble or reverse-compile the Licensed Materials provided in object code format. You may not sublicense, transfer, assign, rent, or lease the Licensed Materials or this Agreement without written permission from TI.

3) **Copyright**—The Licensed Materials are copyrighted. Accordingly, you may either make one copy of the Licensed Materials for backup and/or archival purposes or copy the Licensed Materials to another medium and keep the original Licensed Materials for backup and/or archival purposes. Additionally, if this package contains multiple versions of the Licensed Materials, then you may only use the Licensed Materials in one version on a single computer. In no event may you use two copies of the Licensed Materials at the same time. You must reproduce the copyright notice on each copy or partial copy of the software.

4) a. **Runtime and Applications Software**—You may create modified or derivative programs of software identified as Runtime Libraries or Applications Software, which in source code form remain subject to this Agreement, but object code versions of such derivative programs are not subject to this Agreement. b. **Operating System and Device Driver Software**—Certain products may contain operating system and device driver software. The license granted is for applications development only. A separate license must be obtained from TI to distribute copies.

5) **Warranty**—TI warrants the media to be free from defects in material and workmanship and that the software will substantially conform to the related documentation for a period of ninety (90) days after the date of your purchase. TI does not warrant that the Licensed Materials will be free from error or will meet your specific requirements.

6) **Remedies**—If you find defects in the media or that the software does not conform to the enclosed documentation, you may return the Licensed Materials along with the purchase receipt, postage prepaid, to the following address within the warranty period and receive a refund.

   Texas Instruments Incorporated
   Microprocessor Development Systems MS 730
   12203 Southwest Freeway
   Stafford, Texas 77477

7) **Limitations**—***TI makes no warranty or condition, either expressed or implied, including but not limited to any implied warranties of merchantability and fitness for a particular purpose, regarding the licensed materials.***

   Neither TI nor any applicable licensor will be liable for any incidental or consequential damages, including but not limited to lost profits.

   Because some states do not allow the exclusion or limitation of incidental or consequential damages or limitation on how long an implied warranty lasts, the above limitations or exclusions may not apply to you.

8) **Export Control** — The re-export of United States origin software and documentation is subject to the Export Administration Act of 1969 as amended. Compliance with such regulations is your responsibility.

2547316-9791 revision B

Appendix C

ROM Codes

This appendix defines the scope of code-customized DSPs and describes the procedures for developing prototype and production units. Information on submitting object code and on ordering customer ROM-coded devices is also included.

Topic **Page**

C.1 Scope

A repetitive routine (for example, boot code) or an entire system algorithm can be embedded (programmed) into the on-chip ROM of a TMS320 DSP. With external memory expansion still available, this reduces the total chip count and allows for more flexibility in program design. Multiple functions are easily implemented by a single device, thus enhancing the system’s capabilities. In many instances, embedded ROM code can reduce the bulk and mechanical size of the end application.

The embedded device, due to its customer-specific code, can only be offered for sale as such to that customer or the customer’s formally designated representative. The customer’s intellectual property (that is, his unique embedded code level) within the device is protected by a unique part number, as well as customer copyright indicated by device symbolization.

Code-customized DSP processors offer these advantages:

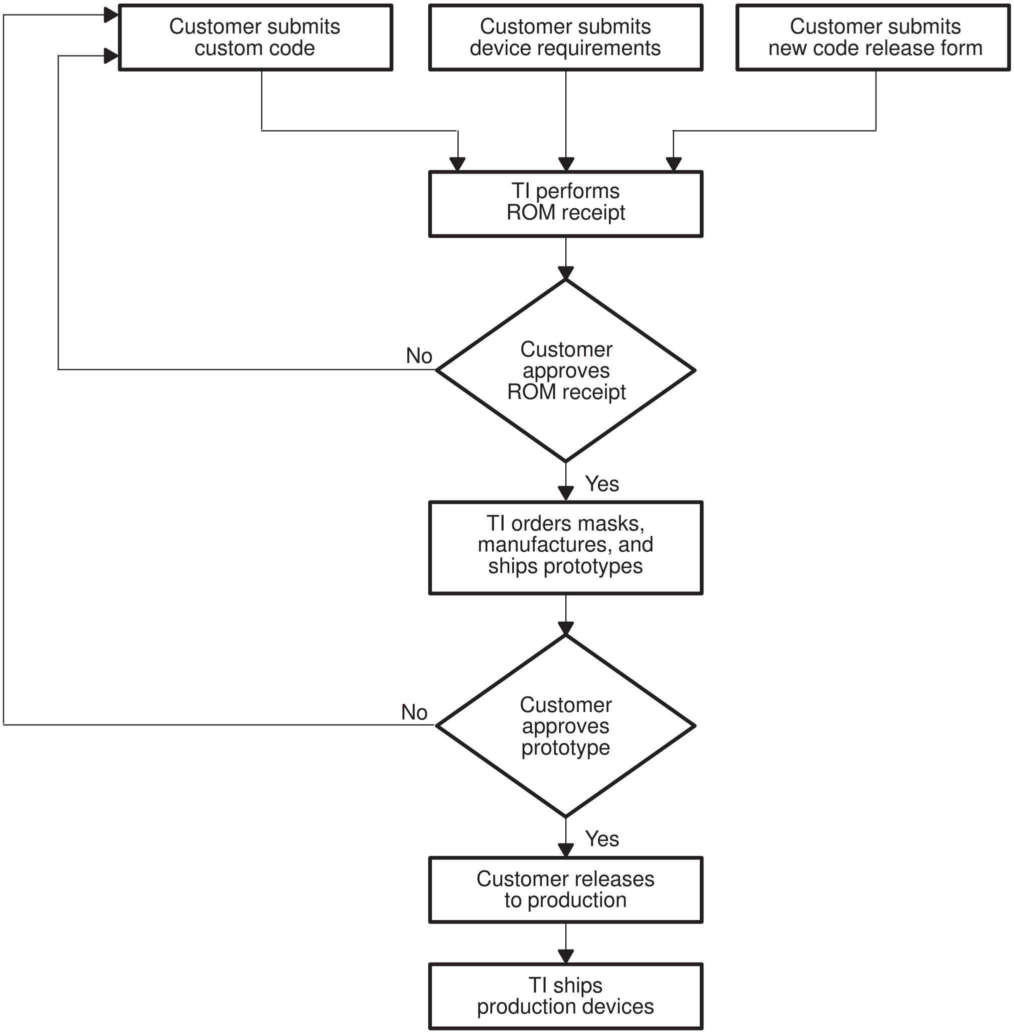
- Lower system cost for volume-driven applications
- Extended system memory expansion capability
- Reduced system hardware and wiring
- More compact/less expensive PCB
- Enhanced security for proprietary software implementations

Standard TMS320 development tools are used to develop, test, refine, and finalize the algorithms. The microprocessor/microcomputer mode (MC/$\overline{\text{MP}}$ for ’C3x; MP/$\overline{\text{MC}}$ for ’C2x, ’C2xx, ’C5x, and ’C54x) is available on all TMS320 DSP devices (with ROM) when on- or off-chip memory access is required. The microprocessor mode is used to develop, test, and refine a system application. In this mode of operation, the TMS320 acts as a standard microprocessor by using external memory only. When the algorithm has been finalized, you may submit the code to Texas Instruments for masking into the on-chip program ROM. At that time, the TMS320 becomes a microcomputer that executes a customized program out of the on-chip ROM. Should the code need changing or upgrading later, the TMS320 may once again be used in the microprocessor mode for development to manage the transition to the revised ROM code. This simplifies the upgrade process by allowing for a “rolling (code) change,” and reduces the possibility of finished and work-in-process inventory obsolescence, while affording an orderly continuation of end-product output.

C.2 Procedure

Figure C–1 illustrates the procedural flow for TMS320 masked parts. When ordering, there is a one-time nonrefundable (NRE) charge for mask tooling and related one-time engineering costs. This charge also covers the costs for a finite number of supplied prototype units. A minimum production order per year is required for any masked-ROM device, and assurance of that order is expected at the time of NRE order acceptance.

Figure C–1. TMS320 ROM Code Prototype and Production Flowchart


Customer submits custom code
Customer submits device requirements
Customer submits new code release form
TI performs ROM receipt
Customer approves ROM receipt
No
Yes
TI orders masks, manufactures, and ships prototypes
Customer approves prototype
No
Yes
Customer releases to production
TI ships production devices

C.2.1 Customer Required Information

For TI to accept the receipt of a customer ROM algorithm, each of the following three items must be received by the TI factory.

1) The customer completes and submits a New Code Release Form (NCRF—available from TI Field Sales Office) describing the custom features of the device (for example, customer information, prototype and production quantities and dates, any exceptions to standard electrical specifications, customer part numbers, and symbolization, package type, etc.). Custom ROM code terms and conditions are also part of the NCRF.
2) If nonstandard specifications are requested on the NCRF, the customer submits a copy of the specification for the DSP in the customer's system, including functional description and electrical specification (including absolute maximum ratings, recommended operating conditions, and timing values).
3) When the customer has completed code development and has verified this code with the development system, the standard TMS320 tagged object code is submitted to the TI factory via any of the following.

   ❑ EPROM devices (currently supported: TMS27C64, TMS27C128, and TMS77C82)

   ❑ MS-DOS formatted disk compatible with IBM PC

   ❑ Electronic ROM transfer: PC-to-PC via Xmodem, Ymodem, Zmodem, or Microsoft's Crosstalk protocol

The completed NCRF, customer specification (if required), and ROM code should be given to the TI Field Sales Office or sent to:

Texas Instruments Digital Signal Processor Products
ATTN: TMS320 DSP Marketing Manager-ROM Receipt, M/S 704
P.O. Box 1443
Houston, Texas 77251–1443

C.2.2 TI Performs ROM Receipt

Code review and ROM receipt is performed on the customer's code and a unique manufacturing ROM code number (such as Dxxxxx) is assigned to the customer's algorithm. All future correspondence should indicate this number. The ROM receipt procedure reads the ROM code information, processes it, reproduces the customer's ROM object code on the same media on which it was received, and returns the processed and the original code to the customer for verification of correct ROM receipt.

C.2.3 Customer Approves ROM Receipt

The customer then verifies that the ROM code received and processed by TI is correct and that no information was misinterpreted in the transfer. The customer must then return written confirmation of correct ROM receipt verification or resubmit the code for processing. This written confirmation of verification constitutes the contractual agreement for creation of the custom mask and manufacture of ROM verification prototype units.

C.2.4 TI Orders Masks, Manufactures, and Ships Prototypes

TI generates the prototype photomasks, processes, manufactures, and tests microcomputer prototypes containing the customer's ROM pattern for shipment to the customer for ROM code verification. These devices have been made using the custom mask but are for the purposes of ROM verification only. For expediency, the prototype devices are tested only at room temperatures (25°C). **Texas Instruments recommends that prototype devices not be used in production systems.** Prototype devices are symbolized with a **P** preceding the manufacturing ROM code number (for example, PDxxxxx) to differentiate them from production devices.

C.2.5 Customer Approves Prototype

The customer verifies the operation of these prototypes in the system and responds with written customer prototype approval or disapproval. This written customer prototype approval constitutes the contractual agreement to initiate volume production using the verified prototype ROM code.

C.2.6 Customer Release to Production

With customer algorithm approval, the ROM code is released to production and TI begins shipment of production devices according to the customer's final specifications and order requirements.

Two lead times are quoted in reference to the preceding flow:

- ❑ **Prototype lead time** is the elapsed time from the receipt of written ROM receipt verification to the delivery of the prototype devices.
- ❑ **Production lead time** is the elapsed time from the receipt of written customer prototype approval to the delivery of production devices. For the latest TMS320 family lead times, contact the nearest TI Field Sales Office.

C.3 Code Submittal

The customer’s object code can be submitted on either 3.5-inch or 5.25-inch disk, EPROM, or via electronic transmittal (that is, modem, Internet, other). For ’C1x or ’C2x family codes, Intel Hex or TI-tagged format is required; for all other families, COFF format from the cross-assembler/linker is needed.

When a code is submitted to Texas Instruments for masking, the code is reformatted by TI to accommodate the TI mask-making and test program generation systems. Application-level verification by the customer is, therefore, necessary. Although the code has been reformatted, it is important that the changes remain transparent to the user and do not affect the execution of the algorithm submitted. Those formatting changes consist essentially of adding ease-of-manufacturing code in reserved and not used (customer) locations only. Resulting code has the code address beginning at the base address of the ROM in the TMS320 device and progressing without gaps to the last address of the ROM on the TMS320 device. Note that because these changes have been made, a checksum comparison is not a valid means of verification. Upon satisfactory verification of the TI returned code, the customer advises TI in writing that it is verified, and this enables release to manufacturing and acceptance of initial orders.

C.4 Ordering

Customer embedded-code devices are user-specified, and thus, each is an unreleased new product until prototype approval and formal release to production. With each initial order of a ROM-coded device, the customer must include written recognition that he understands the following:

The units to be shipped against this order were assembled, for expediency purposes, on a prototype (that is, nonproduction qualified) manufacturing line, the reliability of which is not fully characterized. Therefore, the anticipated reliability of these prototype units cannot be defined.

Sometimes to shorten time to market and upon mutual agreement, the customer may order (and TI will accept) a Risk Production order prior to prototype approval. Under this noncancellable order arrangement, the customer agrees to accept delivery of product containing his code as initially verified and TI agrees to ship to that requirement. The customer is, in effect, agreeing to not change the originally submitted code for the Risk Production order units. He must use the term “Risk Production” in a letter or in a note on the order as a matter of record.

TI does reserve the right to sell excess customer ROM-coded devices as standards to reduce the financial liability incurred through premature ordered quantity reductions or overbuilds. Units thus marketed by TI have all original customer custom symbols or other means of external identification, removed and replaced by a standard product symbol to mask the custom die presence. It is standard practice to require a one-time statement from the customer stating that the customer knows and concurs.

Your local TI Field Sales Office and/or TI Authorized Distributor can be of further assistance on embedded ROM procedure questions and in actually processing your code.

Appendix D

TMS320 PROM Programming

The programmable read-only memory (PROM) versions of the TMS320 family include a 4K-word × 16-bit PROM implemented with an industry-standard PROM cell. These devices can be used for prototyping, early field testing, low-volume production (windowed versions), or high-volume production (one-time programmable versions). The CMOS counterparts of the TMS320 family with a 4K-word masked ROM offer a migration path for cost-effective/higher-volume production. PROM adapter sockets are available that provide 40-pin-to-28-pin conversion for programming the TMS320P15 and TMS320P17. Adapter sockets are also available to provide the 68-pin-to-28-pin conversion required for programming the TMS320P14. Refer to Table 2–13 in Chapter 2, *Selection Guide*, for part numbers.

Key features of the PROM cell include standard programming and verification. The PROM cell also includes a code protection feature that allows code to be protected against copyright violations. The protection feature can be used to prevent the reading of the PROM contents. This appendix describes programming, version verification, and PROM security.

Topic **Page**

D.1 Programming

The TMS320 PROM cell is programmed using the same family and service codes as the TMS27C64 8K × 8-bit EPROM uses. The TMS27C64 EPROM series are ultraviolet-light erasable, electrically programmable read-only memories. They are fabricated using HVCMOS technology. The TMS27C64 is pin compatible with existing 28-pin ROMs and EPROMs.

The TMS320 PROM family, like the TMS27C64, operates from a single 5-V_{DC} supply in the read mode. In the programming mode, an additional 12.5-V_{DC} supply is required. All programming signals are TTL level. Memory locations can be programmed individually, in blocks, or at random. Many of the commercial EPROM programmers can be used for programming outside of the resident system.

In **block programming**, data is loaded into the TMS320 PROM one byte at a time. From the programmer's point of view, data for each memory location is loaded high byte first, low byte second.

CAUTION

The PROM versions of the TMS320 family do not support the **signature** mode available with some EPROM programmers. The signature mode on these programmers places a high voltage (12.5 V_{DC}) on address pin **A9**. The TMS320 PROM cell is not designed for this feature and will be damaged if subjected to this voltage. A 3.9-kΩ resistor is standard on the TI programmer socket between address pin **A9** and the programmer. This protects the device from unintentional use of the signature mode.

Each ROM version of a TMS320 DSP device has a reserved area for TI internal use. When developing a ROM code for release on a TI DSP, do not use this portion of the ROM.

The reserved areas that cannot be used for ROM code development are listed in Table D–1.

Table D–1. TMS320 DSP ROM-Based Devices

Device	On-Chip ROM	No. of Words Reserved for Testing	Address of Reserved ROM
TMS320C5x			
C51	8K	256	1F00 – 1FFF (7936–8191)
C52	4K	256	0F00 – 0FFF (3840–4095)
C53	16K	256	3F00 – 3FFF (16128–16383)
C56	32K	256	7F00 – 7FFF (32512–32767)
C57	32K	256	7F00 – 7FFF (32512–32767)
TMS320C54x			
C541	28K	128	FF00 – FF7Fh (65280–65407)
C545	48K	128	FF00 – FF7Fh (65280–65407)
C546	48K	128	FF00 – FF7Fh (65280–65407)

D.2 Fast and SNAP! Pulse Programming

Two programming algorithms are available for TMS320 PROM devices. The *fast programming* algorithm is normally used to program the entire PROM contents, although individual locations may be programmed separately. Fast programming is supported on the TMS320P14, TMS320P15, and TMS320P17. The other TI algorithm, *SNAP! pulse programming*, can reduce the programming time to a nominal duration of one second. Note that actual programming time varies as a function of the programmer being used. SNAP! pulse programming is supported on the TMS320P14. For more information on these two programming algorithms, consult the appropriate TMS320 user's guide.

D.3 Version Verification

Information on verification of the PROM versions of the TMS320 family is contained in the appropriate TMS320 user's guide.

D.4 PROM Security

The PROM protection mechanism completely disables a TMS320 device and prevents reading of the PROM contents. This guarantees the security of proprietary algorithms. This facility is implemented through a unique PROM cell called the ROM-protect bit (RBIT) cell. Once the contents are programmed into the PROM, the RBIT can be programmed, preventing access to the PROM contents and disabling the microprocessor mode on the TMS320 device. Once programmed, the RBIT can be cleared only by erasing the entire EPROM array with ultraviolet light, thus maintaining security of the proprietary algorithm. Information on programming and verification of the RBIT is available in the appropriate TMS320 user's guide.

Appendix E

Glossary

A

A0–A15: External address pins for data/program memory or I/O devices.

ACC: See *accumulator (ACC)*.

accumulator (ACC): A 32-bit register that stores the results of an arithmetic logic unit (ALU) operation and provides an input for subsequent ALU operations. The ACC is accessible in two halves: accumulator high (ACCH) and accumulator low (ACCL).

active window: The window that is currently selected for moving, sizing, editing, closing, or some other function.

ADC: See analog-to-digital converter.

address: The logical location of program code or data stored in memory.

addressing mode: The method by which an instruction calculates the location of its required data.

ADTR: *Asynchronous data transmit and receive register*. An 8-bit register in the asynchronous serial port that writes the data to transmit and reads the data received. See also *ARSR*.

aliasing: A method of customizing debugger commands; aliasing provides a shorthand method for entering often-used command strings.

ALU: See *arithmetic logic unit*.

analog-to-digital converter (ADC): A successive-approximation converter with internal sample-and-hold circuitry used to translate an analog signal to a digital signal.

ANSI C: A version of the C programming language that conforms to the C standards defined by the *American National Standards Institute.*

AR0–AR7: *Auxiliary Registers 0–7*. Eight 16-bit registers that are used as pointers to an address within the data space address range. The registers are operated on by the auxiliary register arithmetic unit (ARAU) and are selected by the auxiliary register pointer (ARP).

ARAU: See auxiliary-register arithmetic unit.

ARB: See *auxiliary register pointer buffer*. A 3-bit field in status register ST1 that holds the previous value of the auxiliary register pointer (ARP).

archive library: A collection of individual files that have been grouped into a single file.

archiver: A software program that allows you to collect several individual files into a single file called an archive library. The archiver also allows you to delete, extract, or replace members of the archive library, as well as to add new members.

arithmetic logic unit (ALU): The part of the CPU that performs arithmetic and logic operations.

ARP: *Auxiliary register pointer*. A 3-bit field in status register ST1 used as a pointer to the currently selected auxiliary register.

ARSR: *Asynchronous receive shift register*. An 8-bit register in the asynchronous serial port that receives data from the TX pin one bit at a time and, when ARSR is full, it transfers its data to the ADTR. See also *ADTR.*

ASCII: American Standard Code for Information Interchange. A standard computer code for representing and exchanging alphanumeric information.

assembler: A software program that creates a machine-language program from a source file that contains assembly language instructions, directives, and macro directives. The assembler substitutes absolute operation codes for symbolic operation codes, and absolute or relocatable addresses for symbolic addresses.

assembly mode: A debugging mode that shows assembly language code in the DISASSEMBLY and doesn’t show the FILE window, no matter what type of code is currently running.

asynchronous data transmit and receive register (ADTR): An 8-bit register in the asynchronous serial port that writes the data to transmit and reads the data received. See also *ARSR*.

asynchronous serial port control register (ASPCR): A 16-bit register in the asynchronous serial port that controls signals IO0–IO3, breaks, stop bits, auto baud alignment, reset, and emulation mode.

asynchronous transmit shift register (AXSR): An 8-bit register in the asynchronous serial port that receives data from the ADTR and transfers it one bit at a time to the TX pin. See also *ADTR*.

autoinitialization: The process of initializing global C variables (contained in the .cinit section) before beginning program execution.

auxiliary-register arithmetic unit (ARAU): An arithmetic logic unit (ALU) used to calculate indirect addresses using the auxiliary registers as inputs and outputs.

auxiliary register (AR): Eight 16-bit memory-mapped registers (AR0–AR7) that are used for indirect data address pointers, temporary storage, or integer arithmetic processing through the auxiliary register arithmetic unit (ARAU). Each AR is selected by the auxiliary register pointer (ARP).

auxiliary register pointer (ARP): A field in the status register used as a pointer to the currently selected auxiliary register.

auxiliary register pointer buffer (ARB): A field in the status register that holds the previous value of the auxiliary register pointer (ARP).

auxiliary registers: Registers that are used as pointers to an address within the data-space address range. The registers are operated on by the auxiliary register arithmetic unit (ARAU) and are selected by the auxiliary register pointer (ARP).

AXSR: *Asynchronous transmit shift register*. A register in the asynchronous serial port that receives data from the ADTR and transfers it one bit at a time to the TX pin. See also *ADTR*.

B

benchmark: A test of performance of a computer or peripheral device.

bit-reversed addressing: Addressing in which several bits of an address are reversed in order to speed processing of algorithms, such as Fourier transforms.

bit-reversed index addressing: A method of indirect addressing that allows efficient I/O operations by resequencing the data points in a radix-2 FFT program. The direction of carry propagation in the ARAU is reversed.

boot: The process of loading a program into program memory.

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

BOOT port: A port that enables the boot loader. When BOOT is held low, the processor executes the boot loader program after a hardware reset. When BOOT is held high. the processor skips execution of the boot loader.

breakpoint: A point within a program where execution halts because of a previous request.

BRI: Basic rate service of ISDN, providing two B channels and one 16-Kbps D channel.

BSP: See *buffered serial port (BSP)*.

buffered serial port (BSP): An on-chip module that consists of a full-duplex, double-buffered serial port interface and an autobuffering unit (ABU).

burst mode: A synchronous serial port mode in which a single word is transmitted following a frame synchronization pulse (FSX and FSR).

byte: A sequence of eight adjacent bits operated upon as a unit.

C

C: A high-level, general-purpose programming language useful for writing compilers and operating systems and for programming microprocessors.

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

CAD: Computer-aided design.

CALU: *Central arithmetic logic unit*.

CAM: Computer-aided manufacturing.

carry bit: A bit in the status register ST1 used by the ALU for extended arithmetic operations and accumulator shifts and rotates. The carry bit can be tested by conditional instructions.

central arithmetic logic unit (CALU): The 32-bit wide, main arithmetic logic unit that performs arithmetic and logic operations.

central processing unit (CPU): The module that controls and interprets the machine-language program and its execution.

circular addressing: An addressing mode in which an auxiliary register is used to cycle through a range of addresses to create a circular buffer in memory.

CLK register: Bit 0 of this register turns the CLKOUT1 signal on or off.

CLKIN: *Input clock signal*.

CLKR: *Receive clock input pin*. A pin that receives an external clock signal to clock data from the DR pin in to the serial port receive shift register.

clock modes: Options used by the clock generator to change the internal CPU clock frequency to a fraction or multiple of the frequency of the input clock signal.

code: A set of instructions written to perform a task.

COFF: *Common Object File Format*. An implementation of the object file format of the same name developed by AT&T. The TMS320 fixed–point DSP compiler, assembler, and linker use and generate COFF files.

command file: A file that contains options, filenames, directives, or commands for the linker or hex conversion utility.

comment: A source statement (or portion of a source statement) that is used to document or improve readability of a source file. Comments are not compiled, assembled, or linked; they have no effect on the object file.

common object file format (COFF): A binary object file format that promotes modular programming by supporting the concept of *sections. See also COFF.*

configuration control bit (CNF): A bit in status register ST1 used to determine whether the on-chip RAM block B0 is mapped to program space or data space.

constant: A numeric value that can be used as an operand.

continuous mode: A synchronous serial port mode in which only one frame synchronization pulse is necessary to transmit several packets at maximum frequency.

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

CPU cycle: The time it takes the CPU to go through one logic phase (during which internal values are changed) and one latch phase (during which the values are held constant).

current auxiliary register: The auxiliary register pointed to by the auxiliary register pointer (ARP).

D

D0–D31: External data-bus pins that transfer data between the processor and external data/program memory or I/O devices. *See also LD0–LD31*.

DAB: See *direct-address bus (DAB)*.

DARAM: *Dual-access RAM*. Memory that can be accessed twice in a single clock cycle.

data-address generation logic: Logic circuitry that generates the addresses for data-memory reads and writes. This circuitry can generate one address per machine cycle. See also *program-address generation logic.*

data bus: A group of connections used to route data.

data-display windows: Windows for observing and modifying various types of data. This category includes the MEMORY, CPU, DISP, and WATCH windows.

data memory: A memory region used for storing and manipulating data.

data-page pointer (DP): A field in the status register that specifies what is currently selected for direct address generation.

data-read address bus: A 16–bit bus that provides the address for data read operations and is driven by the ’C2cLP CPU.

data-write bus: A 16-bit bus that provides data for a data memory write and is driven by the ’C2xLP CPU.

data-write address bus: A 16-bit bus that provides the address for data write operations and is driven by the ’C2xLP CPU.

decode phase: The phase of the pipeline in which the instruction is decoded (identified).

debugger: A window-oriented software interface that helps to debug ’C2xx programs running on a ’C2xx emulator or simulator.

DIE: See DMA interrupt-enable register.

DIM: *Delta-interrupt mask bit*. Bit 9 of the ASPCR, which enables and disables delta-detect. The delta-detect function allows or prevents interrupts from being generated by changes on the I/O pins.

direct address bus (DAB): A 16-bit bus that provides the data address used by the central processing unit (CPU).

direct memory access (DMA): A mode where a device other than the host processor contends for, and receives, mastership of the memory bus so that data transfers may take place independent of the host.

directives: Special-purpose commands that control the actions and functions of a software tool (as opposed to assembly language instructions, which control the actions of a device).

disassembly window: A window that displays the disassembly of memory contents.

DMA: See *direct memory access (DMA)*.

DMA coprocessor: A peripheral that transfers the contents of memory locations independently of the processor (except for initialization).

DMA-interrupt enable register (DIE): A register (in the CPU register file) that controls to which interrupts the DMA coprocessor responds.

DP: *Data-page pointer*. A field in the status register that specifies what is currently selected for direct address generation

DR bit: *Data ready indicator for the receiver*. A bit that is automatically reset to zero when the receive register is read, or when the device is reset.

DR pin: *Serial-data-receive pin*. A pin that receives serial data into the serial port receive shift register.

DRAB: *Data-read address bus*. A 16-bit bus that provides data for a data-memory read and is driven by the memories or the logic interface.

DRAM: Dynamic random-access memory.

dual-access RAM (DARAM): Memory space that can be read from and written to in the same clock cycle.

DWAB: *Data-write address bus*. A 16-bit bus that provides the address for data-write operations and is driven by the ’C2xLP CPU.

DWEB: *Data-write bus*. A 16-bit bus that provides data for a data-memory write and is driven by the ’C2xLP CPU.

E

EMI: Electromagnetic interference.

emulator: A device that is built to work like another.

emurst: A utility that reads the emulator.

EVM: Evaluation module.

EVRC: Enhanced variable-rate coder.

expression: A statement that describes data and processing as part of a program. The statement usually includes a constant, a symbol, or a series of constants and symbols separated by arithmetic operators.

extended-precision floating-point format: A 40-bit representation of a floating-point number with a 32-bit mantissa and an 8-bit exponent.

extended-precision register: A 40-bit register used primarily for extended-precision floating-point calculations. Floating-point operations use bits 39–0 of an extended-precision register. Integer operations, however, use only bits 31–0.

external interrupt: A hardware interrupt triggered by a pin.

external symbol: A symbol that is used in the current program module but is defined in a different program module.

F

fast Fourier transform (FFT): An efficient method of computing the discrete Fourier transform, which transforms functions between the time domain and frequency domain. The time-to-frequency domain is called the forward transform, and the frequency-to-time domain is called the inverse transformation.

FFT: See *fast Fourier transform (FFT).*

FIFO buffer: *First-in, first-out buffer.* A portion of memory in which data is stored and then retrieved in the same order in which it was stored. Thus, the first word stored in this buffer is retrieved first.

FIR: Finite impulse response.

FMV: Full motion video.

framing error: An error that occurs when a data character received by the asynchronous serial port does not have a valid stop bit.

G

general-purpose input/output pins: Ports that can be used to supply input signals from an external device or output signals to an external device. These pins are not linked to specific uses; rather, they provide input or output signals for a variety of purposes.

global-data memory space: One of four memory spaces. The global-data memory space can either share data with other processors within the system or serve as additional data-memory space.

global-memory allocation register (GREG): An 8-bit memory-mapped register that specifies the size of the global memory space. At reset, the GREG is cleared.

GP: General purpose.

GREG: See *global memory allocation register (GREG)*.

GSM: Global System for Mobile Communications.

H

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

hex conversion utility: A program which accepts COFF files and converts them into one of several standard ASCII hexadecimal formats suitable for loading into an EPROM programmer.

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

high-level language debugging: The ability of a compiler to retain symbolic and high-level language information (such as type and function definitions) so that a debugging tool can use this information.

host-port interface (HPI): An on-chip module consisting of an 8-bit parallel port that interfaces a host processor to the TMS320C57. The HPI has two modes of operation, shared-access mode (SAM) and host-only mode (HOM). Status and control of the HPI is specified in the HPI control register (HPIC). See also *shared-access mode (SAM)* and *host-only mode (HOM)*.

HPI: See *host port interface (HPI)*.

I

I/O switches: Hardware switches on the emulator or EVM board that identify the PC I/O memory space used for emulator–debugger or EVM-debugger communications.

$\overline{\textbf{IACK}}$: *Interrupt acknowledge signal*. An output signal that indicates that an interrupt has been received and that the program counter is fetching the interrupt vector that will force the processor into an interrupt service routine.

IFR: See *interrupt flag register (IFR)*.

IFR: *Interrupt flag register*. A 16–bit memory-mapped register used to flag pending interrupts. The IFT may be read from for identifying pending interrupts and written to for clearing selected interrupts.

IIE: See internal-interrupt-enable register.

IIF: See IIOF flag register.

IIOF flag register (IIF): Controls the function (general-purpose I/O or interrupt) of the four external pins ($\overline{\text{IIOF0}}$ to $\overline{\text{IIOF3}}$). It also contains timer/DMA interrupt flags.

IMR: *Interrupt mask register*. A 16-bit memory-mapped register used to mask external and internal interrupts. you can read from or write to the IMR. A 1 written to any IMR bit position enables the corresponding interrupt (when INTM=0).

index registers: Two registers (IR0 and IR1) that are used by the ARAU for indexing an address.

Input clock signal (CLKIN): A clock source signal supplied to the on-chip clock generator via the CLKIN/X2 pin.

input data-scaling shifter: A 16- to 32-bit barrel left shifter that shifts incoming 16-bit data from 0 to 16 positions left relative to the 32-bit output within the fetch cycle with no cycle overhead.

input/output status register (IOSR): A register in the asynchronous serial port that provides status information about signals IO0–IO3 and about transfers in progress.

instruction register (IR): A 16-bit register that contains the instruction being executed.

instruction: The basic unit of programming that causes the execution of one operation; it consists of an opcode and operands along with optional labels and comments.

internal interrupt: A hardware interrupt caused by an on-chip peripheral.

internal interrupt-enable register: A register (in the CPU register file) that determines whether or not the CPU will respond to interrupts from the communication ports, the timers, and the DMA coprocessor.

interrupt: A signal sent to the CPU that (when not masked) forces the CPU into a subroutine called an interrupt service routine. This signal can be triggered by an external device, an on-chip peripheral, or an instruction (INTR, NMI, or TRAP).

interrupt acknowledge ($\overline{\text{IACK}}$): A signal that indicates that an interrupt has been received, and that the program counter is fetching the interrupt vector location.

interrupt-flag register (IFR): A 16-bit memory-mapped register used to flag several hardware and software interrupts. The IFR may be read to identify pending interrupts and written to clear selected interrupts.

interrupt mask register (IMR): A 16-bit memory-mapped register used to mask several external and internal interrupts. You can read from the IMR to determine which interrupts are masked/unmasked. A write to any IMR bit enables a corresponding interrupt (when INTM = 0).

interrupt service routine (ISR): A module of code that is executed in response to a hardware or software interrupt.

IR: *Instruction register*. A 16-bit register that contains the instruction being executed.

ISA: *Industry Standard Architecture*. A subset of the EISA standard.

ISR: *Interrupt service routine*. A module of code that is executed in response to a hardware or software interrupt.

J

JTAG: *Joint Test Action Group.* A group of 200 members that designed a testability standard sanctioned by IEEE (IEEE Standard 1149.1).

L

LA0–LA30: External address pin*s* for data/program memory or I/O devices. These pins are on the local bus. *See also A0–A30*.

LD0–LD31: External data bus pins that transfer data between the processor and external data/program memory or I/O devices. *See also D0–D31*.

linker: A software tool that combines object files to form an object module that can be allocated into TMS320C6200 system memory and executed by the device.

listing file: An output file, created by the assembler, that lists source statements, their line numbers, and their effects on the SPC.

M

MAC: Multiply and accumulate.

macro: A user-defined routine that can be used as an instruction.

mantissa: A component of a floating-point number consisting of a fraction and a sign bit. The mantissa represents a normalized fraction whose binary point is shifted by the exponent.

map file: An output file, created by the linker, that shows the memory configuration, section composition, and section allocation, as well as symbols and the addresses at which they were defined.

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

master clock output signal (CLKOUT1): The output signal of the on-chip clock generator. The CLKOUT1 high pulse signifies the CPU's logic phrase (when internal values are changed), while the CLKOUT1 low pulse signifies the CPU's latch phase (when the values are held constant).

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

memory-mapped register: One of the on-chip registers mapped to addresses in memory. Some memory-mapped registers are mapped to data memory, and some are mapped to input/output memory.

MFLOPS: *Million floating-point operations per second*. A measure of floating-point processor speed that counts of the number of floating-point operations made per second.

microcomputer mode: A mode in which the on-chip ROM (boot loader) is enabled. This mode is selected via the MP/$\overline{\text{MC}}$ pin.

microprocessor mode: A mode in which the on-chip ROM is disabled. This mode is selected via the MP/$\overline{\text{MC}}$ pin.

MIPS: Million instructions-per-second.

multiplier: A device that generates the product of two numbers.

N

next auxiliary register: The register that will be pointed to by the auxiliary register pointer (ARP) when an instruction that modifies ARP is finished executing.

nonmaskable interrupt (NMI): A hardware interrupt that uses the same logic as the maskable interrupts but cannot be masked.

NTSC: National Television Standards Committee.

O

object file: A file that has been assembled or linked and contains machine-language object code.

off-chip: A device external to the device.

on-chip: An element or module of the device.

opcode: *operation code.* In most cases, the first byte of the machine code that describes the type of operation and combination of operands to the central processing unit (CPU).

operand: The part of an instruction that designates where the central processing unit (CPU) will fetch or store data. The operand consists of the arguments, or parameters, of an assembly language instruction, assembler directive, or macro directive.

operands: The arguments, or parameters, of an assembly language instruction, assembler directive, or macro directive.

options: Command parameters that allow you to request additional or specific functions when you invoke a software tool.

output data-scaling shifter: 32- to 16- bit barrel left shifter. Shifts the 32-bit accumulator output from 0 to 7 bits left for quantization management, and outputs either the 16-bit high or low half of the shifted 32–bit data to the data write bus.

output module: A linked, executable object file that can be downloaded and executed on a target system.

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

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

P

PAB: *Program-address bus*. A 16-bit bus that provides the address for program memory reads and writes.

PAR: *Program-address register*. A register that holds the address currently being driven on the program address bus for as many cycles as it takes to complete all memory operations scheduled for the current machine cycle.

parallel logic unit (PLU): A 16-bit logic unit that executes logic operations from either long immediate operands or the contents of the dynamic bit manipulation register (DBMR) directly upon data locations without affecting the contents of the accumulator (ACC) or product register (PREG).

PC: Personal computer or program counter, depending on the context and use in this book: 1) In installation instructions or information relating to hardware and boards, *PC* means *personal computer* (as in IBM PC). 2) In general debugger and program-related information, *PC* means *program counter*, which is the register that identifies the current statement in a program.

PDM: *Parallel debug manager*. A program used for creating and controlling multiple debuggers for the purpose of debugging code in a parallel-processing environment.

peripheral bus: A bus that is used by the CPU to communicate the DMA coprocessor, communication ports, and timers.

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

pipelining: A design technique for reducing the effective propagation delay per instruction operation by partitioning the operation into a series of four independent stages, each of which performs a portion of the operation.

PLL: Phase-locked loop.

PLU: See *parallel logic unit (PLU)*.

pma: *Program memory address*. A register that provides the address of a multiplier operand that is contained in program memory.

port address: The PC I/O memory space that the debugger uses for communicating with the emulator or EVM. The port address is selected via switches on the emulator or EVM board and communicated to the debugger with the –p debugger option.

PRD: *Timer period register*. A 16-bit memory-mapped register that specifies the period for the on-chip timer. When the timer counter register (TIM) is decremented past zero, the TIM is loaded with the value in the PRD.

PRDB: *Program-read data bus*. A 16-bit bus that provides data for program memory reads and is driven by the memories or the logic interface.

PREG: *Product register*. A 32-bit register that holds the results of a multiply operation. See also *product register (PREG).*

PRESCALER: *prescaling shifter*. A 0- to 16-bit left barrel shifter used to prescale data coming into the arithmetic logic unit (ALU). This shifter is also used as a 0- to 16-bit right barrel shifter of the accumulator (ACC). The shift count is specified by a constant in the instruction or by the value in temporary register 1 (TREG1).

product register (PREG): A 32-bit register that holds the output from the multiplier. The high and low words of the PREG can be accessed individually. See also *multiplier (MULT)*.

product-scaling shifter: A 32-bit shifter that performs a 0, 1, or 4-bit left shift, or a 6-bit right shift of the multiplier product. The left-shift options are used to manage the additional sign bits resulting from the 2s-complement multiply. The right-shift option is used to scale down the number to manage the overflow of product accumulation in the CALU.

product shift mode: One of four modes (–0, –1, –4, or 61–bit) used by the product-scaling shifter.

program address bus: A 16-bit bus that provides the address for program memory reads and writes.

program address generation logic: Logic circuitry that generates the address for program memory reads and writes, and the address for data memory in instructions that require two data operands. This circuitry can generate one address per machine. See also *data address generation logic*.

program controller: Logic circuitry that decodes instructions, manages the pipeline, stores status of operations, and decodes conditional operations.

program counter: A register that contains the address of the next instruction to be executed.

program-read data bus (PRDB): A 16-bit bus that provides data for program memory reads and is driven by the memories or the logic interface.

P-SCALER: *Product shifter*. A 0-, 1-, or 4-bit left shifter that removes extra signed bits (gained in the multiply operation) when fixed-point arithmetic is used; or a 6-bit right shifter that scales the products down to avoid overflow in the accumulation process. The shift mode is specified by the product shift mode (PM) bits.

pulse code modulation (PCM): A technique for digitizing speech by sampling the sound waves and converting each sample into a binary number.

R

READY: *Data ready input*. A memory control signal indicating that an external device is prepared for a bus transaction to be completed.

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

repeat counter (RPTC): A 16-bit register that counts the number of times a single instruction is executed.

repeat counter register: A register (in the CPU register file) that specifies the number of times minus one that a block of code is to be repeated when a block repeat is performed.

repeat mode: A zero-overhead method for repeating the execution of a block of code.

reset: A means to bring the central processing unit (CPU) to a known state by setting the registers and control bits to predetermined values and signaling execution to start at a specified address.

RSR: *Receive shift register*. One of two registers that perform shift operations in and out of the serial port. The other register is the transmit shift register (XSR).

run address: The address where a section runs.

S

SAM: *Shared-access mode.*

SARAM: S*ingle-access RAM*. Memory thatcan be read from or written to only in a single clock cycle.

scalar type: A C type in which the variable in a single variable, not composed of other variables.

SDTR: *Synchronous serial port transmit/receive register*. The SDTR is an I/O-mapped read/write register that sends data to the transmit FIFO buffer, and extracts data from the receive FIFO buffer.

serial-port-control register (SPC): A 16-bit memory-mapped register that contains status and control bits for the serial-port interface. The SPC is identical to the time-division multiplexed serial-port control register (TSPC), except that bit 0 is reserved for the TDM bit.

serial-port interface: An on-chip full-duplex serial port interface that provides direct serial communication to serial devices with a minimum of external hardware, such as codecs and serial analog-to-digital (A/D) converters. Status and control of the serial port is specified in the serial port control register (SPC).

simulator: A development tool that simulates the operation of the device for executing and debugging applications programs by using the device debugger.

single-access RAM (SARAM): Memory space that only can be read from or written to in a single clock cycle.

single-precision floating-point format: A 32-bit representation of a floating-point number with a 24-bit mantissa and an 8-bit exponent.

single-step: A form of program execution that allows you to see the effects of each statement. The program is executed statement by statement; the debugger pauses after each statement to update the data-display windows.

software interrupt: An interrupt caused by the execution of an INTR, NMI, or TRAP instruction.

source file: A file that contains C code or assembly language code that will be compiled or assembled to form an object file.

ST: See status register.

ST0/ST1: Two 16-bit registers that contain status and control bits.

stack: A block of memory used for storing return addresses for subroutines and interrupt service routines or for storing data. The ’C2xLP stack is 16 bits wide and eight levels deep.

status register: A register in the CPU register file that contains global information related to the CPU.

string table: A table that stores symbol names that are longer than eight characters (symbol names of eight characters or longer cannot be stored in the symbol table; instead, they are stored in the string table). The name portion of the symbol’s entry points to the location of the string in the string table.

symbol: A string of alphanumeric characters that represents an address or a value.

symbolic debugging: The ability of a software tool to retain symbolic information so that it can be used by a debugging tool such as a simulator or an emulator.

symbol table: A portion of a COFF object file that contains information about the symbols that are defined and used by the file.

synchronous serial port control register (SSPCR): The 16-bit memory-mapped SSPCR controls the operation of the synchronous serial port.

synchronous serial port transmit/receive register (SDTR): *Transmit interrupt bit*. This maskable bit (bit 4) of the interrupt mask register is tied to a transmit interrupt for the synchronous serial port.

synchronous serial port receive interrupt: *Receive-interrupt pin*. Bit 3 of the interrupt mask register is tied to the receive interrupt for the synchronous serial port.

T

T320C2xLP: *Texas Instruments cDSP core.*

TCR: *Timer control register*. A 16-bit register that controls the operation of the on-chip timer.

TEC: *Texas Instruments embedded gate array*. A gate array embedded with a hardware macro.

TGC: *Texas Instruments gate array.*

temporary register (TREG): A 16-bit register that holds a temporary data value. See also *TREG0*, *TREG1*, and *TREG2*.

time-division multiplexing (TDM): The process by which a single serial bus is shared by devices with each device taking turns to communicate on the bus.

timer: A programmable peripheral that can generate pulses or time events.

timer-control register (TCR): A 16-bit register that controls the operation of the on-chip timer.

timer-period register: A 32-bit memory-mapped register that specifies the period for the on-chip timer.

transmit shift register (XSR): One of two registers that perform shift operations in and out of the serial port. The other register is the receive shift register (RSR).

TREG: *Temporary register*. A register that holds a temporary data value.

TREG0: *temporary register 0*. A 16-bit memory-mapped register that holds an operand for the multiplier. See also *multiplier (MULT)*.

TREG1: *temporary register 1*. A 5-bit memory-mapped register that holds a dynamic prescaling shift count for data inputs to the arithmetic logic unit (ALU). See also *PRESCALER*.

TREG2: *temporary register 2*. A 4-bit memory-mapped register that holds a dynamic bit pointer for the BITT instruction.

U

UART: *Universal asynchronous receiver/transmitter*. Used as another name for the asynchronous serial port.

URST: *Reset asychronous serial port bit*. Bit 13 of the ASPCR resets the asynchronous port.

V

VGA: *Video Graphics Array*. An industry standard for video cards.

W

wait state: A period of time that the CPU must wait for external program, data, or I/O memory to respond when reading from or writing to that external memory. The CPU waits one extra cycle for every wait state.

wait-state generator: A program that can be modified to generate a limited number of wait states for a given off-chip memory space (lower program, upper program, data, or I/O).

wait-state generator-control register (WSGR): This register, which is mapped to I/O memory, controls the wait-state generator.

WATCH window: A window that displays the values of selected expressions, symbols, addresses, and registers.

WD: Watchdog.

word: A 16-bit addressable location in target memory.

word: A word, as defined in this document, consists of a sequence of 16 adjacent bits (two bytes).

WSGR: *Wait-state generator-control register*. This register, which is mapped to I/O memory, control the wait-state generator.

WWW: World Wide Web

X

XDS: Extended development system

Appendix F

Acronyms and Abbreviations

A

ACC: accumulator

A/D: analog-to-digital

ADC: analog-to-digital converter

ADPCM: adaptive differential pulse code modulation

ADTR: asynchronous data transmit and receive register

AIB: analog interface board

AIC: analog interface circuit

ALU: arithmetic logic unit

APLL: ASIC PLL

AR: auxiliary register

ARAM: audio random-access memory

ARAU: auxiliary register arithmetic unit

ARB: auxiliary register pointer buffer

ARP: auxiliary register pointer

ARSR: asynchronous receive shift register

ASCII: American standard code for information interchange

ASIC: application-specific integrated circuit

ASPCR: asynchronous serial port control register

ATM: asynchronous transfer mode

AXSR: asynchronous transmit shift register

B

BGA: ball grid array

BOPS: billion operations per second

BRI: basic rate service (of ISON)

BSP: buffered serial port

BSR: bank-select register

BTT: breakpoint, trace, and time-stamping

C

CAD: computer-aided design

CALU: central arithmetic logic unit

CAM: computer-aided manufacturing

CBCR: circular buffer control register

CD: computer disk

cDSP: customizable digital signal processor

CELP: code excited linear prediction

codec: coders and decoders

COFF: common object file format

CPGA: ceramic pin grid array

CPU: central processing unit

CQFP: ceramic quad flat pack

CRC: cyclic redundancy check

CSSU: compare, select, store unit (Viterbi operator)

D

D/A: digital-to-analog

DAB: direct-address bus

DAC: digital-to-analog converter

DASP: Dallas Application-Specific Products

DMOV: data move

DARAM: dual-access RAM

DBMR: dynamic bit-manipulation register

DIE: DMA interrupt-enable register

DIM: delta-interrupt mask bit

DIP: dual in-line package

DMA: direct memory access

DP: data-page pointer

DRAB: data-read address bus

DRAM: dynamic random-access memory

DSI: digital speech interpolation

DSK: DSP starter kit

DSP: digital signal processor

DTAD: digital tapeless answering device

DTMF: dual-tone multi-frequency (”touch tones”)

DWAB: data-write-address bus

DWEB: data-write bus

E

EEPROM: electrically-erasable programmable read-only memory

EMI: electromagnetic interference

EPK: emulator porting kit

EPROM: erasable programmable read-only memory

EV: event manager

EVM: evaluation module

EVRC: enhanced variable-rate coder

F

FIFO: first-in, first-out

FIR: finite impulse response

FMV: full-motion video

FFT: fast Fourier transform

G

GEL: GO DSP Corporation's extension language

GP: general purpose

GPT CON: general-purpose timer control register

GREG: global memory allocation register

GSM: global system for mobile communications

H

HDD: hard disk drive

HLL: high-level language

HOM: host-only mode

HPI: host-port interface

I

ICM: incoming message

IDE: integrated development environment

IFR: interrupt-flag register

I/O: input/output

IIE: internal interrupt enable

IIR: infinite impulse response

IMR: interrupt-mark register

IR: instruction register

ISA: industry-standard architecture

ISDN: Integrated Services Digital Network

ISO: International Standards Organization

ISR: interrupt service routine

J

JLCC: J-leaded chip carrier

JPEG: Joint Photographic Experts Group

JTAG: Joint Test Action Group (IEEE Standard 1149.1)

L

LAN: local area network

LCCC: leadless ceramic chip carrier

M

MAC: multiply and accumulate

MACD: multiply, accumulate, and data move

MCU: microcontroller unit

MFLOPS: million floating-point operations per second

MIPS: million instructions per second

MP: master processor

MPEG: Moving Picture Expert Group

MPY: multiply

MUX: multiplexor

MVP: multimedia video processor

N

NCRF: New Code Release Form

NRZ: Non-return to zero

NTSC: National Television Standards Committee

O

OCR: optical character recognition

OGM: out-going message

OTP: one-time programmable

P

PAB: program-address bus

PAL: phase alternate line

PAR: program-address register

PCB: printed circuit board

PC: personal computer

PCI: peripheral component interconnect

PCM: pulse code modulation

PCMCIA: Personal Computer Memory Card International Association

PCS: personal communications system

PDA: personal digital assistant

PDC: personal digital cellular

PDM: parallel debug manager

PGA: pin grid array

PIC: Product Information Center

PLCC: plastic J-leaded chip carrier

PLL: phase-locked loop

PLU: parallel logic unit

PM: processor module

PMA: program memory address

PMST: processor mode status register

PP: parallel processor

PPDS: parallel processing development system

PQFP: plastic quad flat pack

PRD: time period register

PREG: product register

PROB: program-read data bus

PROM: programmable read-only memory

PSTN: Public Switched Telephone Network

PWM: pulse-width modulation

R

RAM: random-access memory

ROM: read only memory

RPTC: repeat counter

RSR: receive shift register

RTI: real-time interrupt

S

S/N: signal-to-noise

SAM: shared access mode

SARAM: single-access RAM

SCI: serial communications interface; serial control interface

SCSI: small computer system interface

SDB: software development board

SDRAM: synchronous dynamic random-access memory

SDTR: synchronous serial-port transmit/receive register

SNR: signal-to-noise ratio

SOIC: small outline IC

SPC: section program counter; serial port control register

SPCE: serial port control extension register

SPI: serial-port interface

SRAM: static random-access memory

SRT: shift register transfer

SSPCE: synchronous serial-port control register

ST: status register

T

TBC: test bus controller

TC: transfer controller

TCR: timer-control register

TDM: time division multiplex

TEC: Texas Instruments embedded gate array

TI: Texas Instruments

TI & ME: DSP solutions web site individual page (i.e., Texas Instruments & Me)

TQFP: thin quad flat pack

TR: temporary register

TSPC: time-division-multiplexed serial-port control register

TTL: transistor-to-transistor logic

TTO: Technical Training Organization

U

UART: universal asynchronous receiver/transmitter

URST: reset asynchronous serial-port bit

V

VC: video controller

VLSI: very-large-scale integration

VRAM: video read-access memory

W

WD: watchdog

WDCR: watchdog control register

WWW: World Wide Web

X

XDS: extended development system

XSR: transmit shift register

Index