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.

TEXAS INSTRUMENTS

TMS320 Family Development Support

Reference Guide

1994

Digital Signal Processing Products

TEXAS INSTRUMENTS

Printed in U.S.A., February 1994

SPRU011D

Reference Guide

TMS320 Family Development Support

1994

TMS320 Family Development Support Reference Guide

Texas Instruments

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 Family 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.

More than 100 third-party suppliers provide development support tools and application hardware/software that supplement the TI DSP products and tools. Information on those offerings is provided in the *TMS320 Third-Party Support Reference Guide* (literature number SPRU052).

How to Use This Manual

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

Chapter 1, the *Introduction*, presents an overview of the TI DSP offerings and a listing of the TMS320 digital signal processors now available.

Chapter 2, *The TMS320 Digital Signal Processor Family,* describes the TMS320 family of digital signal processors (DSPs), lists the key features of the devices, and provides a TMS320 road map of products. Also discussed in this chapter are the common DSP applications and the performance benchmarks for the TMS320 DSPs.

Chapter 3, *Code-Generation Tools,* provides an overview of software development. A discussion of software development products includes information on the TMS320C2x/C5x and TMS320C3x/C4x C compilers, the TMS320 macro assembler/linker, the Tartan Ada compiling system for the

TMS320C30 and TMS320C40, and SPOX, a DSP operating system for the TMS320C3x, TMS320C4x, and TMS320C5x.

Chapter 4, *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 programmer's interface (C/assembly source debugger) and the TMS320 software simulators and emulators. Chapter 4 also covers system integration and evaluation tools, TMS320C2x/5x DSP starter kits (DSKs), TMS320 XDS upgrade packages, and the parallel processing development system (PPDS) for the TMS320C40.

Chapter 5, *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). It also covers the TMS320 software cooperative, with information on how to license TI software and on the software technology available for license.

Chapter 6, *TMS320 Seminars and Workshops,* covers seminars and three-day workshops offered by the TI Technical Training Organization (TTO). The chapter discusses design assistance services offered by the TI worldwide Regional Technology Centers (RTCs) and provides a listing of RTC offices and addresses.

Chapter 7, *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**,** *TMS320 Product Information,* covers tabulated information about TMS320 devices and tools such as power dissipation, operating frequency, package type, part number, temperature range, and upgrade kits for the XDS/22 systems. This appendix also provides a detailed explanation of device nomenclature.

Appendix B covers *Factory Repair and Exchange Instructions,* while Appendix C presents the *Program License Agreements.* Appendix D discusses the *ROM Codes,* Appendix E covers *Quality and Reliability,* and Appendix F covers *TMS320 EPROM Programming.*

Related Documentation

Texas Instruments provides extensive documentation to support the TMS320 family devices and development tools. The *TMS320 Third-Party Support Reference Guide* (literature number SPRU052) is a prime source of information. (See Chapter 6 of this book, *TMS320 Documentation and Technical Support* for complete lists of related materials.

Style and Symbol 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
```

Trademarks

IBM, OS/2, PC, PC-AT, PC-DOS, and PC-XT are trademarks of International Business Machines.
DEC, VAX, and VMS are trademarks of Digital Equipment Corp.
MAC II, MacIntosh, and MPW are trademarks of Apple Computer Corp.
MS-DOS is a trademark of Microsoft Corp.
SPOX, SPOX-DSP, and SPOX-LINK are trademarks of Spectron Microsystems, Inc.
Sun-3 and Sun-4 are trademarks of Sun Microsystems, Inc.
TI is a trademark of Texas Instruments Incorporated.
UNIX is a registered trademark of Unix Systems Laboratories, Inc.

If You Need Assistance. . .

If you want to. . .	Do this. . .
Request more information about Texas Instruments Digital Signal Processing (DSP) products	Write to: Texas Instruments Incorporated Market Communications Manager, MS 736 P.O. Box 1443 Houston, Texas 77251–1443
Order Texas Instruments documentation	Call the TI Literature Response Center: **(800) 477–8924**
Ask questions about product operation or report suspected problems	Call the DSP Hotline: **(713) 274–2320** (Monday–Friday, 8:00 AM – 5:00 PM CST)
Contact DSP hotline via facsimile	Call the DSP hotline FAX: **(713) 274–2324** (24 hours)
Contact DSP hotline via electronic mail:	**4389750@mcimail.com**
Request code or application support	Call the DSP BBS: **(713) 274–2323** (24 hours) or by *anonymous ftp* on the following Internet sites: ***evans.ee.adfa.oz.au*** (directory /mirrors/tibbs) ***ti.com*** (directory /mirrors/tms320bbs)
Request pricing and availability information on products	Call the local sales office: See list of offices in Section 7.3
Report mistakes in this document or any other TI documentation	Send your comments to: Texas Instruments Incorporated Technical Publications Manager, MS 702 P.O. Box 1443 Houston, Texas 77251–1443

Contents

Figures

Tables

Examples

Chapter 1

Introduction

This chapter presents an overview of the TI DSP offerings and a listing of the TMS320 digital signal processors. The listing provides a brief description of each of the five generations of TMS320 family products.

1.1 TI DSP Overview

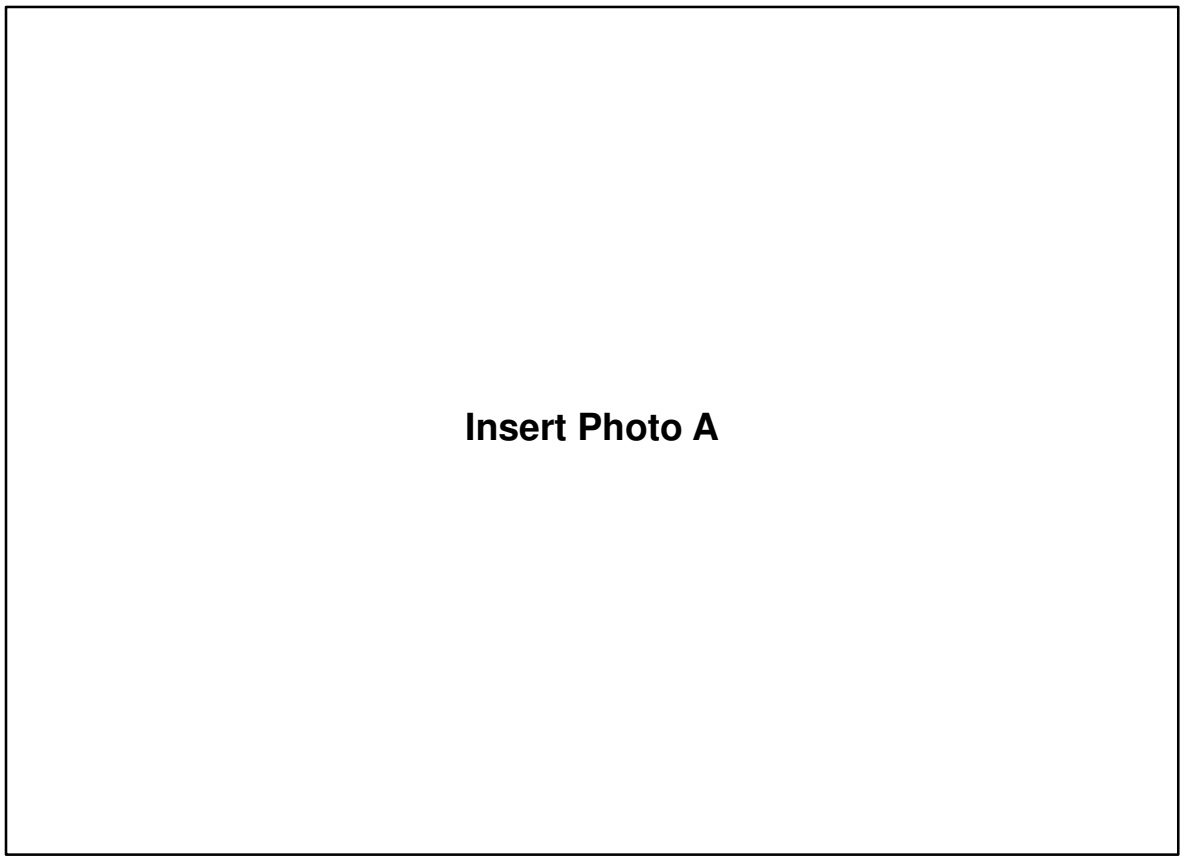
Since the first TMS320 digital signal processor (DSP) was introduced in 1982, Texas Instruments Incorporated (TI) has been dedicated to the advancement of digital signal processing technology and its applications. TI recognizes that fast time to market, increased productivity, and design ease are of primary importance in the development of DSP-based applications. Therefore, TI offers a comprehensive program of world-class development support for TMS320 DSPs that facilitates the design process from system concept to production and allows users to take advantage of rapidly evolving DSP technology.

The TMS320 support program includes leading-edge hardware and software development systems: optimizing C+ and C++ compilers, a user-friendly programming Interface consisting of C/assembly language source debuggers with code-profiling capabilities, low-cost evaluation tools, simulators, realtime emulators, realtime 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 1–1 shows the wide range of development tools available.

Various other support services are also available through the technical hotline, bulletin board service, field technical staff, and Technical Training Organization. A library of textbooks and over 2000 pages of application notes provide extensive information about the TMS320 DSP products.

Support from third-party companies complement the TI product/service offerings. Please consult the *TMS320 Third-Party Support Reference Guide* (literature number SPRU052) for details.

Figure 1–1. TMS320 Family Development Support



1.2 TMS320 Digital Signal Processors

The powerful instruction sets, inherent flexibility, high-speed number-crunching capabilities, and innovative architectural designs have made the high-performance cost-effective TMS320 digital signal processor family the ideal solution for many automotive, computer, consumer, industrial, military, and telecommunication applications. The TMS320 family (see Figure 1–2) consists of five generations. Further expansion of this family is planned, creating even higher-performing more versatile spin-offs and new generations. Presently, the members (by generation) of the TMS320 family are:

- TMS320C1x Devices
  - TMS320C10 — a 20-MHz, fixed-point, CMOS digital signal processor
  - TMS320C10-14 — a 14-MHz version of the TMS320C10
  - TMS320C10-25 — a 25-MHz version of the TMS320C10
  - TMS320C14 — a version of the TMS320C15 with on-chip peripherals of a microcontroller
  - TMS320E14 — an EPROM version of the TMS320C14
  - TMS320P14 — a one-time-programmable (OTP) version of the TMS320E14
  - TMS320C15 — a version of the TMS320C10 with expanded ROM and RAM
  - TMS320C15-25 — a 25-MHz version of the TMS320C15
  - TMS320E15 — an EPROM version of the TMS320C15
  - TMS320E15-25 — an EPROM version of the TMS320C15-25
  - TMS320LC15 — a low-power version of the TMS320C15
  - TMS320P15 — a one-time-programmable (OTP) version of the TMS320E15
  - TMS320P15-25 — an OTP version of the TMS320E15-25
  - TMS320C16 — a version of the TMS320C15 with larger on-chip ROM, wider address space, and higher performance
  - TMS320LC16 — a low-power version of the TMS320C16
  - TMS320C17 — a version of the TMS320C15 with on-chip telecommunication peripherals
  - TMS320C17-25 — a 25-MHz version of the TMS320C17
  - TMS320E17 — an EPROM version of the TMS320C17

- TMS320E17-25 — an EPROM version of the TMS320C17-25
- TMS320LC17 — a low-power version of the TMS320C17
- TMS320P17 — a OTP version of the TMS320E17
- TMS320P17-25 — an OTP version of the TMS320E17-25

❑ TMS320C2x Devices

- TMS320C25 — a 40-MHz, fixed-point CMOS DSP with twice the performance of first-generation devices
- TMS320C25-33 — a 33-MHz version of the TMS320C25
- TMS320C25-50 — a 50-MHz version of the TMS320C25
- TMS320E25 — an EPROM version of the TMS320C25
- TMS320C26 — a version of the TMS320C25 with three times the on-chip RAM. An RS-232C, I/O port, and a byte-wide EPROM bootloader are also included.
- TMS320C28 — a version of the TMS320C25 with expanded ROM and a power-down mode
- TMS320C28-50 — a 50-MHz version of the TMS320C28

❑ TMS320C3x Devices

- TMS320C30 — a 33-MFLOPS, (33-MHz) DSP with two memory-expansion buses, two serial ports, on-chip ROM floating-point, and CMOS
- TMS320C30-27 — a low-cost, 27-MHz version of the TMS320C30
- TMS320C30-40 — a 40-MHz version of the TMS320C30
- TMS320C30-50 — a 50-MHz version of the TMS320C30
- TMS320C31 — a low-cost, 33-MHz version of the TMS320C30 with one memory expansion bus, no on-chip ROM, one serial port, and preprogrammed ROM boot loader
- TMS320LC31 — a low-power version of the TMS320C31
- TMS320C31-27 — a low-cost, 27-MHz version of the TMS320C31
- TMS320C31-40 — a 40-MHz verson of the TMS320C31
- TMS320C31-50 — a 50-MHz verson of the TMS320C31

❑ TMS320C4x Devices

- TMS320C40 — a high-performance, 275-MOPS, 320 Mbytes/second, 32-bit floating-point, multiport, parallel-processing, digital signal processor
- TMS320C40-40 — a 40-MHz version of the TMS320C40

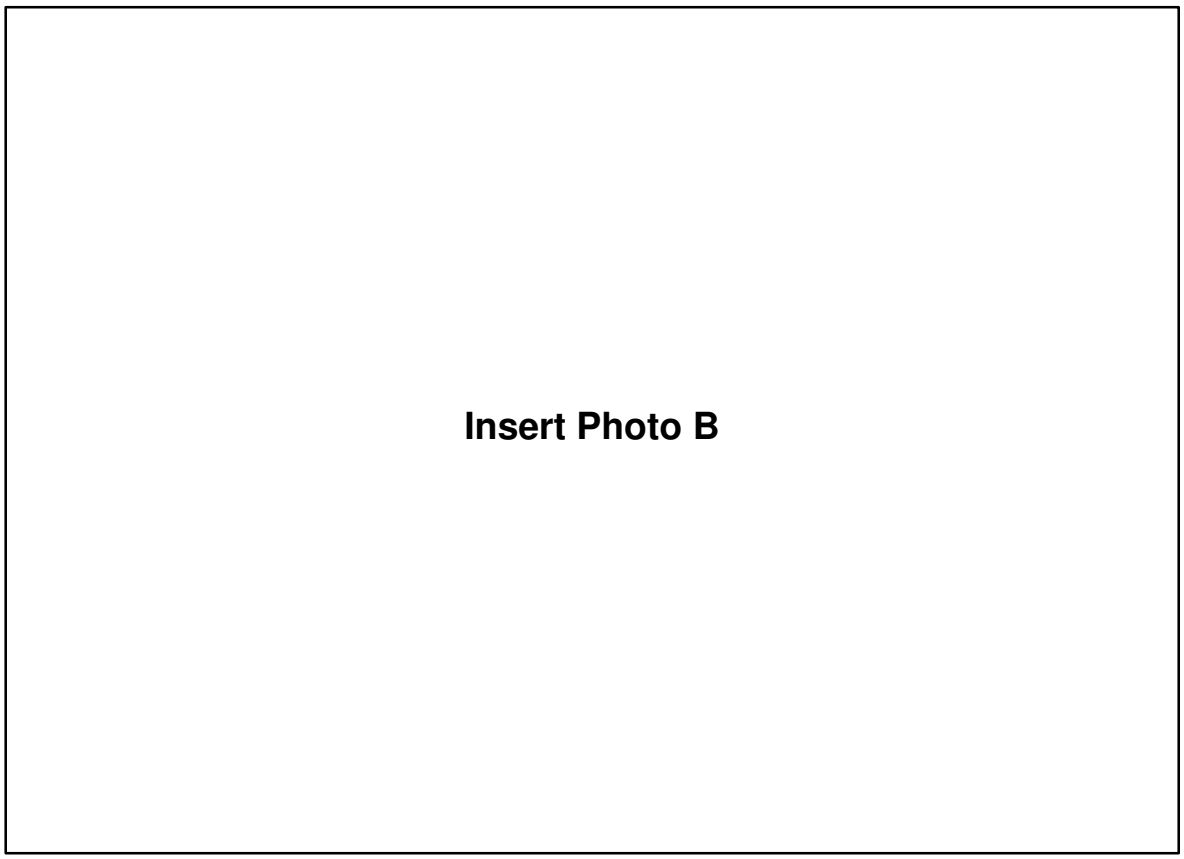
- TMS320C5x Devices
  - TMS320C50 — a fixed-point DSP capable of over twice the performance of previous TMS320 fixed-point generations. Available in instruction cycle times of 25, 35, or 50 ns with 10K-word RAM and 2K-word ROM.
  - TMS320LC50 — a low-power version of the TMS320C50. Available in instruction cycle times of 40 or 50 ns with 10K-word RAM and 2K-word ROM.
  - TMS320C51 — a low-cost, ROM-based version of the TMS320C50 with 2K-word RAM and 8K-word ROM.
  - TMS320BC51 — a version of the TMS320C51 with a preprogrammed ROM bootloader. Available in instruction cycle times of 25, 35, or 50 ns with 2K-word RAM and 8K-word ROM.
  - TMS320LC51 — a low-power version of the TMS320C51. Available in instruction cycle times of 40 or 50 ns with 2K-word RAM and 8K-word ROM.
  - TMS320LBC51 — a low-power version of the TMS320C51 with a preprogrammed ROM bootloader. Available in instruction cycle times of 40 or 50 ns with 2K-word RAM and 8K-word ROM.
  - TMS320C52 — a low-cost, high-performance ROM-based device. Available in instruction cycle times of 25, 35, or 50 ns with 1K-word RAM and 4K-word ROM.
  - TMS320BC52 — a version of the TMS320C52 with a preprogrammed ROM bootloader. Available in instruction cycle times of 25, 35, or 50 ns with 1K-word RAM and 4K-word ROM.
  - TMS320LC52 — a low-power version of the TMS320C52. Available in instruction cycle times of 40 or 50 ns with 1K-word RAM and 4K-word ROM.
  - TMS320LBC52 — a low-power version of the TMS320C52 with a preprogrammed ROM bootloader. Available in instruction cycle times of 40 or 50 ns with with 1K-word RAM and 4K-word ROM.
  - TMS320C53 — a highly integrated device with large on-chip RAM memory and ROM blocks. Available in instruction cycle times of 25, 35, or 50 ns with 4K-word RAM and 16K-word ROM.
  - TMS320BC53 — a version of the TMS320C53 with a preprogrammed ROM bootloader. Available in instruction cycle times of 25, 35, or 50 ns with 4K-word RAM and 16K-word ROM.

- TMS320LC53 — a low-power version of the TMS320C53. Available in instruction cycle times of 40 or 50 ns with 4K-word RAM and 16K-word ROM.
- TMS320LBC53 — a low-power version of the TMS320C53 with a pre-programmed ROM bootloader. Available in instruction cycle times of 40 or 50 ns with 4K-word RAM and 16K-word ROM.

❑ Application-Specific DSPs

- TMS320SS16—a single-chip, half-duplex transcoder that is pin selectable as a 64-kbps PCM passthrough device, a 32-kbps ADPCM transcoder, and a 16-kbps subband coding transcoder
- TMS320SA32—a CMOS, half-duplex, 32-kbps, ADPCM transcoder that is fully compatible with the 32-kbps ADPCM standards

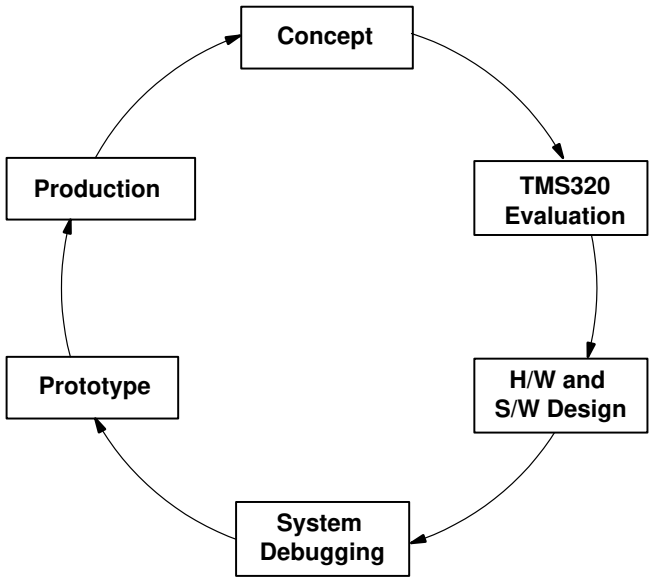
Figure 1–2. TMS320 Devices



1.3 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 3–1 shows a typical application flow.

Figure 1–3. Typical TMS320 Application Development Flow



- **Concept.** The typical application development flow often begins with the concept for a new or upgraded system design. You can look at system benchmarks (available on the TI DSP bulletin board service) and review application notes and algorithms for information on a proven solution to your specific application. In addition, you can reference TI user's guides and DSP textbooks or seek assistance from the TI field technical staff and hotline. Hands-on training on TMS320 devices and development tools is also available.

- **TMS320 system evaluation.** Usually a TMS320 DSP device is chosen according to performance-per-dollar criteria. To help you make the best selection, TI provides extensive documentation on device specifications and capabilities. You can use a variety of support tools during this stage—including DSP starter kits (DSKs), evaluation modules (EVMs), the SPOX operating system, the assembler/linker, and C, C++ language

compilers—to evaluate the processor's performance, benchmark time-critical code, and determine which TMS320 device to use when implementing a specific algorithm. Algorithms written in a high-level language such as C can be easily ported into and tested on TMS320 DSPs by using a TMS320 compiler.

- **Software and hardware designs.** You can design these modules in parallel by using a TMS320 simulator, assembler/linker, compiler, and EVM for software development and by using TMS320 behavioral models and emulators for hardware development. The hotline 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 via the bulletin board service (BBS). EPROM 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 1–4 shows development product integration. The appropriate TMS320 support product is indicated for each stage of development. Table 1–1 provides a matrix of the features of the TMS320 simulation/emulation development tools, comparing capabilities such as development purpose, software and hardware features, and amount of memory.

Figure 1–4. TMS320 Development Product Integration

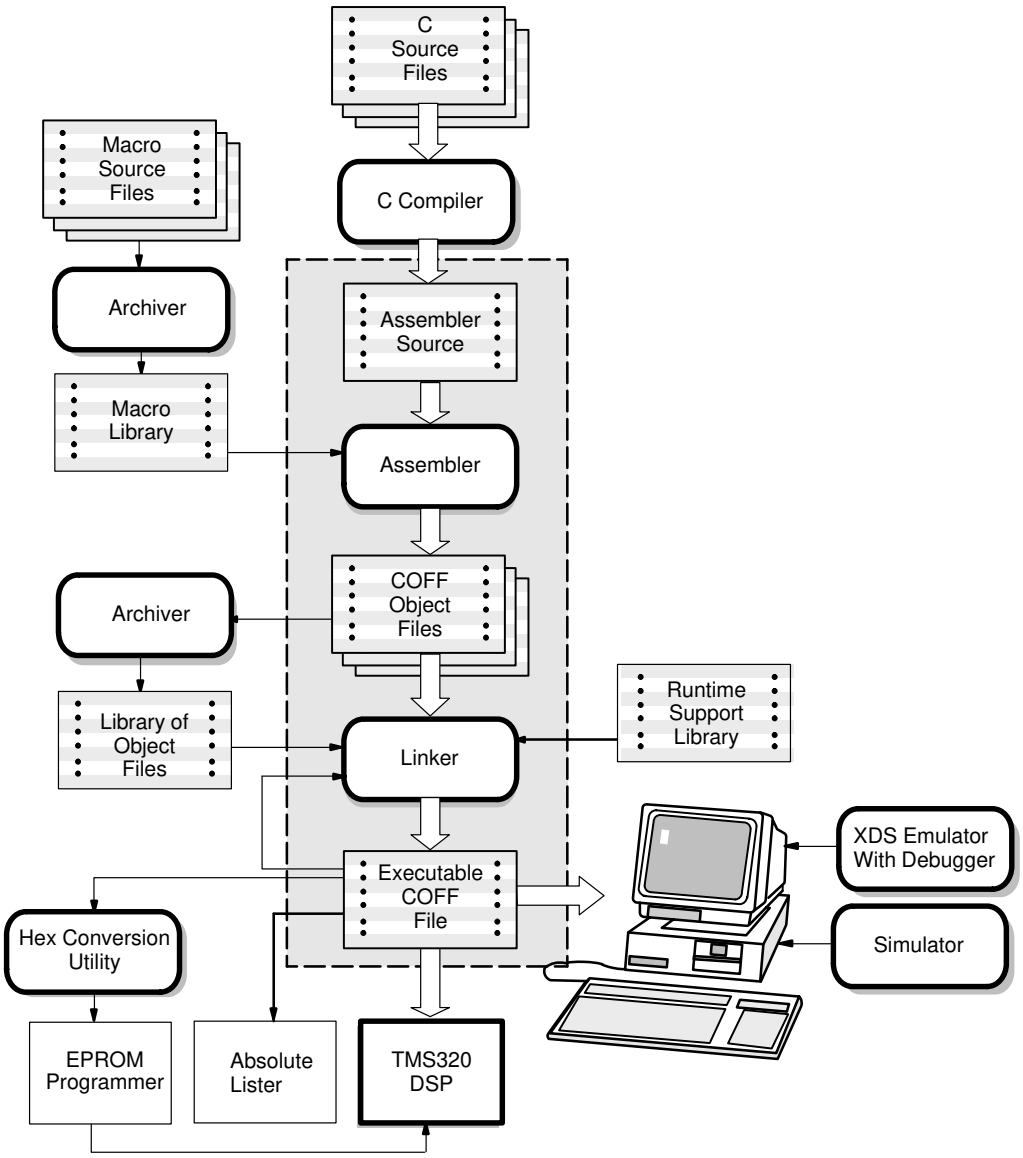


Table 1–1. Features of TMS320 Simulation/Emulation Development Tools

Features	EVM	Simulator	XDS/22	XDS510
TMS320 device supported:	’C10 ’C16/C1x ’C2x/C5x ’C3x ’C4x	’C1x ’C2x ’C3x ’C4x ’C5x	’C1x ’C2x	’C4x ’C5x ’C3x
Development purpose:				
Evaluation/benchmarking	Yes	Yes	Yes	Yes
Software design	Yes	Yes	Yes	Yes
Hardware design	No	No	Yes	Yes
Line-by-line or reverse assembler	Yes	Yes	Yes	Yes
Modify/display memory and registers	Yes	Yes	Yes	Yes
Single-stepping	Yes	Yes	Yes	Yes
Breakpoint on instruction acquisition	Yes	Yes	Yes	Yes
Breakpoint on memory access/read/write	No	Yes	Yes	Yes
Time-stamping/clock counter	No	Yes	Yes	No
Real-time trace samples	No	No	Yes	No
Multiuser system	No	Yes	No	Yes
HLL user interface	Yes	Yes	Yes	Yes
Files associated with I/O ports	No	Yes†	No	No
Full-speed in-circuit emulation:				
From on-board memory	Yes	N/A	Yes	No
From target memory	No	N/A	Yes	Yes
Amount of memory: (words)				
On-board program/data (TMS320C1x)	4K/--	N/A	4K	N/A
On-board program/data (TMS320C2x)	N/A	N/A	4K/4K	N/A
On-board program/data (TMS320C3x)	16K	N/A	N/A	N/A
On-board program/data (TMS320C4x)	384K	N/A	N/A	- - -
On-board program/data (TMS320C5x)	N/A	N/A	N/A	- - -
Program/data expansion	N/A	N/A	64K‡	N/A§

† This purpose is not supported by the ’C1x simulator.
‡ The Memory expansion board (included in the TMS320C2x XDS/22) allows for memory expansion to 64K total words of program and data memory, configurable in 1K-word blocks.
§ Program/data expansion is dependent upon the user’s target system.

Chapter 2

The TMS320 Digital Signal Processor Family

The TMS320 family of 16-/32-bit single-chip digital signal processors combines the flexibility of a high-speed controller with the numerical capability of an array processor, offering an inexpensive alternative to custom VLSI and bit-slice processors.

This chapter discusses these TMS320 DSP devices, applications, and benchmarks.

Topic **Page**

2.1 TMS320 Family Overview

The combination of the TMS320's high degree of parallelism and its specialized digital signal processing (DSP) instruction set provide speed and flexibility to produce a CMOS microprocessor family that is capable of executing up to 50 MFLOPS (million floating-point operations per second) or 275 MOPS (million operations per second). The TMS320 family optimizes speed by implementing functions in hardware that other processors implement through software or microcode. This hardware-intensive approach provides the design engineer with power previously unavailable on a single chip. The newest TI generation of floating-point DSPs — TMS320C4x — is designed for high-performance, parallel-processing applications.

The TMS320 family consists of five generations (three fixed-point and two floating-point) of digital signal processors. The fixed-point devices are members of the TMS320C1x, TMS320C2x, or TMS320C5x generation, and the floating-point devices belong to the TMS320C3x or TMS320C4x generation. Figure 2–1 shows the TMS320 family. Table 2–1 provides a tabulated overview of each member's memory capacity, number of I/O ports (by type), cycle time, package type, technology, and availability.

Many features are common among these TMS320 processors. When the term TMS320 is used, it refers to all five generations of DSP devices. When referring to a specific member of the TMS320 family (e.g.,TMS320C15), the name also implies enhanced-speed in MHz (-14, -25, etc.), erasable/programmable (TMS320E15), low-power (TMS320LC15), and one-time-programmable (TMS320P15) versions. Specific features are added to each processor to provide different cost/performance alternatives. Software compatibility is maintained throughout the family to protect your investment. Each processor has code-generation, system integration, and debug tools to facilitate the design process.

Figure 2–1. The TMS320 Family Road Map

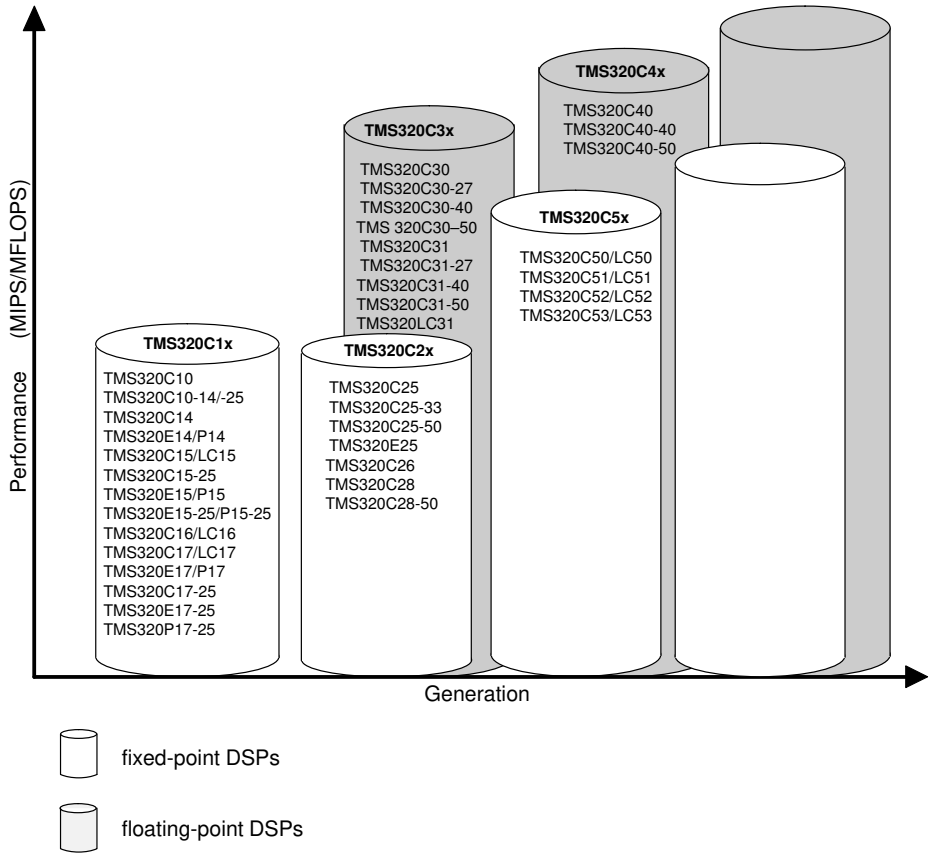


Table 2–1. TMS320 Family Overview

Data Type	Device†	Memory (Words)				I/O‡				On-Chip Timer	Cycle Time (ns)	Package
		On-Chip			Off-Chip	Ser	Par	DMA	Com			
		RAM	ROM	EPROM	Dat / Pro							
Fixed-Point (16-Bit Word Size)	TMS320C10§	144	1.5K	–	– / 4K	–	8×16	–	–	–	200	DIP/PLCC
	TMS320C10-14	144	1.5K	–	– / 4K	–	8×16	–	–	–	280	DIP/PLCC
	TMS320C10-25§	144	1.5K	–	– / 4K	–	8×16	–	–	–	160	DIP/PLCC
	TMS320C14	256	4K	–	– / 4K	1	7×16	–	–	4	160	PLCC
	TMS320E14§	256	–	4K	– / 4K	1	7×16	–	–	4	160	CERQUAD
	TMS320E14-25§	256	–	4K	– / 4K	1	7×16	–	–	4	167	CERQUAD
	TMS320P14	256	–	4K	– / 4K	1	7×16	–	–	4	160	PLCC
	TMS320C15§	256	4K	–	– / 4K	–	8×16	–	–	–	200	DIP/PLCC/ PQFP
	TMS320C15-25§	256	4K	–	– / 4K	–	8×16	–	–	–	160	DIP/PLCC
	TMS320E15§	256	–	4K	– / 4K	–	8×16	–	–	–	200	DIP/CER-QUAD
	TMS320E15-25	256	–	4K	– / 4K	–	8×16	–	–	–	160	DIP/CER-QUAD
	TMS320LC15	256	4K	–	– / 4K	–	8×16	–	–	–	200	DIP/PLCC
	TMS320P15	256	–	4K	– / 4K	–	8×16	–	–	–	200	DIP/PLCC
	TMS320P15-25	256	–	4K	– / 4K	–	8×16	–	–	–	160	DIP/PLCC
	TMS320C16	256	8K	–	– / 64K	–	8×16	–	–	–	114	PQFP
	TMS320LC16	256	8K	–	– / 64K	–	8×16	–	–	–	250	PQFP
	TMS320C17	256	4K	–	– / –	2	6×16	–	–	1	200/160	DIP/PLCC
	TMS320E17	256	–	4K	– / –	2	6×16	–	–	1	200/160	DIP

† Refer to Table A-2 for TMS320 DSP military part numbers and information.

‡ Ser = serial; Par = parallel; DMA = direct memory access (Int = internal; Ext = external); Com = parallel communication ports

§ A military version is available/planned; contact the nearest TI field sales office for availability.

Table 2–1. TMS320 Family Overview (Continued)

Data Type	Device†	Memory (Words)				I/O‡¶				On-Chip Timer	Cycle Time (ns)	Package Type
		On-Chip			Off-Chip	Ser	Par	DMA	Com			
		RAM	ROM	EPROM	Dat / Pro							
Fixed-Point (16-Bit Word Size)	TMS320LC17	256	4K	–	– / –	2	6×16	–	–	1	200	DIP/PLCC
	TMS320P17	256	–	4K	– / –	2	6×16	–	–	1	160	DIP
	TMS320C25§	544	4K	–	64K / 64K	1	16×16	Ext	–	1	100	PGA/PLCC/ PQFP
	TMS320C25-33	544	4K	–	64K / 64K	1	16×16	Ext	–	1	120	PLCC
	TMS320C25-50§	544	4K	–	64K / 64K	1	16×16	Ext	–	1	80	PGA/PLCC
	TMS320E25	544	–	4K	64K / 64K	1	16×16	Ext	–	1	100	PQFP/PLCC
	TMS320C26§	1.5K	–	–	64K / 64K	1	16×16	Ext	–	1	100	PLCC
	TMS320C28	544	8K	–	64K / 64K	1	16×16	Ext	–	1	100	PQFP/PLCC
	TMS320C28-50	544	8K	–	64K / 64K	1	16×16	Ext	–	1	80	PQFP/PLCC

† Refer to Table A-2 for TMS320 DSP military part numbers and information.
‡ Ser = serial; Par = parallel; DMA = direct memory access (Int = internal; Ext = external); Com = parallel communication ports
§ A military version is available/planned; contact the nearest TI field sales office for availability.
¶ Programmed transcoders (TMS320SS16 and TMS320SA32) are also available. See Section 2.6, *Application-Specific DSPs*, for details.

Table 2–1. TMS320 Family Overview (Continued)

Data Type	Device†	Memory (Words)				I/O‡¶				On-Chip Timer	Cycle Time (ns)	Package
		On-Chip			Off-Chip	Ser	Par	DMA	Com			
		RAM	ROM	EPROM	Dat / Pro							
Fixed-Point (16-Bit Word Size)	TMS320C50§	10K	BL	–	64K / 64K	2	64K×16#	Ext	–	1	50/35/ 25/20‖	PQFP
	TMS320C51	2K	8K	–	64K / 64K	2	64K×16#	Ext	–	1	50/35/ 25/20‖	PQFP/TQFP
	TMS320BC51	2K	BL	–	64K / 64K	2	64K×16#	Ext	–	1	50/35/ 25/20‖	PQFP/TQFP
	TMS320C52	1K	4K	–	64K / 64K	1	64K×16#	Ext	–	1	50/35/ 25/20‖	PQFP/TQFP
	TMS320BC52	1K	BL	–	64K / 64K	1	64K×16#	Ext	–	1	50/35/ 25/20‖	PQFP/TQFP
	TMS320C53	4K	16K	–	64K / 64K	2	64K×16#	Ext	–	1	50/35/ 25/20‖	PQFP/TQFP
	TMS320BC53	4K	BL	–	64K / 64K	2	64K×16#	Ext	–	1	50/35/ 25/20‖	PQFP/TQFP

† Refer to Table A-2 for TMS320 DSP military part numbers and information.

‡ Ser = serial; Par = parallel; DMA = direct memory access concurrent with CPU operation (Int = internal; Ext = external); Com = parallel communication ports, BL = bootloader

§ A military version is available/planned; contact the nearest TI field sales office for availability.

¶ Programmed transcoders (TMS320SS16 and TMS320SA32) are also available. See Section 2.6, *Application-Specific DSPs*, for details.

Sixteen of these parallel I/O ports are memory-mapped.

‖ Planned

Table 2–1. TMS320 Family Overview (Concluded)

Data Type	Device†	Memory (Words)				I/O‡¶				On-Chip Timer	Cycle Time (ns)	Package Type
		On-Chip			Off-Chip	Ser	Par	DMA	Com			
		RAM	ROM	EPROM	Dat / Pro							
Floating-Point (32-Bit Word Size)	TMS320C30†	2K	4K	–	16M□	2	16M×32◊	Int/Ext	–	2(6)‖	60	PGA and PQFP
	TMS320C30-50	2K	4K	–	16M□	2	16M×32◊	Int/Ext	–	2(6)‖	40	PGA and PQFP
	TMS320C30-27	2K	4K	–	16M□	2	16M×32◊	Int/Ext	–	2(6)‖	74	PGA and PQFP
	TMS320C30-40	2K	4K	–	16M□	2	16M×32◊	Int/Ext	–	2(6)‖	50	PGA and PQFP
	TMS320C30-50	2K	4K	–	16M□	2	16M×32◊	Int/Ext	–	2(6)‖	40	PGA and PQFP
	TMS320C31§	2K	☆	–	16M□	1	16M×32	Int/Ext	–	2(4)‖	60	PQFP
	TMS320LC31	2K	☆	–	16M□	1	16M×32	Int/Ext	–	2(4)‖	60	PQFP
	TMS320C31-27	2K	☆	–	16M□	1	16M×32	Int/Ext	–	2(4)‖	74	PQFP
	TMS320C31-40	2K	☆	–	16M□	1	16M×32	Int/Ext	–	2(4)‖	50	PQFP
	TMS320C31-50	2K	☆	–	16M□	1	16M×32	Int/Ext	–	2(4)‖	40	PQFP
	TMS320C40	2K	4K☆	–	4G□	–	4G×32◊	Int/Ext	6	2	40	PGA
	TMS320C40-40	2K	4K☆	–	4G□	–	4G×32◊	Int/Ext	6	2	50	PGA

† Refer to Table A-2 for TMS320 DSP military part numbers and information.
‡ Ser = serial; Par = parallel; DMA = direct memory access concurrent with CPU operation (Int = internal; Ext = external); Com = parallel communication ports
§ A military version available/planned; contact the nearest TI field sales office for availability.
¶ Programmed transcoders (TMS320SS16 and TMS320SA32) are also available. See Section 2.6, *Application-Specific DSPs*, for details.
‖ Includes the use of serial port timers
☆ Preprogrammed ROM bootloader
□ Single logical memory space for program, data, and I/O; not including on-chip RAM, peripherals, and reserved spaces
◊ Dual buses

2.2 TMS320C1x Devices

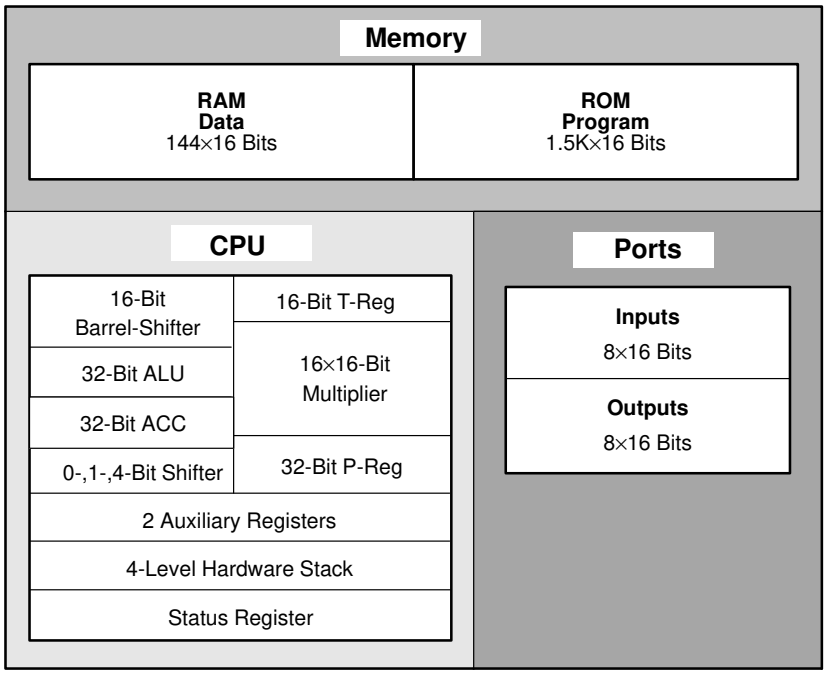
The TMS320C1x generation of the TMS320 family includes the TMS320C10, TMS320C10-14/C10-25, TMS320C14/E14/E14-25/P14, TMS320C15/E15/LC15/P15/P15-25, TMS320C15-25/E15-25, TMS320C16/LC16, and the TMS320C17/E17/LC17/P17, processed in CMOS technology. Note when referring to any member of the TMS320 family (e.g., TMS320C15) that the name also implies enhanced-speed versions as well as its associated erasable/programmable (EPROM) and one-time-programmable (OTP) products. TMS320C1x refers to all members within the TMS320C1x generation. Figure 2–2 is a block diagram of the architecture for the TMS320C10 family.

The **TMS320C10** has a 200-ns instruction cycle time, or 5 MIPS (million instructions per second) performance, and is capable of achieving a 16×16-bit multiplication within a single 200-ns cycle. The TMS320C10 is also available in a microcomputer version with 1.5K words of program ROM on-chip and up to 2.5K words of off-chip program memory for a total of 4K words. This ROM-code version can also operate entirely from off-chip ROM for ease of prototyping, code update, and field upgradeability.

The **TMS320C10-14**, a 14-MHz version of the TMS320C10, provides a low-cost alternative for DSP applications not requiring the maximum operating frequency of the TMS320C10. The device can execute 3.5 MIPS and has a 280-ns instruction cycle time.

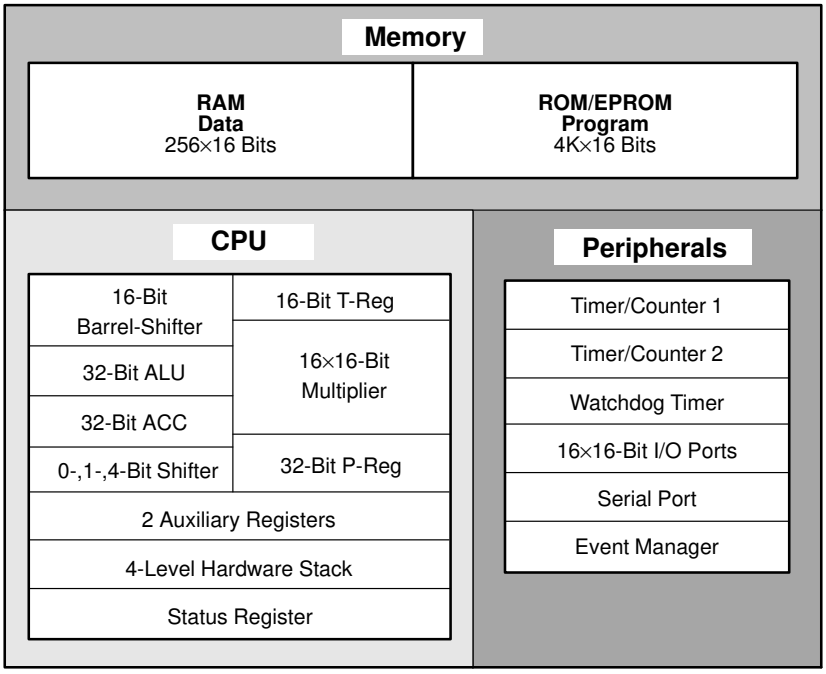
The **TMS320C10-25**, a 25-MHz version of the TMS320C10, has a 160-ns instruction cycle time. Its low power consumption and higher speed (6.25 MIPS) make it well-suited for high-performance DSP applications.

Figure 2–2. TMS320C10 Block Diagram



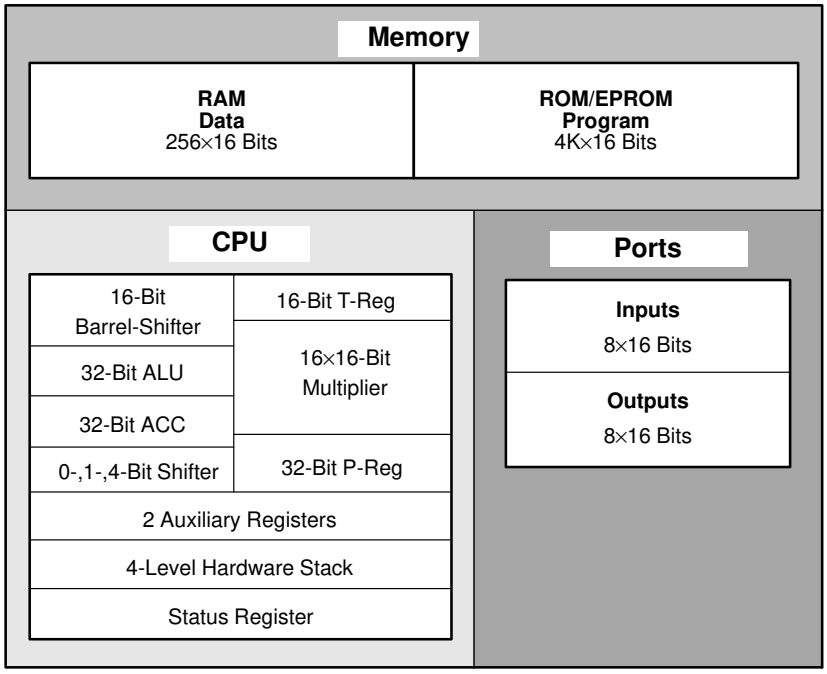
The **TMS320C14**, **TMS320E14**, **TMS320E14-25** and **TMS320P14** are fully object-code compatible with the TMS320C10 and offer the high performance of a DSP with the on-chip peripherals of a microcontroller. The TMS320C14 incorporates 256 words of RAM, 4K words of ROM (TMS320C14) or EPROM (TMS320E14 and TMS320P14), and five major peripheral blocks optimized for control applications. These peripheral blocks include four timers/counters (two 16-bit general-purpose timer/counters, one serial port timer, and one watchdog timer), an event manager, 16-bit selectable I/O pins, and an asynchronous serial port. Figure 2–3 is a block diagram of the architecture for the TMS320C14 family.

Figure 2–3. TMS320C14 Block Diagram



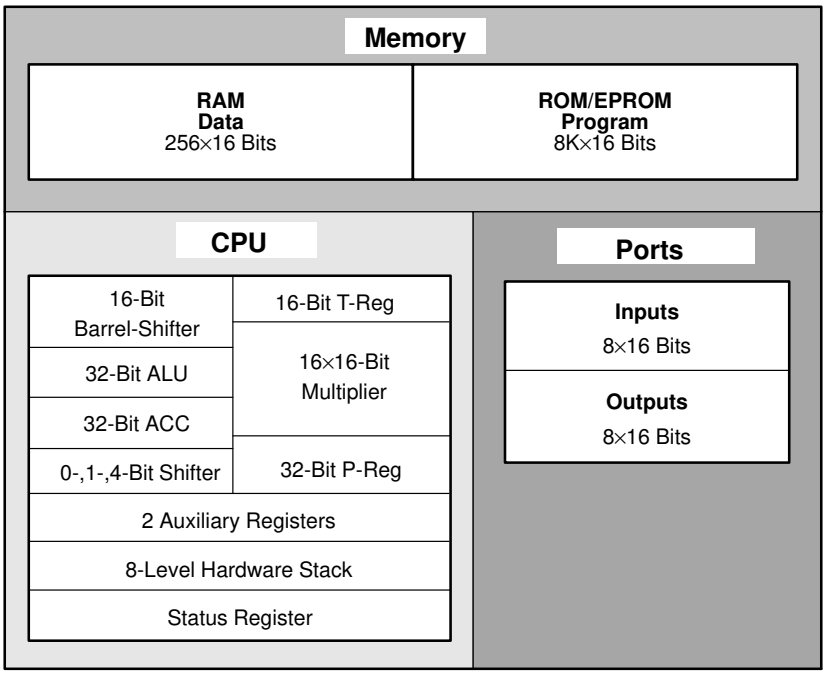
The **TMS320C15**, **TMS320E15**, **TMS320LC15**, and **TMS320P15** are fully object-code and pin compatible with the TMS320C10 and offer an expanded on-chip RAM of 256 words and an on-chip program ROM (TMS320C15 and TMS320LC15) or EPROM (TMS320E15 and TMS320P15) of 4K words. The devices are processed in CMOS technology. The TMS320C15 is also available in a 160-ns version, the **TMS320C15-25** and **TMS320E15-25**. Figure 2–4 is a block diagram of the architecture for the TMS320C15 family.

Figure 2–4. TMS320C15 Block Diagram



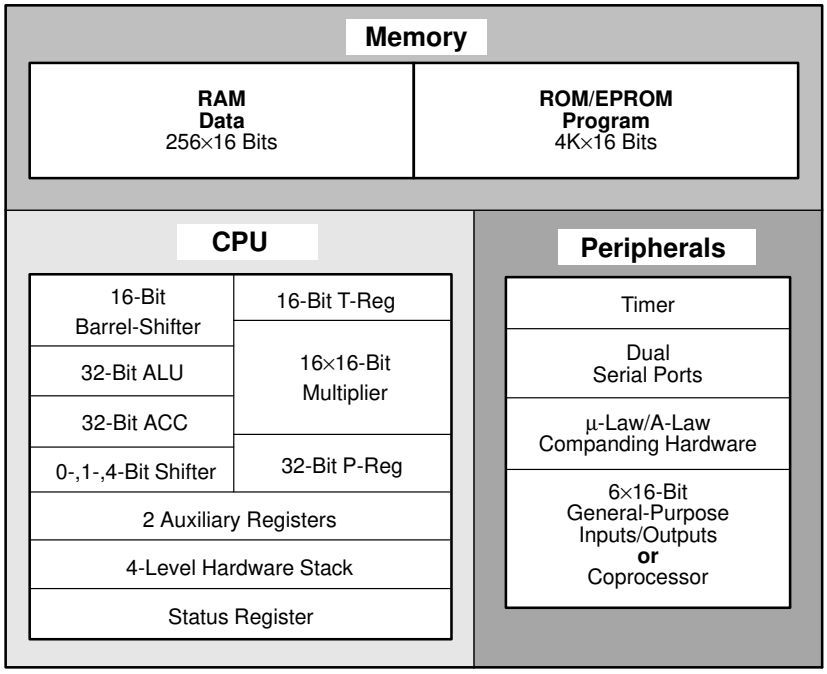
The **TMS320C16** and **TMS320LC16** are fully object-code compatible with the TMS320C10 and each device offers an on-chip RAM of 256 words, an expanded on-chip program ROM of 8K words, and 64K words of external memory. The TMS320C16 is the highest performance member of the TMS320C1x generation, operating at a 114-ns instruction cycle time. It also features an eight-level hardware stack, as well as separate I/O write and memory write signals. The TMS320LC16 is a low-cost version of the TMS320C16. Figure 2–5 is a block diagram of the architecture for the TMS320C16 family.

Figure 2–5. TMS320C16 Block Diagram



The **TMS320C17**, **TMS320E17**, **TMS320LC17**, and **TMS320P17** are dedicated microcomputers with 256 words of on-chip RAM and 4K words of on-chip ROM (TMS320C17 and TMS320LC17) or EPROM (TMS320E17 and TMS320P17). The TMS320C17 features a dual-channel serial interface, on-chip companding hardware (μ-law/A-law), a serial port timer, and a 16-bit latched coprocessor port for direct microprocessor/I/O interface. Figure 2–6 is a block diagram of the architecture for the TMS320C17 family.

Figure 2–6. TMS320C17 Block Diagram



Key features of the TMS320C1x generation are listed below. Specific devices that have a particular feature are enclosed in parentheses.

- CPU
  - 114-ns single-cycle instruction execution time (TMS320C16); 8.75 MIPS
  - 160-ns single-cycle instruction execution time (TMS320C14/C15-25/E14/E15-25/P15-25); 6.25 MIPS
  - 200-ns single-cycle instruction execution time (TMS320C10/C15/C17/E15/E17/LC15/LC17/P15/P17); 5 MIPS
  - 250-ns single-cycle instruction execution time (TMS320LC15/LC16); 4 MIPS
  - 280-ns single-cycle instruction execution time (TMS320C10-14); 3.5 MIPS
  - 32-bit ALU/accumulator
  - 16×16-bit parallel multiplier with a 32-bit product
  - 0- to 16-bit barrel shifter
- Peripherals
  - Four-level hardware stack (except TMS320C16)
  - Eight-level hardware stack (TMS320C16)
  - 16 individual bit-selectable I/O pins (TMS320C14/E14/P14)
  - Eight input and eight output channels
  - Event manager with 6-channel PWM D/A converter, plus up to 6 compare outputs and up to 4 capture inputs (TMS320C14/E14/P14)
  - Serial port with asynchronous mode (TMS320C14/E14/P14)
  - Dual-channel serial port with timer (TMS320C17/E17/P17/LC17)
  - Direct interface to combo-codecs (TMS320C17/E17/LC17/P17)
  - On-chip μ-law/A-law companding hardware (TMS320C17/E17/P17/LC17)
  - 16-bit coprocessor interface (TMS320C17/E17/P17/LC17)
  - Two general-purpose timers/counters (TMS320C14/E14/P14)
  - Watchdog timer (TMS320C14/E14/P14)
  - On-chip clock generator
- Memory
  - 144-word on-chip data RAM (TMS320C10)
  - 256-word on-chip data RAM (TMS320C14/C15/C16/C17)
  - 1.5K-word on-chip program ROM (TMS320C10)
  - 4K-word on-chip program ROM (TMS320C14/C15/C17/LC15/LC17)
  - 8K-word on-chip program ROM (TMS320C16)
  - 4K-word on-chip program EPROM (TMS320E14/E15/E17/P14/P15/P17)

- Memory interfaces
  - EPROM code protection for copyright security
  - 4K-word total external memory at full speed (except TMS320C16)
  - 64K-word total external memory at full speed (TMS320C16)
  - 16-bit bidirectional data bus at 50-Mbps transfer rate
- Single 5-V_{DC} supply
- Single 3-V_{DC} supply (TMS320LC15/LC17)
- 12.5-V_{DC} supply additionally required for programming EPROM (TMS320E14/E15/E17/P14/P15/P17)
- Packages
  - 40-pin DIP (TMS320C10/C15/C17/E15/E17/LC15/LC17/P15/P17)
  - 44-pin PLCC (TMS320C10/C15/C17/LC15/LC17/P15/P17)
  - 44-pin CERQUAD (TMS320E15/E17)
  - 68-pin PLCC (TMS320C14/P14)
  - 68-pin CERQUAD (TMS320E14)
  - 64-pin PQFP (TMS320C16)
- CMOS technology (TMS320C1x/E1x/LC1x/P1x)
- Commercial and military versions available

2.3 TMS320C2x Devices

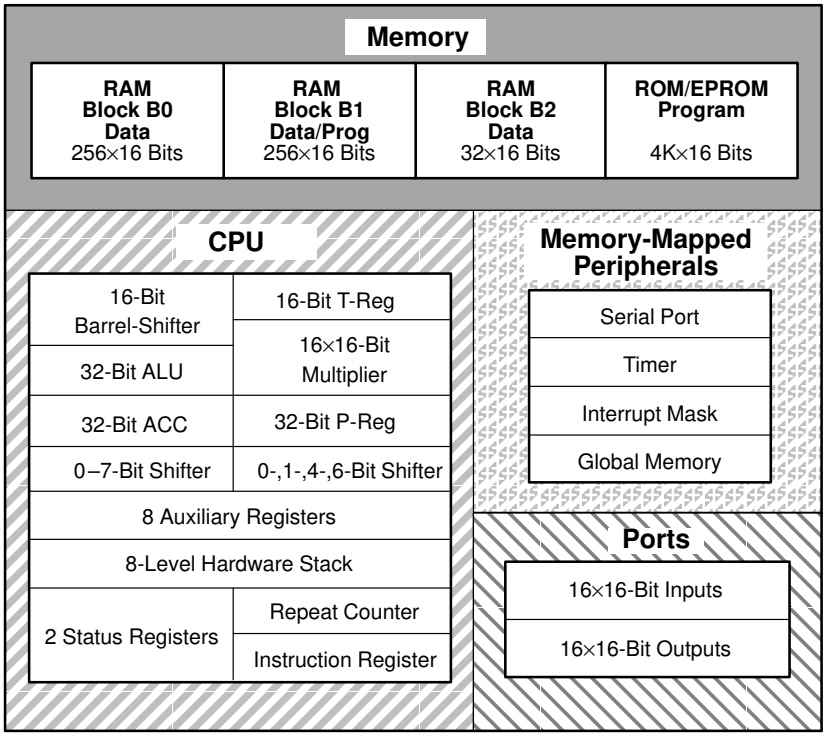
The TMS320C2x generation of the TMS320 family of digital signal processors includes the TMS320C25/E25, TMS320C25-33/C25-50, TMS320C26, TMS320C28, and TMS320C28-50. The architecture of these devices is extended from that of the TMS320C10. This section briefly describes each device, lists key features, and provides a block diagram.

The **TMS320C25** and **TMS320E25** are capable of an instruction cycle time of 100 ns. **They are pin and object-code upward-compatible with the TMS32020**. The TMS320C25's enhanced feature set includes 24 additional instructions (133 total), eight auxiliary registers, an eight-level hardware stack, 4K words of on-chip program ROM (TMS320C25) or EPROM (TMS320E25), a bit-reversed/indexed-addressing mode, and the low-power dissipation that is inherent to the CMOS process.

The **TMS320C25-33** is a 33-MHz version of the TMS320C25, capable of executing 8.25 MIPS. The TMS320C25-33 is object-code and pin compatible with the TMS320C25. It is designed for applications that require higher performance than that of the TMS320C1x devices but costs less than higher-speed TMS320C2x devices.

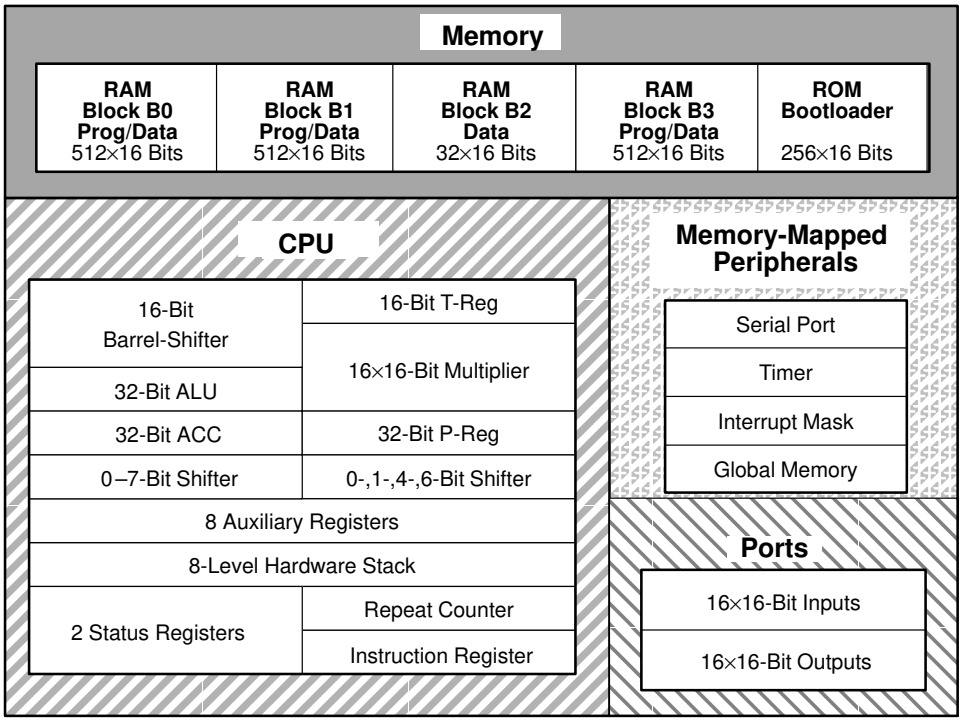
The **TMS320C25-50** is a 50-MHz version of the TMS320C25, capable of executing 12.5 MIPS. The TMS320C25-50 is object-code and pin compatible with the TMS320C25. Its higher speed makes it well-suited for high-performance DSP applications. Figure 2–7 is a block diagram of the architecture of the TMS320C25/'E25 family.

Figure 2–7. TMS320C25/E25 Block Diagram



The **TMS320C26** is a TMS320C25 with three times the on-chip RAM. It has a 1.5K×16-bit program/data RAM and 256×16-bit program ROM. It is pin and object-code compatible with the TMS320C25. The on-chip RAM is composed of three 512×16-bit segments, which can be independently configured as program memory or data memory. An RS232C, I/O port, or byte-wide EPROM bootloader is also included when the ’C26 is operated in the microcomputer mode. Figure 2–8 is a block diagram of the architecture of the TMS320C26 family.

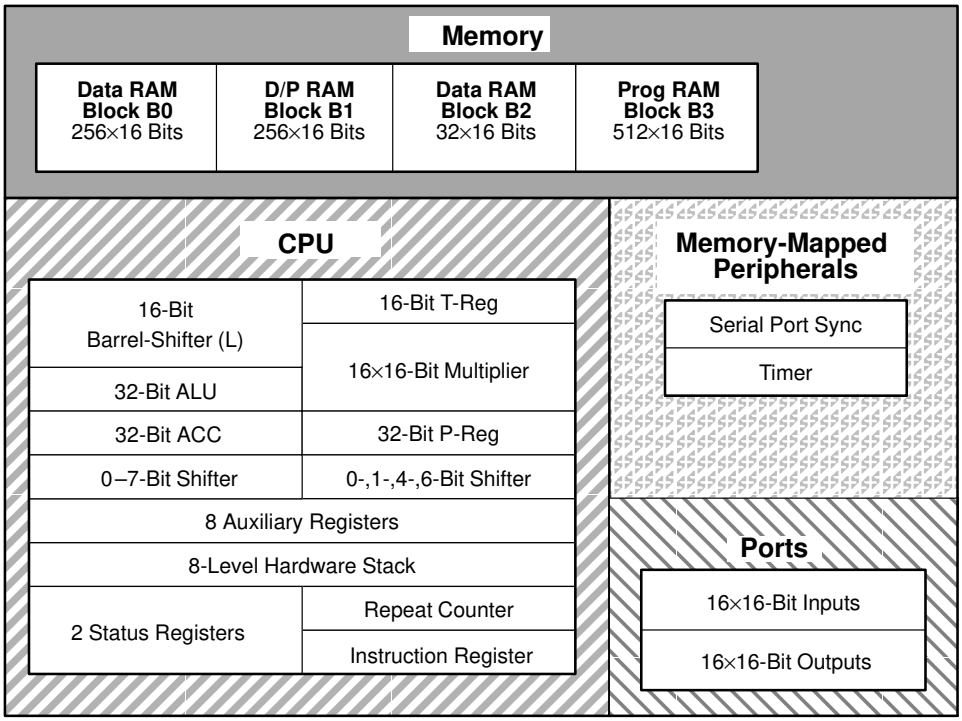
Figure 2–8. TMS320C26 Block Diagram



The **TMS320C28** is a TMS320C25 with 8K words of on-chip ROM and 544 words of RAM that can be configured between data and program. The off-chip memory is 64K-bit x 16-bit program and 64K-bit x 16-bit data. The ’C28 has power-down circuitry that reduces power consumption to 500 μW. It is also both object-code compatible with all ’C2x devices and pin compatible with the TMS320C25.

The **TMS320C28-50** is a 50-MHz version of the TMS320C28 capable of executing 12.5 MIPS. Its higher speed makes it well suited for high-performance DSP applications. Figure 2–9 is a block diagram of the architecture of the TMS320C28 family.

Figure 2–9. TMS320C28 Block Diagram



Key features of the TMS320C2x generation are listed below. Specific devices that have a particular feature are enclosed in parentheses.

- CPU
  - 80-ns single-cycle instruction execution time (TMS320C25-50); 12.5 MIPS
  - 100-ns single-cycle instruction execution time (TMS320C25/C26/E25); 10 MIPS
  - 120-ns single-cycle instruction execution time (TMS320C25-33); 8.3 MIPS
  - Single-cycle multiply/accumulates
  - Repeats for efficient use of program space and enhanced execution
  - Block moves for data/program management
  - Indexed-addressing mode
  - Bit-reversed indexed-addressing mode for radix-2 FFTs
  - 32-bit ALU/accumulator
  - Eight auxiliary registers with dedicated arithmetic unit
  - 16×16-bit parallel multiplier with a 32-bit product
  - 0- to 16-bit parallel shifter
  - Power-down mode (TMS320C28)
- Peripherals
  - Eight-level hardware stack
  - Sixteen input and sixteen output channels
  - Serial port for direct codec interface
  - Concurrent DMA using extended-hold operations
  - On-chip timer for control operations
  - On-chip clock generator
  - RS232, I/O and byte-wide EPROM bootloader
- Memory
  - 544-word programmable on-chip data RAM (except TMS320C26)
  - 1568-word programmable on-chip data RAM (TMS320C26)
  - 256-word on-chip ROM (TMS320C26)
  - 4K-word on-chip program ROM (TMS320C25)
  - 4K-word on-chip program EPROM (TMS320E25)
  - 128K-word total data/program memory space
- Memory interfaces
  - Wait states for slower off-chip communications
  - Synchronization input for synchronous multiprocessor configurations
  - Global data memory interface
- Upward compatibility with TMS320C1x source code
- Single 5-V_{DC} supply
- 12.5-V_{DC} supply additionally required for programming EPROM (TMS320E25)

- Packages
  - 68-pin PGA (TMS320C25/C26)
  - 68-pin PLCC (TMS320C25/C25-33/C25-50/C26)
  - 68-pin CERQUAD (TMS320E25)
  - 80-pin PQFP (TMS320C25/C20)
- CMOS technology
- Commercial and military versions available

2.4 TMS320C3x Devices

The TMS320C3x — the first 32-bit floating-point generation of the TMS320 family of DSPs — includes the TMS320C30/C30-27/C30-40/C30-50, TMS320C31/C31-27/C31-40/C31-50, and TMS320LC31. This section briefly describes each device, lists key features, and provides a block diagram.

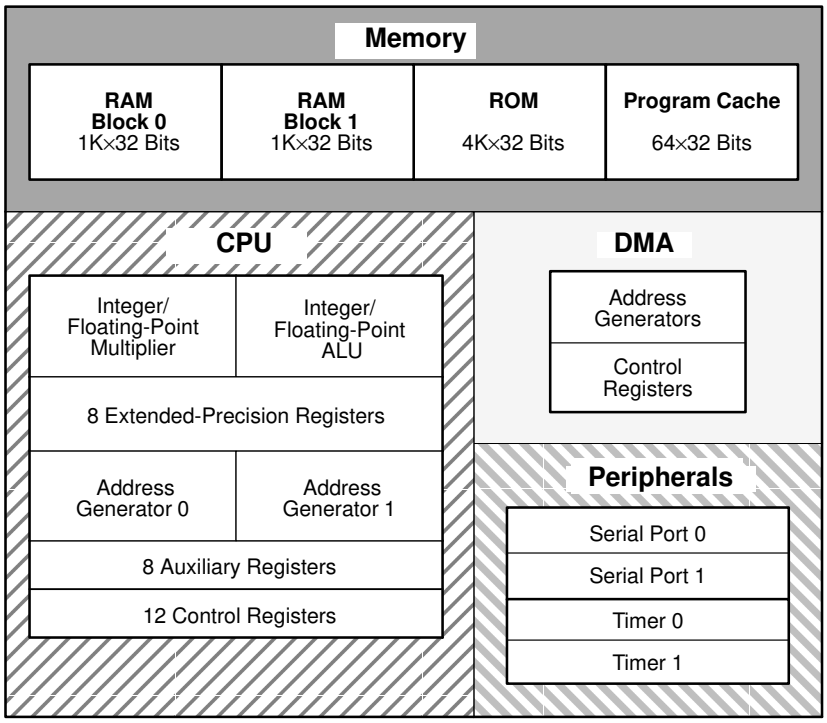
The **TMS320C30,** the first member of the TMS320C3x generation, is a high-performance, 32-bit device that is capable of executing up to 50 MFLOPS (million floating-point operations per second). The architecture of the TMS320C30 is specifically designed to be an efficient compiler platform. This feature makes it possible to program realtime DSPs, using the TI extended-precision, optimizing C compiler. Greater parallelism and general-purpose features facilitate high performance and ease of use. Applications are greatly enhanced by the large address space, multiprocessor interface, internally/externally generated wait states, two timers, two serial ports, multi-interrupt structure, and multitasking capabilities. The TMS320C30 supports a wide variety of system applications, ranging from host-processor to dedicated-coprocessor.

The **TMS320C30-27** is a 27-MHz version of the TMS320C30, capable of executing up to 27 MFLOPS. The TMS320C30-27 is object-code and pin compatible with the TMS320C30. It is designed for those applications that would greatly benefit from the attributes of a low-cost, 32-bit floating-point TMS320C30 processor.

The **TMS320C30-40** is a 40-MHz version of the TMS320C30, capable of executing up to 40 MFLOPS. The TMS320C30-40 is object-code and pin compatible with the TMS320C30. Its higher speed makes it well-suited for more demanding DSP applications.

The **TMS320C30-50** is a 50-MHz version of the TMS320C30, capable of executing up to 50 MFLOPS. The TMS320C30-50 is object-code and pin compatible with the TMS320C30. Its higher speed makes it well-suited for high performance DSP applications. Figure 2–10 is a block diagram of the architecture of the TMS320C30 family.

Figure 2–10. TMS320C30 Block Diagram



The **TMS320C31** is a low-cost TMS320C3x, incorporating all of the functionality and speed of the TMS320C30's core CPU at a cost comparable to that of the high-speed/fixed-point DSPs. This has been achieved by omitting the expansion bus, the 4K×32-bit internal ROM, and one serial port, adding a preprogrammed ROM bootloader, and by packaging the TMS320C31 in a 132-pin QUAD FLAT PACK (QFP) package. It is designed for DSP applications requiring cost-effective floating-point performance.

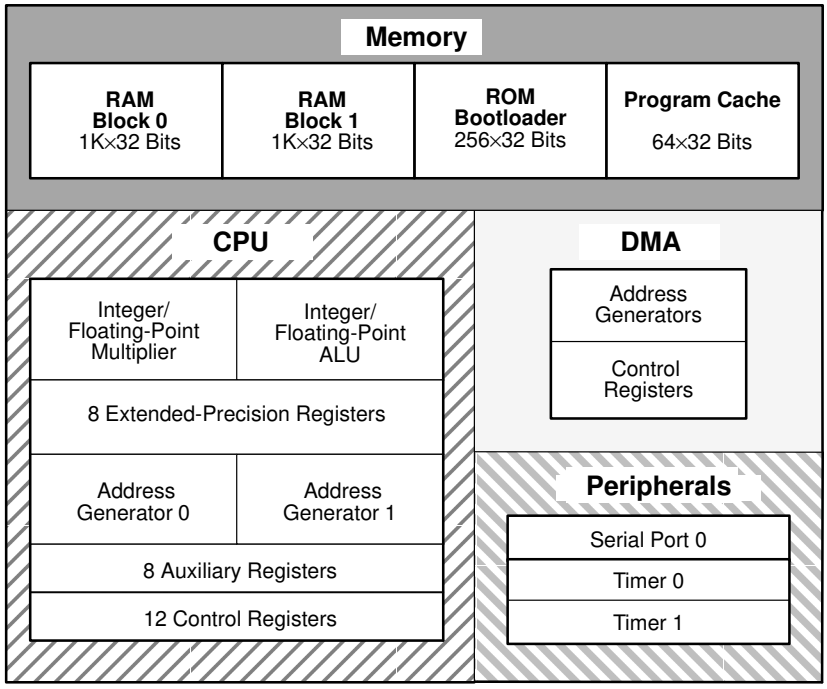
The **TMS320C31-27** is a 27-MHz version of the TMS320C31, capable of executing up to 27 MFLOPS. The TMS320C31-27 is object-code and pin compatible with the TMS320C31. This is TI's lowest cost floating-point solution available today for DSP applications.

The **TMS320C31-40** is a 40-MHz version of the TMS320C31, capable of executing up to 40 MFLOPS. The TMS320C31-40 is object-code and pin compatible with the TMS320C31. It is designed for DSP applications requiring cost-effective floating point performance.

The **TMS320C31-50** is a 50-MHz version of the TMS320C31, capable of executing up to 50 MFLOPS. The TMS320C31-50 is object-code and pin compatible with the TMS320C31. It is designed for DSP applications requiring high performance, cost-effective floating point performance.

The **TMS320LC31** is a 33 MHz, low-power version of the TMS320C31 that significantly reduces power consumption. The TMS320LC31 is object code and pin compatible with the TMS320C31. Figure 2–11 is a block diagram of the architecture of the TMS320C31/LC31 family.

Figure 2–11. TMS320C31/LC31 Block Diagram



Key features of the TMS320C3x generation are listed below. Specific devices that have a particular feature are enclosed in parentheses.

- CPU
  - 40-ns single-cycle instruction execution time (TMS320C30-50 and TMS320C31-50)
    - 50 MFLOPS
    - 25 MIPS
  - 50-ns single-cycle instruction execution time (TMS320C30-40 and TMS320C31-40)
    - 40 MFLOPS
    - 20 MIPS
  - 60-ns single-cycle instruction execution time (TMS320C30/C31/LC31)
    - 33.3 MFLOPS
    - 16.7 MIPS
  - 74-ns single-cycle instruction execution time (TMS320C30-27 and TMS320C31-27)
    - 27 MFLOPS
    - 13.5 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
  - Two- and three-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 accumulation/product, extended-precision registers
  - 32-bit barrel shifter

- Peripherals
  - On-chip memory-mapped Direct Memory Access (DMA) controller for concurrent CPU – I/O operations
  - Memory-mapped serial ports to support 8/16/24/32-bit full-duplex transfers
    - One serial port (TMS320C31/LC31)
    - Two serial ports (TMS320C30)
  - 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 single-cycle dual-access on-chip ROM block (TMS320C30/C30-27/C30-40/ C30-50)
  - Two 1K×32-bit single-cycle dual-access on-chip RAM blocks
  - 16M-word addressing space
  - Preprogrammed bootloader (TMS320C31/C31-27/C31-40/C31-50/LC31)
- Memory interfaces
  - Two memory/I/O expansion buses (TMS320C30/C30-27/C30-40/C30-50)
  - One external memory bus (TMS320C31/C31-27/C31-40/C31-50/LC31)
- Low-power, 3.3 volts (TMS320LC31)
  - Two power-down modes: 2-MHz operation and idle
- CMOS technology
- Packages
  - 181-pin PGA (TMS320C30/C30-27/C30-40/C30-50)
  - 132-pin PQFP (TMS320C31/C31-27/C31-40/C31-50/LC31)
  - 208-pin PGA (TMS320C30/C30-27/C30-40/C30-50)

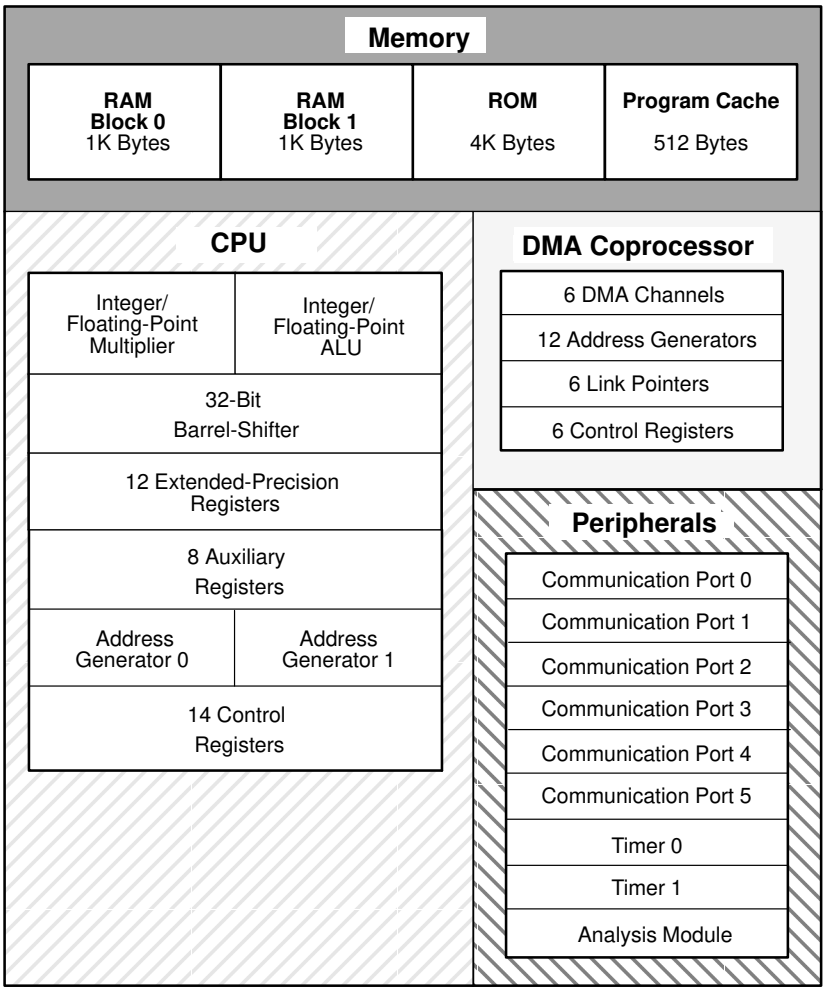
2.5 TMS320C4x Devices

TMS320C4x devices — the world's first 32-bit floating-point DSP designed for parallel processing — include the TMS320C40 and the TMS320C40-40. This section briefly describes the devices, lists key features, and provides a block diagram.

The **TMS320C40** — the first member of the TMS320C4x generation — is a 50-MHz, floating-point, CMOS DSP designed for parallel processing. The TMS320C40 architecture combines on one chip a high-performance central processing unit (CPU) with a multichannel direct-memory-access (DMA) coprocessor, six communication ports, memory, program cache, 32-bit global and local memory buses, two timers, and an analysis module. The TMS320C40 has a 40-ns cycle time, executes 275 MOPS (million operations per second), and achieves 320 Mbytes/second throughput.

The **TMS320C40-40** is a 40-MHz version of the TMS320C40 that has a 50-ns cycle time and is capable of executing 225 MOPS and 256 Mbytes/second. The TMS320C40-40 is object-code and pin compatible with the TMS320C40. Figure 2–12 is a block diagram of the architecture of the TMS320C40 family.

Figure 2–12. TMS320C40 Block Diagram



Key features of the TMS320C4x generation are listed below.

- CPU
  - 40-ns single-cycle instruction execution time (TMS320C40)
    - 275 MOPS
    - 320 Mbytes/second
  - 50-ns single-cycle instruction execution time (TMS320C40-40)
    - 225 MOPS
    - 256 Mbytes/second
  - Single-cycle 40/32-bit floating-point/integer multiplier and ALU operations
  - Interlocked instructions for parallel-processing support
  - Single-cycle reciprocal and reciprocal square root seed generation

- Byte and half-word manipulation capabilities
- Zero-overhead loops and single-cycle branches
- Single-cycle subroutine calls and returns
- Relocatable reset and interrupt vectors
- IEEE-standard floating-point format support
- Three-operand instructions
- 32-bit barrel shifter
- Two auxiliary register arithmetic units (address generators)
- Twelve extended-precision registers
- Eight auxiliary registers and fourteen control registers
- 32-bit instruction words, data words, and addresses
- Source code upward compatible from the TMS320C3x

❑ Peripherals
- Six communication ports for direct interprocessor or processor – I/O communications
- Self-programmable six-channel DMA coprocessor to maximize sustained CPU performance
- Analysis module for parallel-processing development and debugging
- (IEEE Std 1149.1–1990) scan interface for development and testing
- Two memory-mapped 32-bit timers

❑ Memory
- 128-word instruction cache
- Two 1K-word single-cycle dual-access RAM blocks
- One 4K-word single-cycle dual-access ROM block
- 4G-word addressing space

❑ Memory Interfaces
- Two independent 32-bit memory interfaces to support shared memory configurations
- Access-type status pins for intelligent bus arbitration
- Independent asynchronous high-impedance capability on data and address buses

❑ CMOS technology

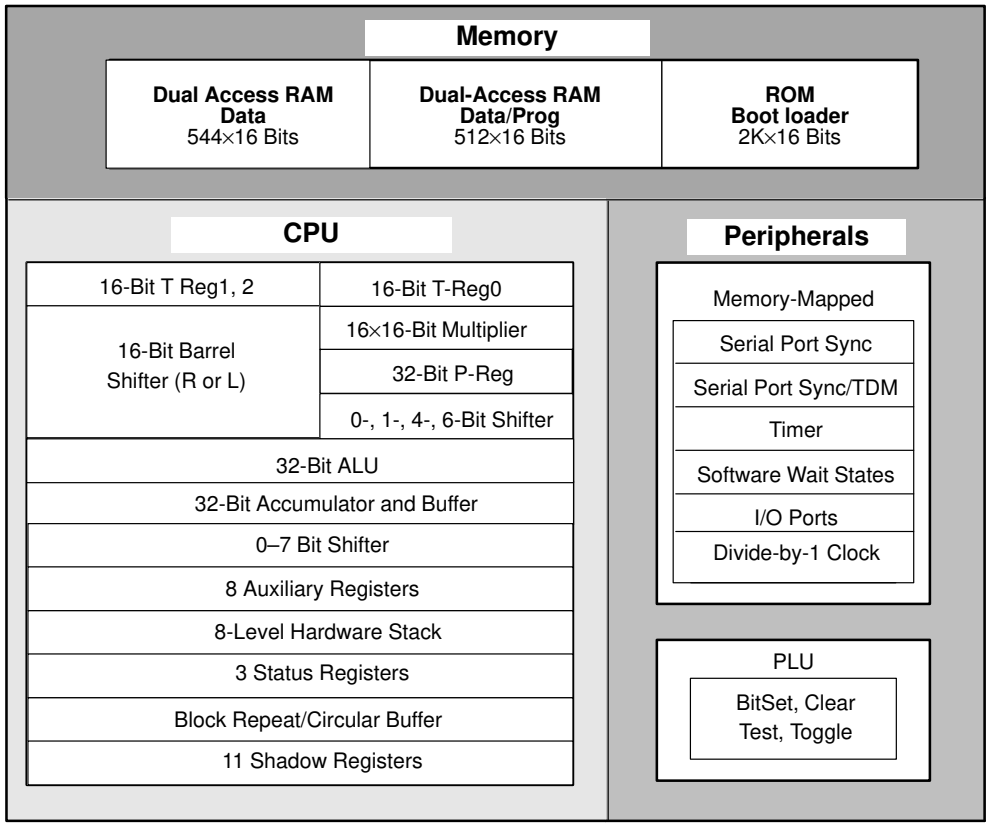
❑ 325-pin ceramic PGA package

2.6 TMS320C5x Devices

The TMS320C5x generation is the highest-performance generation of the TI 16-bit fixed-point digital signal processors and includes the TMS320C50 and the TMS320C51/C52/C53. The ’C5x performance level is achieved through a faster cycle time, larger on-chip memory space, and systematic integration of more signal-processing functions. This section briefly describes each ’C5x device, lists key features, and provides block diagrams.

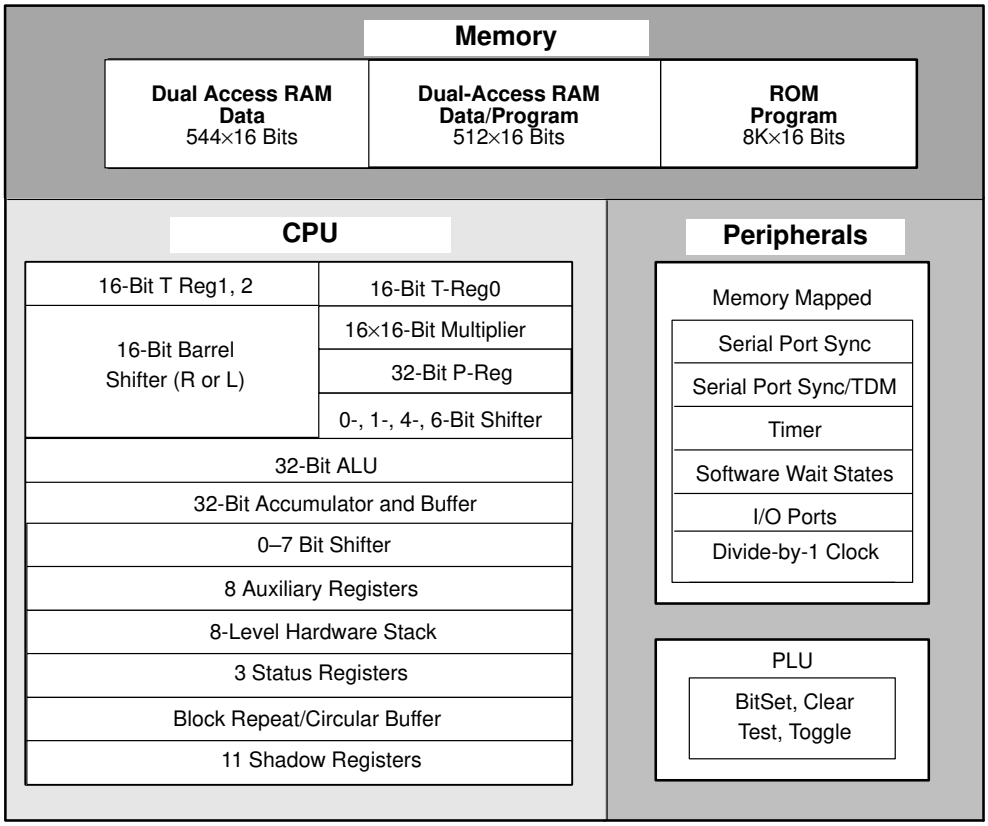
The **TMS320C50/LC50** features large on-chip RAM blocks, which are ideal for form function emulation. It is source-code upward-compatible with the first- and second-generation TMS320 devices. The TMS320LC50 is the low-power version of the ’C50 that significantly reduces power consumption.

Figure 2–13. TMS320C50 Block Diagram



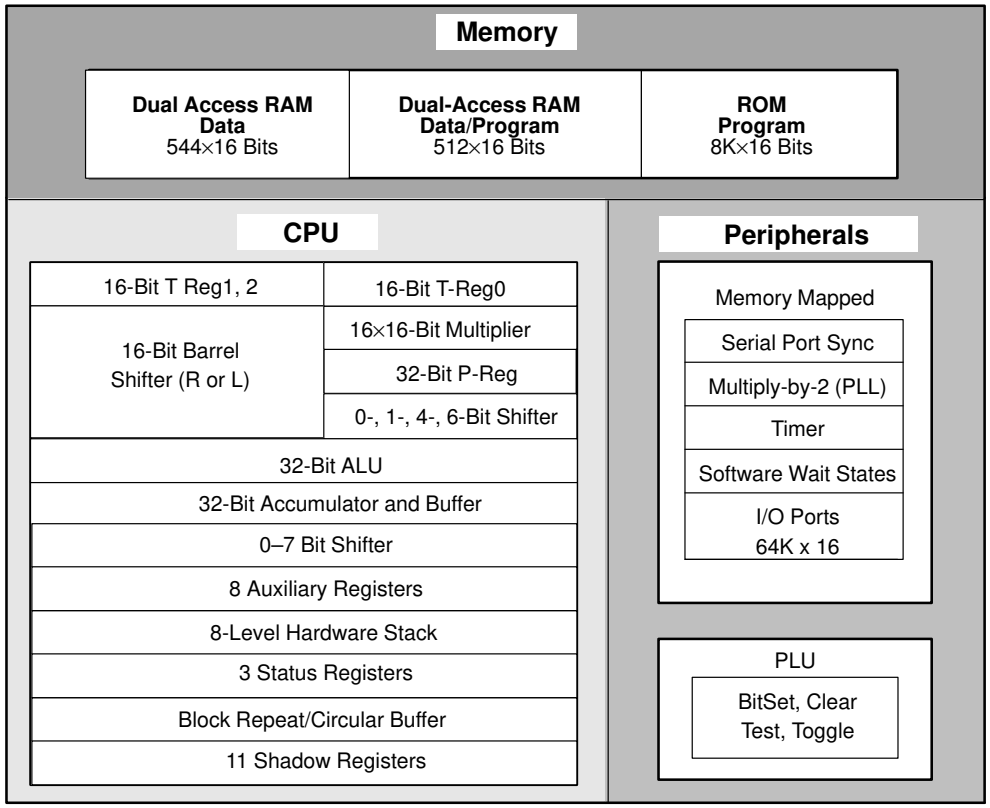
The **TMS320C51**/**LC51** features a user-maskable ROM for implementing information that was preprogrammed by TI. It is object-code and pin compatible with the TMS320C50. The TMS320LC51 is the low-power version of the ’C51 that significantly reduces power consumption.

Figure 2–14. TMS320C51 Block Diagram



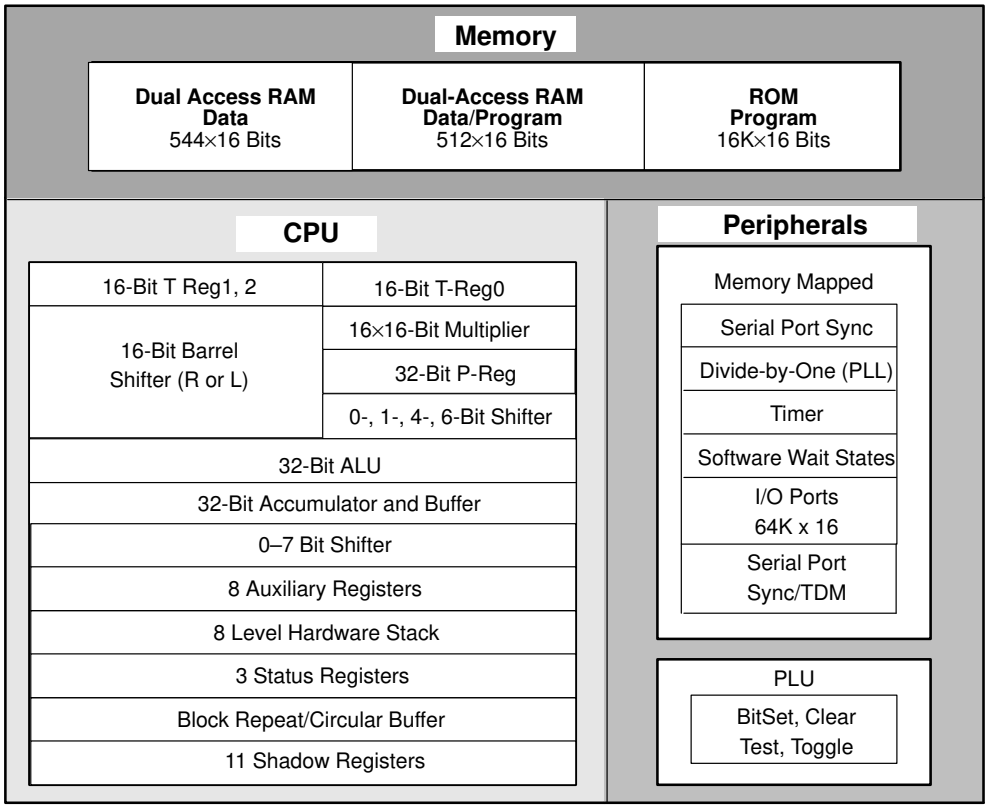
The **TMS320C52/LC52** provides a superb combination of both low cost and high performance. The ’C52 offers twice the performance of other devices within the same price range. The TMS320LC52 is the low power version of the ’C52 that significantly reduces power consumption.

Figure 2–15. TMS320C52 Block Diagram



The **TMS320C53/LC53** provides an even greater integration of on-chip ROM. Bringing memory on chip reduces power and board space requirements. On-chip memory also provides a considerable increase in performance per dollar spent.

Figure 2–16. TMS320C53 Block Diagram



Following are the key features of the TMS320C5x generation. Specific devices that have a particular feature are enclosed in parentheses.

- CPU
  - 25-ns single-cycle instruction execution time; 40 MIPS
  - 35-ns single-cycle instruction execution time; 28.6 MIPS
  - 50-ns single-cycle instruction execution time; 20 MIPS
  - Single-cycle multiply/accumulate for program code
  - Single-cycle/single-word repeats and block repeats for program code
  - Block memory moves for better program/data management
  - Four-deep executable pipeline to handle delayed branches, calls, and returns
  - Indexed-addressing mode
  - Bit-reversed/indexed-addressing mode to facilitate FFTs
  - Power-down modes
  - 32-bit ALU, 32-bit accumulator, and 32-bit accumulator buffer
  - Eight auxiliary registers with a dedicated arithmetic unit for indirect addressing
  - 16-bit parallel logic unit (PLU)
  - 16×16-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 codec
  - Time-division multiplexed (TDM) serial port (TMS320C50/’C51/’C53)
  - Interval timer with period and control registers for software stops, starts, and resets
  - Concurrent external DMA performance, using extended holds
  - On-chip clock generator
  - Divide-by-one clock generator (TMS320C50/C51/C53)
  - Multiply-by-two clock generator (TMS320C52)
  - Meets IEEE Std 1149.1–1990 testability specification

- Memory
  - 10K×16-bit single-cycle on-chip program/data RAM (TMS320C50)
  - 2K×16-bit single-cycle on-chip program/data RAM (TMS320C51)
  - 1K x 16 RAM (TMS320C52)
  - 4K x 16 RAM (TMS320C53)
  - 2K×16-bit single-cycle on-chip boot ROM (TMS320C50)
  - 8K×16-bit single-cycle on-chip program ROM (TMS320C51)
  - 4K x 16 ROM (TMS320C52)
  - 16K x 16 ROM (TMS320C53)
  - 1056×16-bit dual-access on-chip data/program RAM
- 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 TMS320C1x and TMS320C2x devices
- Single 5-V or 3.3-V supply
- Static CMOS technology
- 100-pin THIN QFP (’C51, ’C52, ’C53)
- 132-pin QUAD FLAT PACK package (’C50,’C51, ’C53)
- 100-pin QUAD FLAT PACK package (’C52)

2.7 Application-Specific DSPs

In addition to general-purpose DSPs, TI offers standard off-the-shelf DSPs for specific applications.

2.7.1 TMS320SS16

The TMS320SS16 is a single-chip, half-duplex transcoder that is pin selectable as a 64-kbps PCM passthrough device, a 32-kbps ADPCM transcoder, and a 16-kbps subband coding (SBC) transcoder. The transcoder implementation is fully compatible with the 32-kbps Adaptive Differential Pulse Code Modulation (ADPCM) standard defined by ANSI and Consultative Committee for International Telephone and Telegraph (CCITT) recommendation G.721 bis.

The TMS320SS16 has been optimized for either encoding from 64-kbps PCM to the lower transcoder data rate or decoding from the transcoder rate to 64-kbps PCM. The device can be configured as a transcoder in SBC or ADPCM mode for transmitting (encoding) or receiving (decoding), using A-law or μ-law PCM format. Master or slave mode selection allows the device to generate the serial port and framing clocks or to respond to those active in the system. The device also provides a PCM passthrough mode, overriding the transcoding options and enabling a 64-kbps signal transmission or connectivity testing. Data is received and transmitted through the device's serial ports. The subband coder algorithm has been developed by Atlanta Signal Processors, Inc.

The TMS320SS16 can be used in PBX, MUX, PC voice mail, and voice/sound synthesis applications where toll quality speech at lower bit rates is desirable. The configuration of the device can be changed during operation. Switching to the SBC option results in an 8-sample or 1-ms delay following acknowledgment of the select. Switching to the ADPCM option results in a single sample or 125-μs delay following acknowledgment of the select. Select is acknowledged within one sample period.

Key features of the TMS320SS16 include:

- Single-chip, half-duplex transcoder with pin-selectable bit rate:
  - 64-kbps PCM passthrough
  - 32-kbps CCITT G.721 bis and ANSI-compatible ADPCM
  - 16-kbps subband coder
- Transmit/receive select option
- Internal/external serial clock and framing selection option
- Half-duplex transcoder operation

- μ-Law/A-Law PCM selection option
- Serial I/O
- 40-pin DIP
- 20.5-MHz clock rate
- CMOS technology
- Single 5-V supply

2.7.2 TMS320SA32

The TMS320SA32 is a CMOS half-duplex, 32-kbps ADPCM transcoder. The transcoder implementation is fully compatible with the 32-kbps ADPCM standards defined by ANSI (May 1986) and CCITT (July 1986) G.721. A single device is necessary for either the encoding from 64-kbps PCM to 32-kbps ADPCM or for the decoding from 32-kbps ADPCM to 64-kbps PCM. The same device can be pin-selected for transmitting (encoding) or receiving (decoding) and for processing the PCM data, according to either the μ-law or A-law formats. Data is received and transmitted on a single 8-bit parallel data bus. The 4-bit ADPCM data is right-justified on the 8-bit bus. Data is read at 125-μs intervals when $\overline{\text{BIO}}$ is active low, and written within 53 μs after the read occurs. Typical power dissipation of the device is 165 mW.

Key features of the TMS320SA32 include:

- ANSI 32-kbps ADPCM compatible
- CCITT G.721 32-kbps ADPCM compatible
- Half-duplex transcoder operation
- Transmit/receive selection option
- μ-Law/A-Law PCM selection option
- 8-bit parallel data bus
- 8-kHz PCM sample rate
- 20.5-MHz clock rate
- 165-mW typical power dissipation
- CMOS technology
- Single 5-V supply
- 40-pin DIP

2.8 Customizable DSPs (cDSPs)

Texas Instruments also has the capability to allow customers in high-growth, high-volume applications to optimize DSPs and DSP systems for their specific needs. Together, the user and the TI design team find the best cost and performance alternatives for a customizable DSP (cDSP) solution. TI has over 30 standard single-chip DSP devices to meet a wide range of performance per dollar requirements. However, if user-application requirements and high volume dictate a custom DSP solution, the cDSP capability offers proven, industry-standard DSP architectures and world-class support from start to finish.

2.9 Typical Applications and Performance Benchmarks

The TMS320 family's unique versatility and realtime 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 2–2 lists typical TMS320 family applications.

Table 2–2. Typical Applications of the TMS320 Family

General-Purpose DSP	Graphics/Imaging	Instrumentation
Digital Filtering Convolution Correlation Hilbert Transforms Fast Fourier Transforms Adaptive Filtering Windowing Waveform Generation Discrete Cosine Transforms Hartley Transforms	3-D Rotation Robot Vision Image Transmission/ Compression Pattern Recognition Image Enhancement Homomorphic Processing Workstations Animation/Digital Map	Spectrum Analysis Function Generation Pattern Matching Seismic Processing Transient Analysis Digital Filtering Phase-Locked Loops
Voice/Speech	**Control**	**Military**
Voice Mail Speech Vocoding Speech Recognition Speaker Verification Speech Enhancement Speech Synthesis Text-to-Speech	Disk Control Servo Control Robot Control Laser Printer Control Engine Control Motor Control	Secure Communications Radar Processing Sonar Processing Image Processing Navigation Missile Guidance Radio Frequency Modems
Telecommunications		**Automotive**
Echo Cancellation ADPCM Transcoders Digital PBXs Line Repeaters Channel Multiplexing 1200- to 19200-bps Modems Adaptive Equalizers DTMF Encoding/Decoding Data Encryption Low-Speed Transcoders/ Vocoders ISDN Basic/Primary Rate Interfaces	FAX Cellular Telephones Speaker Phones Digital Speech Interpolation (DSI) X.25 Packet Switching Video Conferencing Spread Spectrum Communications Answering Machines	Engine Control Vibration Analysis 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

(Continued on the next page)

Table 2–2. Typical Applications of the TMS320 Family (Concluded)

Consumer	Industrial	Medical
Radar Detectors Power Tools Digital Audio/TV Music Synthesizer Educational Toys Answering Machines Multimedia	Robotics Numeric Control Security Access Power Line Monitors	Hearing Aids Patient Monitoring Ultrasound Equipment Diagnostic Tools Prosthetics Fetal Monitors

Table 2–3 shows benchmarks for the five generations. Note that these benchmarks offer only an approximation of the system-level performance that is to be expected. These performance values can be further improved by additional optimization of the algorithms for specific design goals, such as CPU loading and program space requirements. Table 2–4 shows the performance/generation relationship of several fundamental DSP operations for all five generations.

Table 2–3. TMS320 DSP System Benchmarks

	CPU Loading§				
Application	**TMS320C1x**	**TMS320C2x**	**TMS320C3x**	**TMS320C4x**	**TMS320C5x**
Echo cancellation (CCITT G.165) using echo length of 16 ms	—	50%	22%	~14.5%	25%
Data encryption (ANSI X3.92-1981) using data rate of 42 kbps	100%	52%	<15%	<10%	26%
Split-band modem (CCITT V.22/212A) using full-duplex	64%	30%	<14%	<10%	15%
32-kbps ADPCM (CCITT G.721) using half-duplex	100%	50%	<25%	<17%	25%
2.4-kbps LPC-10 (DOD 43) using half-duplex	76%†‡	40%‡	<17%	<12%	20%
2.4-kbps LPC-10 (DOD 52) using half-duplex	—	87%†‡	<35%	<24%	44%
16-kbps subband coder using full-duplex	64%‡	35%	<16%	<11%	18%

† Requires external program memory
‡ Requires external data memory
§ The assumed instruction rate for the loading factor is 5 MIPS (’C1x), 10 MIPS (’C2x), 20 MIPS (’C5x), 16.7 MIPS (’C30), and 25 MIPS (’C40).

Table 2–4. TMS320 DSP Family Benchmarks

Benchmark	TMS320C1x Cycles @ 114 ns		TMS320C2x Cycles @ 80 ns		TMS320C3x Cycles @ 50 ns		TMS320C4x Cycles @ 40 ns		TMS320C5x Cycles @ 25 ns	
FIR Filter										
20 Tap	49	179 kHz	29	431.03 kHz	25	800 kHz	24	1.04 MHz	27	1.48 MHz
64 Tap	133	65.9 kHz	73	171.73 kHz	69	290 kHz	68	367.64 kHz	71	563.4 kHz
67 Tap	139	63.1 kHz	76	164.47 kHz	72	278 kHz	71	352.11 kHz	74	540.5 kHz
IIR Filter										
4X Biquad	44	199.3 kHz	36	347.22 kHz	23	870 kHz	23	1.0875 MHz	36	1.11 MHz
5X Biquad	56	156.6 kHz	43	284.09 kHz	27	741 kHz	27	929.9 kHz	43	930.2 kHz
Transpose Biquad	69	127.1 kHz	54	231.48 kHz	37	541 kHz	37	675.6 kHz	54	740.7 kHz
Dot Product	6	0.684 µs	6	0.480 µs	4	0.200 µs	4	0.160 µs	6	0.150 µs
Matrix Multiply										
2×2 Times 2×2	24	2.74 µs	21	1.68 µs	12	0.600 µs	12	0.480 µs	21	0.525 µs
3×3 Times 3×1	24	2.74 µs	22	1.76 µs	13	0.650 µs	13	0.520 µs	22	0.550 µs
Memory-to-Memory FFT										
64-Point Radix 2	3687	420 µs	3088	247 µs	2025	101 µs	1515	60.6 µs	2895	72.38 µs
256-Point Radix 2	41478	4.73 ms	17602	1.408 ms	9058	0.453 ms	7645	0.3058 ms	15980	0.40 ms
1024-Point Radix 2	331237	37.8 ms	113467	9.007 ms	39429	1.972 ms	38629	1.545 ms	82761	2.07 ms
Port-to-Memory FFT										
64-Point Radix 2	2954	337 µs	1621	130 µs	947	47.3 µs	947	37.8 µs	1811	45.2 µs
256-Point Radix 2	41478	4.73 ms	8520	0.682 ms	4303	0.215 ms	4303	0.172 ms	9475	0.237 ms
1024-Point Radix 2	331237	37.8 ms	56286	4.503 ms	21105	1.055 ms	21105	0.844 ms	48055	1.20 ms

Chapter 3

Code-Generation Tools

Texas Instruments supports designers in the complete development of their application, from concept to production; resulting 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**

For part numbers, refer to Appendix A; for availability, cost, and other information, contact the nearest TI Field Sales Office or authorized distributor.

3.1 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 ANSI C compilers translate ANSI-standard, C language files into highly efficient TMS320 assembly language source files, which are then 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 TMS320C2x, TMS320C3x, TMS320C4x, and TMS320C5x 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 debugging information, which is used by the debugger, allowing source-level debugging in C. This enhances productivity and shortens the development cycle for both fixed-point ('C2x/'C5x) and floating-point ('C3x/'C4x) applications.

This section discusses two different compilers that have similar capability but different target processors—the *TMS320 fixed-point optimizing ANSI C compiler* that supports the 'C2x and 'C5x generations and the *TMS320 floating-point optimizing ANSI C compiler* that supports the 'C3x and 'C4x generations.

Notes:

1) The *TMS320 fixed-point optimizing ANSI C compiler* is a full-featured C compiler that supports the TMS320C2x and TMS320C5x product families. Throughout this book, this compiler is referred to as the *fixed-point compiler*.
2) The *TMS320 floating-point optimizing ANSI C compiler* is a full-featured C compiler that supports the TMS320C30, TMS320C31, and TMS320C40. Throughout this book, this compiler is referred to as the *floating-point compiler*.
3) The *TMS320 optimizing ANSI C compilers* are also referred to as the *TMS320 ANSI C compilers* and include both the fixed-point and floating-point compilers.

The TMS320 ANSI C compilers feature an ANSI-standard, runtime-support library 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 re-entrant.

The benefits of ANSI C support are:

- **Standardization**. C, although one of the most portable programming languages, has suffered from a lack of 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 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

3.1.1 Floating-Point Compiler

The floating-point compiler supports TMS320C3x and TMS320C4x devices and performs both global optimizations and loop optimizations such as strength reduction. Additionally, it thoroughly analyzes code in order to optimize the usage of memory and register variables. It also searches for vector/matrix operations by mapping memory accesses to those TMS320C4x 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 TMS320C4x. This simplifies the usage of this device's powerful features for multiprocessor applications.

3.1.2 Fixed-Point Compiler

The fixed-point compiler supports TMS320C2x and TMS320C5x DSPs and enhances productivity by enabling you to program in C. C code is easier to prototype, debug, and benchmark than assembly language.

The fixed-point compiler translates the widely used ANSI C language directly into highly optimized assembly language for either a TMS320C2x or a TMS320C5x 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.

3.1.3 Features of TMS320 ANSI C Compilers

Key features of the TMS320 ANSI C compilers include:

- ❑ Complete and exact conformance with the ANSI C specification
- ❑ Highly efficient code. The compiler incorporates state-of-the-art generic and target-specific optimizations (described in detail within the succeeding subsections).
- ❑ ANSI standard runtime-support library
- ❑ ROM-able, relocateable, and re-entrant 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 in-line 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.

3.1.3.1 Additional Features Specific to the Floating-Point C Compiler

In addition to the features common to all TMS320 ANSI C compilers, certain key features are specific to the floating-point compiler:

- 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 ’C30 also has a “short multiplication” option that generates efficient MPYI instructions (24×24-bit multiplication on the TMS320C3x yielding a 32-bit resultant) for integer multiplications instead of runtime-support calls.
- For the ’C30 and ’C40, additional runtime support is provided by SPOX, a real-time DSP operating system developed by Spectron MicroSystems. SPOX provides I/O support, a DSP math/matrix library, and a real-time operating system called *executive*.

3.1.4 TMS320 ANSI C Compiler Optimizations

The efficiency of a C compiler depends upon the scope and number of optimizations the C compiler performs, as well as upon the application. The TMS320 C compilers perform a wide variety of optimizations to improve the efficiency of the compiled code. The following list and explanations that follow 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
  - In-line expansion of function calls
- Optimizations Specific to the Floating-Point Compiler
  - Register variables
  - Register tracking/targeting
  - Cost-based register allocation
  - Autoincrement addressing modes
  - Repeat blocks
  - Delayed branches
  - Use of registers for passing function arguments
  - Parallel instructions
  - Conditional instructions
  - Loop unrolling
- Optimizations Specific to the Fixed-Point Compiler
  - Register variables
  - Cost-based register allocation

- Autoincrement addressing modes
- Repeat blocks
- Delayed calls and returns
- Arranging of variables on the local frame
- Elimination of unnecessary LDPK instructions

3.1.4.1 General-Purpose Optimizations

Differences in the same optimizations for floating-point and fixed-point compilers are highlighted.

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 3–1.

Alias Disambiguation

Programs written in C generally use many pointer variables. Frequently, compilers are unable to determine whether or not two or more l (lower case 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 3–1 and Example 3–3 for floating-point compilers, and Example 3–2 for fixed-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 3–1. 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 **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 3–2. 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.*

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 3–3 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 written as either floating-point or fixed-point repeat blocks, or by using 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.

In-Line Expansion of Function Calls

The special keyword *inline* directs the compiler to replace calls to a function with in-line code, saving the overhead associated with a function call as well as providing increased opportunities to apply other optimizations. See Example 3–3 and Example 3–4.

Example 3–3. 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.

Example 3–4. In-Line 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 in-line 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.*

Example 3–5. In-Line 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.*

3.1.4.2 Optimizations Specific to the Floating-Point Compiler

The following optimizations are peculiar 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 3–6 and Example 3–7.

Example 3–6. 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 3–6.

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 don't 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 3–7.

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 3–4 and Example 3–7.

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.

Example 3–7. 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.

Delayed 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 3–8.

Example 3–8. 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. See Example 3–3.

Parallel Instructions

Several floating-point instructions such as load/load, store/operate, and multiply/add can be paired with each other 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 3–7 and Example 3–4.

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 can determine 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 3–9.

Example 3–9. 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 3-instruction sequence and no branches.

3.1.4.3 Optimizations Specific to the Fixed-Point Compiler

The following optimizations are peculiar to the fixed-point compiler.

Register Variables

The compiler maximizes the use of the address registers of the ’C2x/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 don’t overlap can be allocated to the same register.

Autoincrement Addressing

For pointer expressions of the form ***p++***, the compiler uses efficient ’C2x/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 (repeat block) 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. For the ’C2x, which does not have a repeat block instruction, the compiler allocates an AR as the loop counter and implements the loop with a BANZ instruction.

Example 3–10. Repeat Blocks, Autoincrement Addressing, Strength Reduction, Induction Variable Elimination, Register Variables, and Loop Test Replacement

```
int a[10], b[10];
scale (int k)
{
     int i;
     for (i = 0; i < 10; ++i)
      a[i] = b[i] *k;
```

Fixed-point compiler (’C5x option) output is:

```
_scale:
     ...
     LRLK    AR6, _a         :AR6 = &a[0]
     LRLK    AR5, b          ;AR5 = &b[0]
     LACK    9
     SAMM    BRCR
     LARK    AR2, -3+LF1     ;AR2 = &k
     MAR     *0+, AR5
     RPTB    L4-1            ;repeat block 10 times
     LT      *+,  AR2        ;T = *AR5++
     MPY     *,   AR6        ;P = T ** AR2
     SPL     *+,  AR5        ;*AR6 ++= P
L4:
     ...
```

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.

Example 3–11. 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
```

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 a 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 3–11.

Arranging Variables on the Local Frame

Local variables are accessed by adjusting AR2 to point to the variable on the frame and then 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 – 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.

Example 3–12. 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 and 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.

Example 3–13. 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.

3.2 TMS320 ANSI C Compiler-Supported Hosts

The TMS320 ANSI C compilers support the following systems:

Host	System	Version
IBM PC or compatible	MS-DOS	3.0 or higher
	OS/2	1.1
Sun-3 or Sun-4	SunOS	4.1 or higher

3.3 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 TMS320C1x/C2x/C5x devices, whereas the floating-point assembler and linker support TMS320C3x/C4x devices. This section discusses the two assembler and linker toolsets (floating point and fixed point) and highlights their individual 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 user's code
- Maintains an SPC (section program counter) for each section of object code
- Defines and references global symbols
- Assembles conditional blocks
- Supports macros, allowing the user to define macros either in-line with or within a macro library

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 (common object file format) 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 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. Therefore, an object format converter must be utilized to convert the COFF object file into Intel, Tektronix, or TI-tagged hex object format. The converted file can then be downloaded into the EPROM programmer.

3.4 Tartan Ada Compiler

Tartan, Inc. develops full-function Ada optimizing compilation systems for the TMS320C3x and TMS320C4x DSPs. The compiler targeted to the 'C30 has been validated by the U.S. Government's *Ada Compiler Validation Capability* under test suite version 1.11.

Standard components of the compilation systems are:

- ❏ Highly optimizing compiler
- ❏ Ada Librarian
- ❏ Small, modular runtimes
- ❏ Standard, predefined Ada packages
- ❏ ARTclient package, permitting access to tasking data structures and operations
- ❏ Intrinsics package, permitting access to hardware capabilities
- ❏ Math package of elementary functions
- ❏ Cross-reference facility
- ❏ AdaScope debugger
- ❏ Linker, object librarian and utilities
- ❏ Help facility and documentation

The Ada compiler produces fast, compact code through the use of Ada-specific optimizations, optimizations that take advantage of the processor's architecture features, and a full range of classical optimizations. Five optimization levels permit proper optimization strategy at each point in the development cycle.

Support for Ada language features includes:

- ❏ Representation specifications for type sizes, record layout, enumeration values, object addresses, and interrupt entries
- ❏ Unchecked deallocation and conversion
- ❏ Insertion of routines written in machine code
- ❏ All Ada predefined pragmas and the implementation-defined pragmas Foreign_body and Linkage_name

'C3x- and 'C40-specific features include:

- Access to many processor-specific native instructions
- Circular and bit-reversed addressing
- Delayed branch functionality
- Repeat-block and repeat-single instructions

Compiler switches permit generation of 16-bit PC-relative conditional call instructions, control of interrupt latency time using the RPTS instruction, and specification of the number of wait states for the memory in which the program is executed.

The **Tartan Ada Librarian** implements the Ada language requirements for separate compilation and dependency control. It supports multiple libraries and multiple accesses. It also permits usage of non-Ada object files within an Ada program.

The **Tartan linker** is a fast, flexible linker for embedded Ada programs. It supports precise control over placement in memory of code, data, and constants for individual packages, modules, sections, and subprograms. It eliminates unused program sections from both the executable program images and the highly modularized Tartan Ada runtimes. An interface to the Texas Instruments TMS320C3x cross-assembler is also provided, including conversion of the output to Tartan's object file format.

The **Tartan AdaScope debugger** provides complete window-oriented, source-level, symbolic, and assembly-level debugging for Ada programs via Ada-like commands. It operates remotely from the host system to the DSP processor, using the TI XDS500 controller, or it can be run entirely on the host by using the simulator.

The Tartan Ada compilation systems can be hosted on either the Digital Equipment Corporation VAX series equipment running the VMS operating system (version 5.2 or later) or on the Sun SPARC platforms running the SunOS operations system (version 4.1.1 or later). Available options include an interface to Spectron's SPOX-DSP vector, matrix, and filter math functions; TI simulator; facilities for customizing the runtimes; and the AdaScope retargeting kit to adapt to a different hardware configuration or communications protocol.

For more information, contact:

Tartan, Inc.
300 Oxford Drive
Monroeville, PA 15146
Telephone: (412) 856–3600
FAX: (412) 856–3636

3.5 Tartan C++ Compiler

The Tartan C++ includes facilities for building, debugging, and maintaining real-time DSP applications using the C++ high-level programming language. Tartan offers two C++ development systems, one for use with the Texas Instruments' TMS320C3x DSPs and the other targeted for use with the TMS320C4x DSP devices.

Each Tartan C++ compilation system includes the following components:

- Tartan C++ optimizing compiler
- Tartan linker
- Shell to invoke compiler and linker
- C and C++ runtime libraries
- Tartan C debugger for TI XDS510 or TI XDS510WS
- Object file utilities
- Complete documentation
- Optional C++ simulator for the TMS320C3x or TMS320C4x DSP devices.

The Tartan C++ compilers process programs written in either the C or C++ languages. The compilers support several extensions and compatibility features in C and C++:

- Hardware floating-point computations in 32-bit single and 40-bit extended precision formats
- Software emulated double-precision (64-bit) floating-point computations
- Built-in complex numbers (C++ only)
- Enhanced asm statements with symbolic references to C and C++ code
- Optional K&R (Kernighan and Ritchie) compatible C code
- Optional cfront-compatible C++ code. (Cfront is the original C++ to C translator and was a product name from AT&T.)

The Tartan **linker** provides flexible support for memory layout, overlays, page table construction, and library searches. Sections can be allocated manually, by section name, or module name. Special features on either the 'C3x or 'C4x support the placement and the initialization of data for either RAM or ROM implementations. The algorithms used are compatible with those implemented by the Texas Instruments linker. The Tartan linker also allows you to link Tartan C++ program object files with object files created with TI's C compiler.

The Tartan **shell** is a cc-like shell program that is provided to simplify the compilation and/or linking of DSP applications. Shell options provide complete

control over the compilation and linking process, including generation of debug information, setting compiler optimization levels, controlling message generation, displaying tool invocations, etc.

The Tartan C **debugger** is a source-level, symbolic debugger for C programs. It's multiwindow user interface lets you monitor source code, machine code, C data, and processor registers simultaneously. The debugger supports command scripts, session logs, interactive evaluation of C expressions, conditional and unconditional breakpoints, and display of variable contents, types and code.

The object file utilities package includes a number of convenient object file utilities, including:

- Object file librarian to create and maintain object code libraries
- Converters for two-way compatibility with the Texas Instruments traditional COFF object file format
- Tools for viewing the contents of object and executable files

The Tartan C++ compiler is capable of running on several host systems such as:

- PCs running DOS, OS/2, or Microsoft Windows
- Sun SPARC Workstations running SunOS 4.1

The SPARC-hosted Tartan C++ versions support TI's XDS-510WS emulator for the 'C3x and 'C4x DSP families. It also supports TI XDS-500 and XDS-510 through a workstation-to-PC serial link.

The PC-hosted Tartan C++ versions support TI's XDS-510XL emulator for 'C3x and 'C4x development.

For more information, contact:

Tartan, Inc.
300 Oxford Drive
Monroeville, PA. 15146
Phone: 412-856-3600
Fax: 412-856-3636

3.6 SPOX—TMS320 DSP Operating System

SPOX is the industry's first realtime DSP operating system. The SPOX operating system is a powerful software tool, developed by Spectron Microsystems Inc., which provides a hardware-independent software base for realtime DSP applications. SPOX features a set of high-level, C-callable, software functions, which are independent of the underlying hardware platform, thus insulating realtime DSP applications from many low-level system details. The SPOX operating system plays an integral role in application development, from concept of new algorithms to integration of application software onto production hardware.

SPOX differs from other operating systems in that its capabilities are *application-specific*, augmenting high-level programming languages like C with functional components targeted especially for realtime DSP. Provided with SPOX is an extensible-DSP math package, configurable-hierarchical memory management capabilities, device-independent stream I/O, and a multitasking kernel. SPOX is an operating system for the TMS320 DSPs that offer developers two important benefits—software productivity and application portability.

The cost of developing software for advanced realtime DSP applications continues to escalate. Dramatic improvements in hardware price/performance have led to a software crisis in which new, more sophisticated DSP processors become available faster than applications can be developed.

Gains in productivity are achieved through off-the-shelf software functions, supported by SPOX, which accelerate design and implementation of applications and provide flexibility for growth and maintenance of the product. These functions are equally important in achieving portability, in that the same programming interface is implemented across all of current TI DSP architectures and next-generation DSP processors as well.

3.6.1 SPOX Software Components

SPOX is packaged as a system of software components that can be ported to your own TMS320 DSP hardware platform and then integrated with application programs targeted for this platform. A key feature of all SPOX components is reusability, meaning that a tested and documented body of off-the-shelf software can be embedded in a wide range of systems and need not be redundantly developed with each new application.

The components of SPOX have been designed to provide software functions that address the different requirements arising in advanced DSP systems:

- SPOX-OS enables concurrent program tasks to synchronize their execution and respond to external events in realtime. It also provides dynamic memory management and device-independent stream I/O.

- SPOX-LIBC enables programs running on target DSP processors to execute C standard I/O functions locally or interactively with adjoining host computer systems.
- SPOX-MATH is a comprehensive library of math functions that manipulate standard DSP data objects: vector, matrix, and filter.
- SPOX ’C30/’C40 Math Source is the source code to the SPOX-MATH library of math functions written in TMS320C30/C40 assembly language. Complete documentation is provided to facilitate customizing or generating your own SPOX-compatible ’C30/’C40 math functions in assembly language.

3.6.2 SPOX Enhancements for the TMS320C4x

For the TMS320C4x DSPs, SPOX has been enhanced with parallel-processing support and DMA and communication port drivers. SPOX provides a complete set of parallel-processing primitives and offers a rich core of software functions for use within a TMS320C40 parallel DSP system.

3.6.3 Host Development Software Packages

The SPOX Application Library is available for IBM PCs and Sun workstations to provide developers a low-cost and easy environment for writing SPOX applications. Some of its features include:

- The SPOX application library is a low-cost comprehensive programming interface that serves as an easy environment for writing portable DSP applications in a high-level language like C. With a SPOX host-development software package, complete DSP applications can be written and executed on an IBM PC, a compatible, or a Sun workstation without using DSP hardware. Since the SPOX application library is common across different hardware platforms, SPOX application programs recompile and execute on a variety of DSP targets, including off-the-shelf DSP boards or your custom DSP system. The SPOX library is available in two versions:
  - SPOX/DOS runs on an IBM PC, PC-XT, PC-AT or compatible computer under DOS
  - SPOX-Sun runs on a SUN-3 or Sun-4 workstation under SunOS
- SPOX is currently packaged with the TMS320C3x XDS1000 (refer to Section 4.4) and is available on many third-party products (refer to the *TMS320 Third-Party Support Reference Guide*).

3.6.4 SPOX DBUG

SPOX DBUG is a software package that extends the capabilities of TI’s powerful TMS320C3x C source debugger with features specifically designed to sim-

plify the development of real-time multitasking SPOX OS applications. It allows you to perform the following debug and profile functions.

- **Display SPOX-OS objects**. Interactive display of the status of any SPOX-OS object—including tasks, streams, and arrays—allows you to fine tune the application to optimize system resources and to identify the source of application errors quickly.
- **Set task-specific breakpoints**. Setting breakpoints in a multitasking environment where more than one task can execute the same reentrant function stops execution when any task—not just the task being debugged—reaches the breakpoint. SPOX DBUG solves this problem by allowing you to specify breakpoints that apply only to a particular task.
- **Monitor and display system-performance characteristics**. SPOX DBUG complements the instruction-cycle counting capability of the Texas Instruments C debugger by providing overall system-performance measurements, such as total CPU load. This feature can be used to evaluate a system's capability for performing additional computation or future enhancements.
- **Invoke SPOX-OS system calls**. Interactive execution of SPOX-OS system calls enables you to experiment with the capabilities of SPOX-OS by executing arbitrary system calls and displaying their results. This same capability can be used to force certain rare application execution sequences, thereby verifying the correctness of the application and enhancing its reliability.

Chapter 4

System Integration and Debugging Tools

The system integration and debugging tools for the TMS320 family of digital signal processors include the TMS320 Programmer's Interface (C/assembly source debugger), simulators, standard evaluation modules (EVMs), analog interface board (AIB2), emulators (XDSs—extended development systems), and TMS320 XDS/22 upgrade packages. These tools are described in the following sections.

Topic **Page**

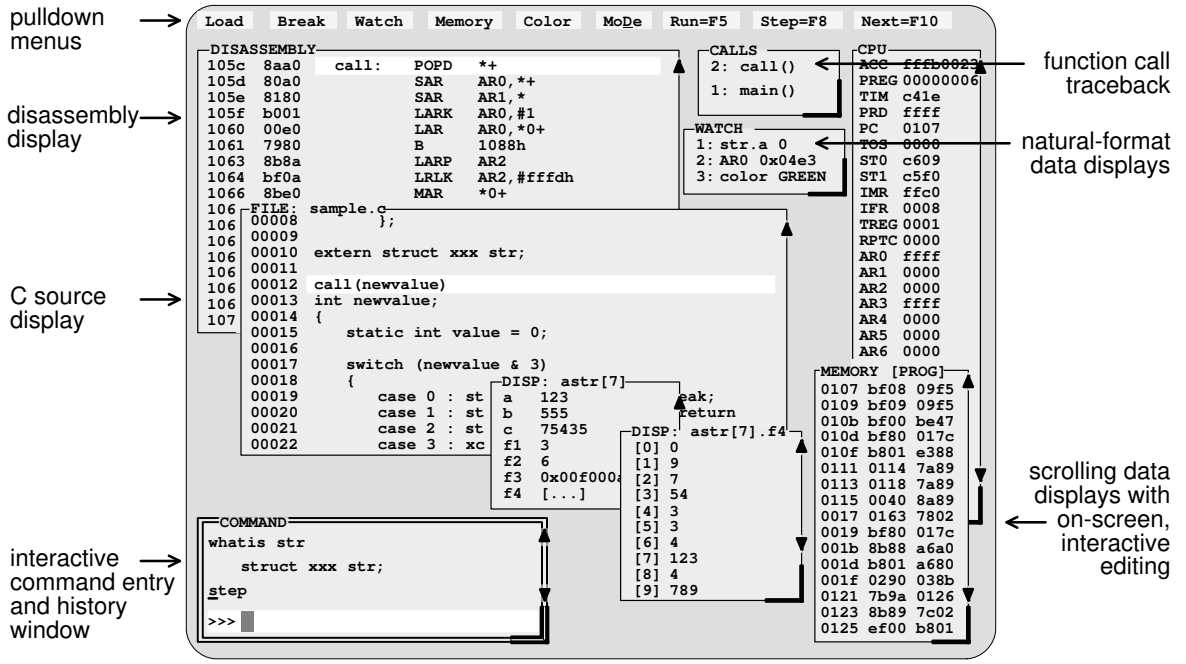
4.1 TMS320 Programmer's Interface (C/Assembly Source Debugger)

The TMS320 programmer's interface brings new levels of power and flexibility to embedded systems development. The interface/debugger is now available for the TMS320C16 EVM, and all TMS320C2x, TMS320C3x, TMS320C4x, and TMS320C5x tools.

The debugger is an advanced software interface that runs on a PC and works with existing TI debugging tools such as the unique, scan-based, realtime, TMS320 emulator, the XDS510. The debugger provides complete control over programs written in C or assembly language.

The debugger improves productivity by enabling you to debugging 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.

Figure 4–1. Debugger's Customized Display



The debugger is easy to learn and use. Its window-/mouse-/menu-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 system's needs

(see Figure 4–1). A shortened learning curve and increased productivity reduce the software development cycle, speeding products to market.

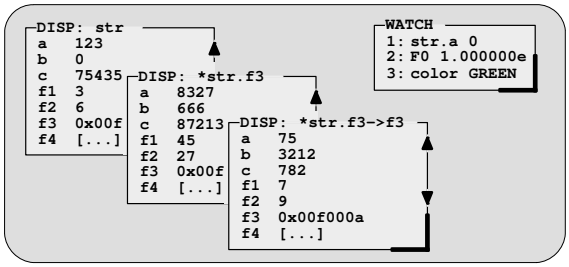
4.1.1 Debugger Features

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 TMS320C40 and TMS320C5x, the C/assembly debugger has been enhanced with special parallel-processing capabilities (multiple-processor debug/breakpoint and single-step).
- **Multilevel debugging**. The debugger allows you to debugging 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, state-of-the-art, 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. The display's colors, physical appearance of displayed features (such as window borders), and window size and position can be changed.
- **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 re-enter commands.
- **On-screen editing**. Any data value displayed in any window can easily be changed 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 displaying and editing the values of variables, arrays, structures, pointers — any kind of data — in their natural format (float, int, char, enum, or pointer). Entire linked lists can be displayed (see Figure 4–2).

- **Powerful command set**. The TMS320 debugger supports a small but 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.
- **Simplicity**. The debugger's simple setup, default configurations, "canned" commands, and inherent flexibility facilitate sophisticated debugging within a short time.

Figure 4–2. Debugger's Data Display



4.1.2 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 the programmer where to focus their 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:

- **Elegant user interface**. The TI code profiler shares the same fully configurable, window-oriented, and mouse-driven 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 simultaneously.
- **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 ability to choose profile areas, the type of statistical data, and sorting criteria ensures an efficient, customized display of the statistics. The data can also 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.

Profiling capability is available in the TMS320C3x, TMS320C4x, and TMS320C5x C source debuggers and simulators. Profiling is supported in versions that run on Windows, OS/2, and SunOS operating systems.

4.2 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 nonrealtime. With the inexpensive software simulator, you can debugging 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 (VAX/VMS) or hundreds of instructions per second (IBM PC under PC-DOS). TMS320 simulators are available for the following devices:

- TMS320C1x
- TMS320C2x
- TMS320C3x
- TMS320C4x
- TMS320C5x

4.2.1 Simulator Features

Key features common to all 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 (TMS320C25)

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 for the TMS320C1x/C2x/C5x simulators 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 re-executed to regain the same machine state during another simulation session.

4.2.2 TMS320C1x Simulator

The TMS320C1x software simulator supports all devices in the TMS320C1x generation and offers the same common features as the TMS320C2x and TMS320C5x simulators (refer to subsections 4.2.1, 4.2.3, and 4.2.6). In addition, the TMS320C1x simulator has an I/O file associated with eight ports.

The TMS320C1x simulator is available for IBM PC-DOS/MS-DOS (5.25-inch floppy) and VAX/VMS (in backup format on 1600-bpi magnetic tape) operating systems. The PC configuration requires a minimum of 256K-bytes of RAM for the TMS320C1x simulator.

4.2.3 TMS320C2x Simulator

The TMS320C2x simulator's software program simulates the operation of the TMS320C2x generation of DSPs. The simulator has commands that specify wait cycles for external data, program, and I/O memory, resulting in a more

flexible, accurate timing analysis for interrupt generation at user-specified intervals. The simulator uses a TMS320 programmer's interface (described in Section 5.1) that allows you to debugging code in C, assembly, or both. Key features of the TMS320C2x simulator include:

- ❑ Simulation of the TMS320C2x digital signal processor's entire instruction set
- ❑ Simulation of the TMS320C2x peripheral's key features
- ❑ Command entry from either menu-driven keystrokes (menu mode) or line mode
- ❑ Help menus for all screen-displayed modes
- ❑ Interface that can be user-customized
- ❑ 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
- ❑ Execution modes
  - ■ Single/multiple instruction count
  - ■ Single/multiple cycle count
  - ■ Until condition is met
  - ■ While condition exists
  - ■ Unrestricted run with halt by keyed input
- ❑ Trace execution with display choices
  - ■ Designated expression values
- ❑ 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
- ❑ File-associated I/O with 16 ports
- ❑ Programmable breakpoints on:

- Instruction acquisition
- Memory reads and writes (data or program)
- Data patterns on the data bus or the program bus
- Error conditions

❑ Commands to modify and inspect registers as well as individual locations in a register

The TMS320C2x simulator is available for IBM PC-DOS/MS-DOS (5.25-inch floppy), VAX/VMS (in backup format on 1600-bpi magnetic tape), and Sun-3/4 UNIX (in TAR format on 1600-bpi magnetic tape) operating systems. The PC configuration requires a minimum of 640K-bytes of RAM for the TMS320C2x simulator.

4.2.4 TMS320C3x Simulator

The TMS320C3x simulator's software program simulates the operation of the TMS320C3x 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 (described in Section 5.1), allowing the user to debugging code in C, assembly, or both. Key features of the TMS320C3x simulator include:

❑ Simulation of the TMS320C3x digital signal processor's entire instruction set

❑ Simulation of the TMS320C3x peripheral's key features

❑ Command entry from either menu-driven keystrokes (menu mode) or line mode

❑ Help menus for all screen-displayed modes

❑ Simulation of external interrupts to the TMS320C3x.

❑ Interface that can be user-customized

❑ 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 32-bit values
- Assembled source

❑ Execution modes

- Single/multiple instruction count
- Single/multiple cycle count
- Until condition is met
- While condition exists
- For set loop count
- Unrestricted run with halt by keyed input

- Trace execution with display choices
  - Designated expression values
  - Cache memory
  - Instruction pipeline
- Simulation of cache utilization
- 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

The simulator lets you verify and monitor the state of the processor. Simulation speed is on the order either of thousands of instructions per second (VAX VMS and SUN-3 UNIX) or of hundreds of instructions per second (PC-DOS/MS-DOS).

The TMS320C3x simulator is available for IBM PC-DOS/MS-DOS (5.25-inch floppy), VAX/VMS (in backup format on 1600-bpi magnetic tape), and the SUN-3/4 UNIX (in TAR format on 1600-bpi magnetic tape) operating systems. The PC configuration requires a minimum of 512K bytes for the TMS320C3x simulator.

4.2.5 TMS320C4x Simulator

The TMS320C4x software simulator uses the TI Programmer'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 high-level language debug. Key features of the TMS320C4x simulator include:

- Simulation of the TMS320C4x digital signal processor's entire instruction set
- Simulation of the TMS320C4x peripherals' key features
- Completely configurable high-level language debugging interface (refer to Section 4.1)

- Command entry from either menu-driven keystrokes (menu mode) or line mode
- Help menus for all screen-displayed modes
- 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 same time) as
  - Hexadecimal 32-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
  - Unrestricted run with halt by keyed input
- Trace execution with display choices
  - Designated expression values
  - Cache memory
  - Instruction pipeline
- Simulation of cache utilization
- 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

The simulator lets you verify and monitor the state of the processor without additional hardware. The TMS320C4x software simulator operates on IBM PCs and Sun-3/4 machines for:

- Modeling of the TMS320C4x's addressable memory
- Modeling of the TMS320C4x's additional features
- Operation of the programmer's interface

4.2.6 TMS320C5x Simulator

The TMS320C5x software simulator also uses the TI programmer's interface. Key features of the TMS320C5x simulator include:

- Simulation of the entire TMS320C5x instruction set
- Simulation of the TMS320C5x peripherals' key features (serial port and timer)
- High-level language debugging interface (refer to Section 4.1)
- Simulation parameters quickly stored/retrieved from files to facilitate preparation for individual sessions
- Reverse assembly of source assembly code, C code, or both (this allows editing and reassembling of 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
  - Unrestricted run with halt by keyed input
- Trace execution with display choices
  - Designated expression values
  - Instruction pipeline for easy optimization of code
- Cycle counting
  - Display of the number of clock cycles in a single-step operation or in the run mode

The simulator lets you verify and monitor the state of the processor without having to install additional hardware. The TMS320C5x software simulator operates on IBM PCs, VAX/VMS, and Sun-3/Sun-4 UNIX machines for:

- Modeling of the TMS320C5x's addressable memory
- Modeling of the TMS320C5x's additional features
- Operation of the debugging interface

4.3 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
- XDS—eXtended Development Support emulators
- Behavioral models

4.3.1 DSP Starter Kit (DSK)

The TI TMS320 DSP Starter Kit (DSK) is an ideal low-cost tool for first-time users interested in evaluating a DSP platform. Available for the TMS320C2x and TMS320C5x DSPs, the DSK allows you to experiment with and use a DSP for real-time signal processing. The DSK allows users to write and run real-time source code, evaluate that code, and debug their 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:

- RS-232 serial port for communicating with your PC
- 2.1-mm jack that allows you to attach a simple wall-mounted AC transformer as your power supply
- Two standard RCA jacks (for analog I/O) that provide direct connections to a microphone, speaker, or other analog devices
- An on-board EPROM that allows the DSK to communicate with your PC, (TMS320C5x only)
- 1.5K words of on-chip RAM on the 40MHz TMS320C2x board, or 10K of on-chip RAM words on the 40MHz TMS320C5x board
- 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 debugging assembly code
- A small command set that reduces the amount of instructions the user must learn

The TMS320 DSKs run on PC-AT with MS-DOS or PC-DOS (version 4.01 and later).

4.3.2 Evaluation Modules

TMS320 evaluation modules (EVMs) are low-cost development boards used for device evaluation, benchmarking, and limited system debug. EVMs are available for the TMS320C16, TMS320C2x, TMS320C3x, and TMS320C5x.

Common TMS320 EVM features include:

- Modification/display of memory and registers
- Assembler
- Software single-step and breakpoint capabilities
- On-board memory
- Host upload/download capabilities
- I/O capability

4.3.2.1 TMS320C16 EVM

The TMS320C16 Evaluation Module (EVM) is a low-cost, PC-AT plug-in card that lets you evaluate certain characteristics of the TMS320C1x DSPs to see if they meet the requirements of an application. The TMS320C16 EVM carries a TMS320C16 DSP on board to allow full-speed verification of your TMS320C1x code. The TMS320C16 is a speed and memory superset of the ’C10 and the ’C15, and its peripherals are identical to the peripherals of both of those devices. Therefore, the ’C16 EVM can be used to evaluate the suitability of the ’C16, the ’C15, or the ’C10 for a given application.

Other key features of the TMS320C16 EVM include:

- 32-MHz operation
- Single-step and breakpoints
- 64K external program SRAM on board
- Easy-to-use, windows-oriented, mouse-driven user interface that features windows displaying CPU registers, memory locations, disassembled code, and other variables needed for debug

- ❑ TMS320 fixed-point assembler/linker
- ❑ Optional analog interface board available (interfaced via external connector)

4.3.2.2 TMS320C2x EVM

The TMS320C2x EVM carries a TMS320C26 DSP on-board to allow full-speed verification of TMS320C2x code. The TMS320C26 has 1568 words of on-chip RAM; it is pin compatible to the ’C25 , and it’s peripherals are identical to the peripherals of the ’C25. Therefore, the ’C2x EVM can be used to evaluate the suitability of the ’C26 and ’C25 for a given application.

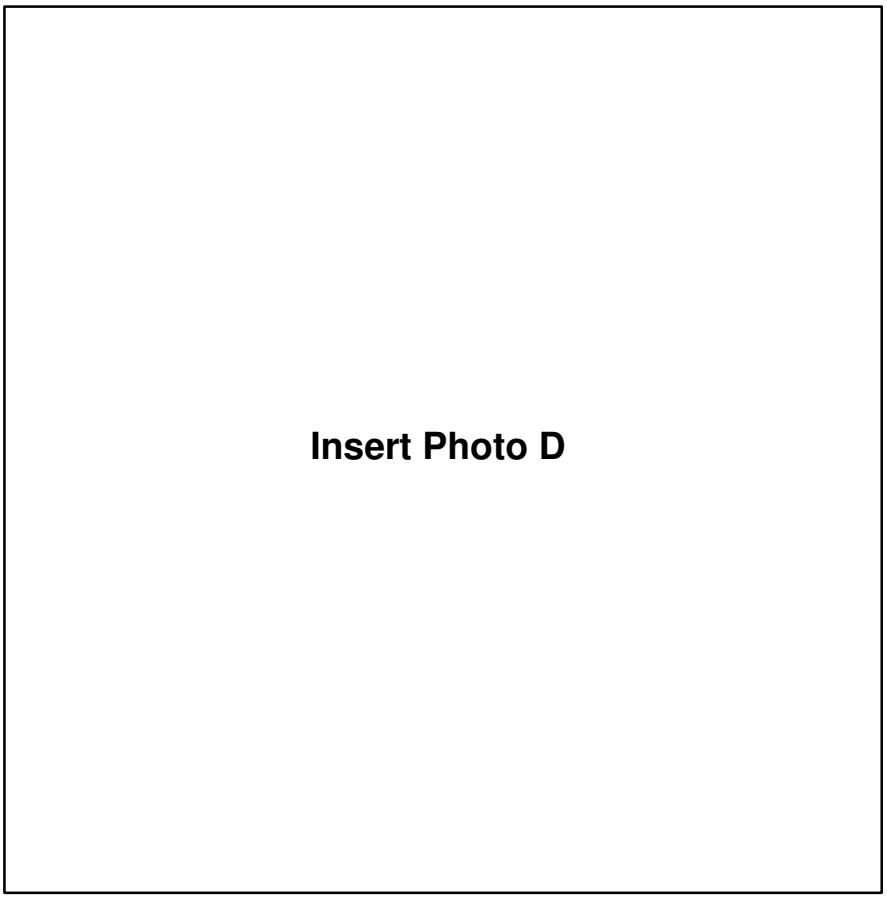
Key features of the ’C2x EVM include:

- ❑ 80 ns instruction cycle time
- ❑ 64K external zero wait-state SRAM on-board
- ❑ Voice-quality analog data acquisition via the TLC32046 (analog interface circuit)
- ❑ Standard RCA analog input and output connectors for direct connection to microphone and speaker
- ❑ External TDM serial port
- ❑ 16-bit bidirectional PC host communications port
- ❑ I/O expansion bus for application use
- ❑ Onboard emulation logic for source debugger support
- ❑ IBM PC compatible, 16-bit half card, mappable in one of four memory locations.

4.3.2.3 TMS320C3x EVM

The TMS320C3x evaluation module (see Figure 4–3) eliminates the cost barrier to evaluating and developing floating-point DSP applications. The TMS320C3x EVM is the first floating-point DSP tool that bridges the price/performance gap between software simulators and expensive development platforms.

Figure 4–3. TMS320 EVMs



The TMS320C3x EVM enables you to benchmark and evaluate code in real-time while the device is operating at 30 MHz in the rich development environment of the TMS320C3x assembler/linker and C/assembly source debugger interface. Applications can be benchmarked and tested easily with the analog-ready interface.

The TMS320C3x EVM comes complete with a PC half-card and software package. The EVM board contains:

- One TMS320C30 — a 33-MFLOPss, 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 via 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-driven interface supports downloading, executing, and debugging of assembly code or C code.
- The TMS320C3x assembler/linker is also included with the EVM. For high-level language programming, the optimizing ANSI C and the Ada compilers are offered separately.

The TMS320C3x EVM is supported on PC-AT/MS-DOS (version 3.00 or higher) platforms.

4.3.2.4 TMS320C5x EVM

The TMS320C5x EVM carries a TMS320C50 DSP on-board to allow full-speed verification of TMS320C5x code. The TMS320C50 has 10K on-chip ROM; it is pin compatible with the ’C51 and ’C53, and it’s 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 the ’C53 for a given application.

Key features of the TMS320C5x EVM include:

- 50 ns instruction cycle time
- 64K external zero wait-state SRAM on board
- Voice quality analog data acquisition via the TLC32046 (analog interface circuit)
- Standard RCA connector analog input and output for direct connections to microphone and speaker
- Embedded emulation support via 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

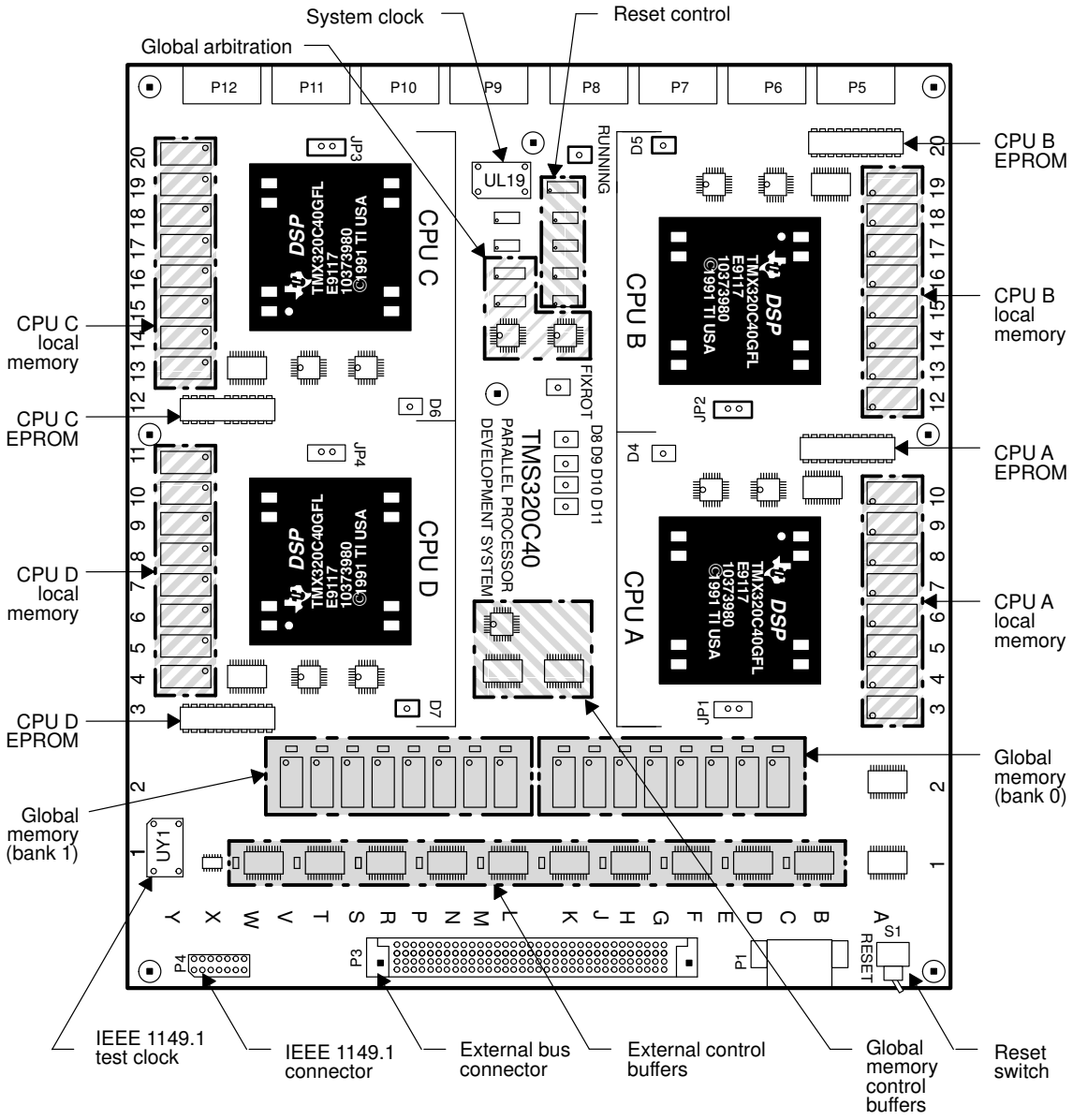
4.3.3 TMS320C4x Parallel Processing Development System

The TMS320C40 Parallel Processing Development System (PPDS) 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 realtime in a rich development environment with the power and speed of the TMS320C40 PPDS.

Key features of the PPDS include:

- Four on-board TMS320C40 parallel processors. Each TMS320C40 is supported by a local bus consisting of:
  - 64K×32-bit words of zero wait-state static RAM (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 TMS320C40s and external peripherals to the PPDS's TMS320C40s.
- An IEEE Std 1149.1–1990-compliant (JTAG) test connector that serves as an interface for connecting the XDS510 in-system emulator

Figure 4–4. The TMS320C40 PPDS Board Layout



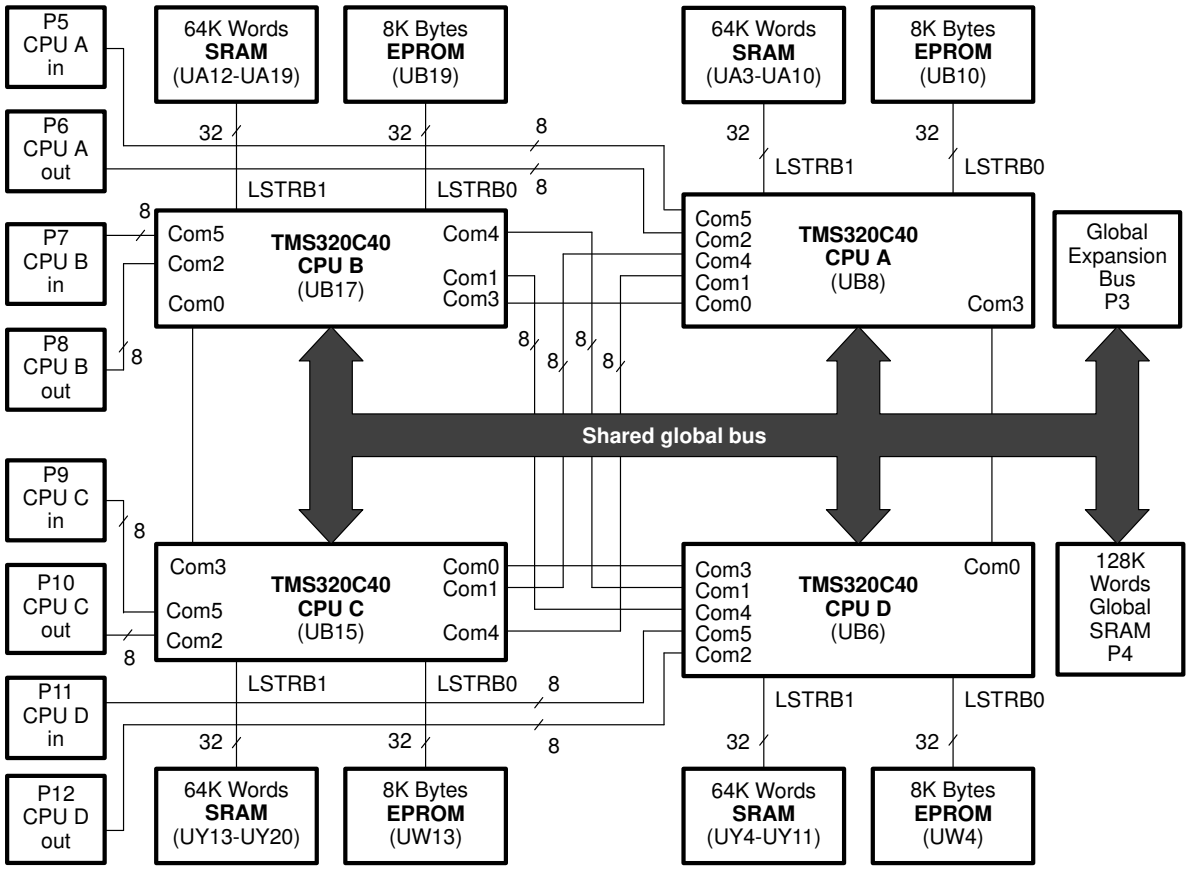
The PPDS is placed on the desktop and is controlled through the XDS510, available separately. The PPDS is shipped with a dedicated desktop stand and its own 20-amp, 50-watt power supply. You also need a C compiler.

Each TMS320C40 on the PPDS has direct connections to each of the other TMS320C40s 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 TMS320C40 also has two communication ports pinned out to external connectors on the left edge of the board, allowing other TMS320C40-based boards or peripheral boards to be connected to the TMS320C40s on the PPDS.

The TMS320C40s are also connected on a shared bus that has arbitration logic that decides which TMS320C40 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.

Figure 4–5. TMS320C40 PPDS Block Diagram



Even though the PPDS is used with the XDS510, each TMS320C40 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 debugging multiprocessing algorithms.

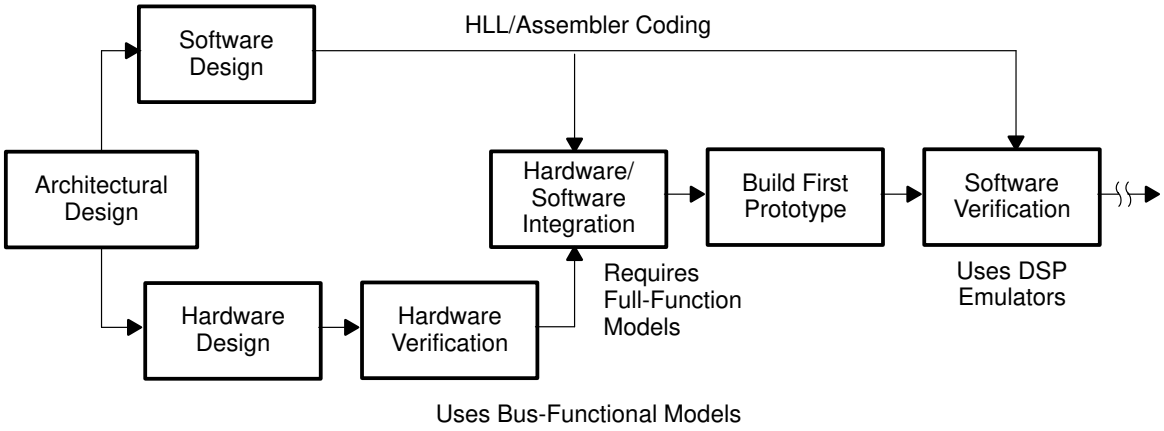
4.3.4 Behavioral Models

TMS320 behavioral models concentrate on the external behavior of DSP devices and their electrical timing from a “pin-out” perspective. Instead of building a prototype, you can use software to evaluate a design. Entire subsystems can be simulated quickly on an engineering workstation, drastically reducing the cycle time and the cost of developing board-level systems. The models also clarify and help manage the increasing complexity of DSP systems (see Figure 4–6).

Two types of DSP behavioral models are available: bus-functional and full-functional. Bus-functional models simulate DSPs by executing bus cycles that are under control of a C-like language, exercising the system through the DSP’s pins. Full-functional models exercise the full instruction set of the DSP by executing object code stored in memory. Bus-functional and full-functional models have equally accurate timing.

TI and Logic Automation Incorporated (LAI) have jointly developed behavioral models of TI semiconductor devices. Models run on most major simulators, including those from Mentor Graphics Corporation, Valid Logic Systems, HHV Systems, Inc., Hewlett Packard, Vantage Analysis Systems, and Gateway Design Automation Corp.

Figure 4–6. Behavioral Models in DSP Development



4.4 TMS320 Emulators

The TMS320 Extended Development Systems (XDSs) are powerful, full-speed emulators used for system-level integration and debug. TI provides conventional in-circuit emulators as well as the world's first in-system scan-based emulators (XDS/22 and XDS510). The XDS510 emulator is currently available for TMS320C3x DSPs, TMS320C4x DSPs, and TMS320C5x DSPs. The XDS/22 supports TMS320C1x (except TMS320C16) and TMS320C2x DSPs.

4.4.1 Scan-Based Emulators

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/software characteristics (reduced internal bus visibility, highly pipelined architectures, faster cycle times, higher-density packaging) that are inherent to sophisticated 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 realtime 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.

4.4.1.1 TMS320C3x, TMS320C4x, and TMS320C5x XDS510 Emulators

The TMS320C3x, TMS320C4x, and TMS320C5x emulators (XDS510) are user-friendly, PC-based development systems, which have all of the features necessary to perform full-speed, in-circuit emulation with the 'C3x, 'C4x and 'C5x DSPs. They are the first in-circuit 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 5-wire interface acts as a scan path to every memory and register location in the DSP device. Refer to Figure 4–7.

Key features of the XDS510 include:

- Full-speed execution and monitoring of the device within your target system via a 14-pin target connector
- Global run/stop/breakpoint of parallel-processing DSPs
- High-level language 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
- Windowed user interface similar to that of the ’C3x, ’C4x, or the ’C5x simulator
- Loading/inspecting/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 via a 14-wire cable, which runs from the XDS510 to the target system. The IEEE 1149.1 scan path controls the device within the targeted application and provides access to all of the registers as well as to 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 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 halts 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 are 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 Programmer's Interface for fast, easy debugging of C and assembly source code.

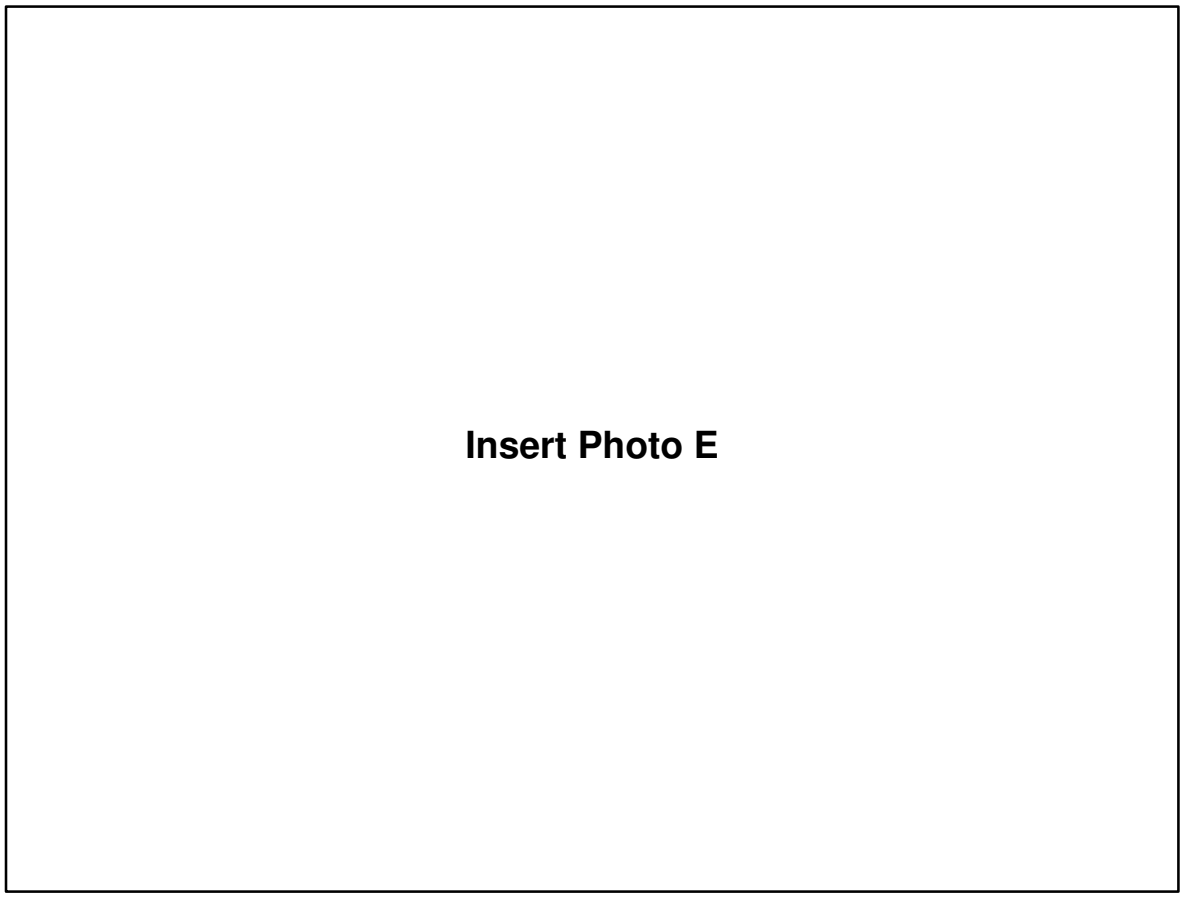
The emulator's configurability gives your system flexibility. You can configure both memory and screen color. The address range, memory type, and access type assigned to each location can also be configured. The memory map, which may include EPROM, SRAM, DRAM, and on-chip memory and peripherals, can be configured to reflect the actual peripheral environment of the target system, including wait states and access privileges.

The TMS320C3x, TMS320C4x, and TMS320C5x XDS510 emulator packages include:

- ❑ TMS320C3x, TMS320C4x, or TMS320C5x emulator PC board
- ❑ TMS320C3x, TMS320C4x, or TMS320C5x 5-wire interface cable with 14-pin connector
- ❑ TMS320C3x, TMS320C4x, or TMS320C5x user interface software (5.25-inch floppy)

The XDS510 emulators operate on PC-AT systems running PC-DOS/MS-DOS, Windows 3.0, and OS/2 and require one 16-bit slot.

Figure 4–7. XDS510 Scan-Based Emulators



4.4.2 XDS/22 Emulators

There are currently four different versions of the XDS/22: three for the TMS320C1x and one for the TMS320C2x. Differences between the versions are covered in the individual descriptions later in this section. The many similarities are discussed in the general descriptions in the four following subsections.

4.4.2.1 TMS320 XDS/22 System Configurations

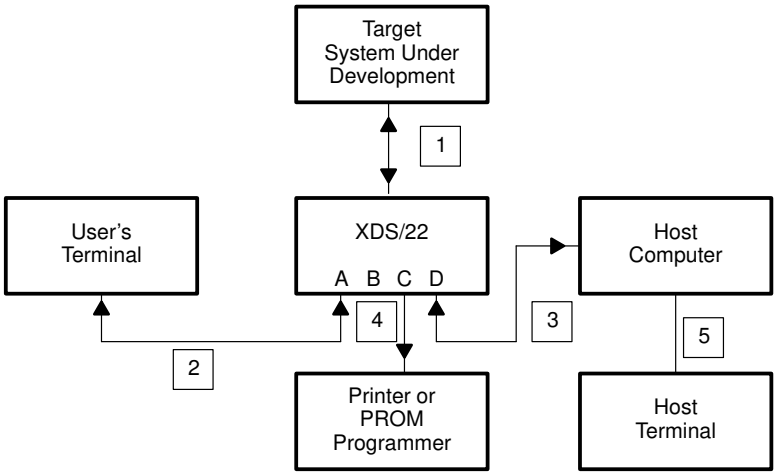
The XDS/22 can be configured to operate in one of four modes:

- Stand-alone mode
- Host computer mode
- PC mode (single-user system)

- Multiprocessor mode

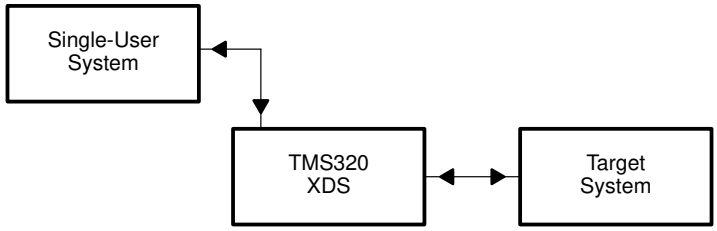
The stand-alone mode or minimal configuration requires only the XDS/22 and your terminal. However, the XDS/22 is best used with a host computer (see Figure 4–8), where TMS320 programs can be written on a familiar editor, assembled, and then downloaded into the XDS/22. Once a debugging session is complete, TMS320 code can be uploaded to the host computer for storage.

Figure 4–8. TMS320 XDS/22 Host Computer Mode



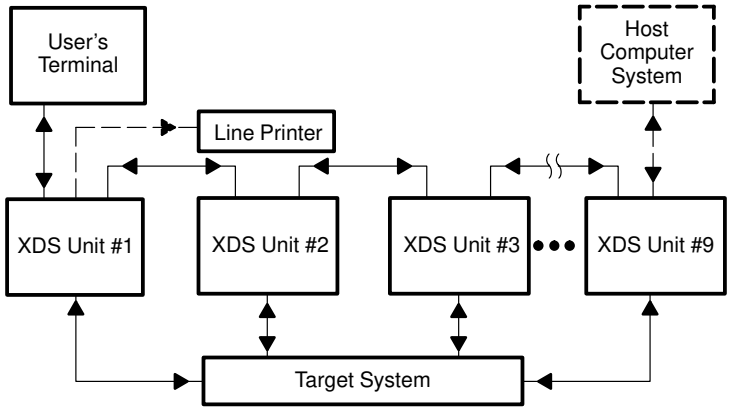
In the PC mode, the TMS320 XDS/22 can support host uploads and downloads over a single port to allow a single-user system, such as an IBM's PC or compatible, to function as both a terminal and a host when connected to the XDS (see Figure 4–9). Terminal emulation software for the single-user system is required in this configuration. Communications software packages are commercially available, which allow PC-AT or compatible to function as both a terminal and a host for the XDS/22.

Figure 4–9. TMS320 XDS/22 Single-User System



The emulator's multiprocessor mode allows up to nine XDS/22s to be connected together in a daisy-chain fashion and controlled by a single terminal, as shown in Figure 4–10. A single host computer can also be connected.

Figure 4–10. TMS320 XDS/22 Multiprocessor Mode



TMS320 XDS/22 Communication

The TMS320 XDS/22 emulator provides communication links to standard RS-232C ports and also has debugging capabilities with a prompting XDS monitor and full-speed hardware breakpoints and trace. The communications system establishes linkage with the user's terminal, a PROM programmer or printer, and a host computer system. The functions of this communication link are:

- To transmit data files from the emulator to an external device (upload)
- To receive data files from an external device and store them into the emulator's memory (download)
- To pass downloaded data received from an external device to a PROM programmer or logging device
- To transmit data stored in the emulator's memory to a PROM programmer or logging device

Each TMS320 XDS/22 unit is equipped with four standard RS-232C ports for communications with external devices. Only three ports are used; the fourth port is reserved for future expansion.

TMS320 XDS/22 Debugging

The TMS320 XDS/22 monitor features a simple, yet powerful, set of commands for full debugging of the target system. The monitor-type commands provide complete control of both the emulator 's functions and the target system. The monitor uses extensive prompt menus for commands and parameter

definition. You can access registers by using variable names assigned to each. Debugging sessions can also be logged for further analysis via a line printer.

Emulator commands offer flexibility in defining the test conditions for emulation sessions. Commands can be combined in a variety of ways to form short procedures that allow several commands to be executed sequentially. Repeat functions allow procedures or individual commands to be executed indefinitely until you stop them or until they reach a user-defined breakpoint condition.

The XDS/22 supports breakpoint, trace, and time-stamping (BTT) operations. Up to ten software breakpoints and four sequential hardware breakpoints may be defined. This provides a method for testing and debugging small segments of programs. In the monitor mode, all registers and memory locations can be inspected and modified.

The XDS/22 emulators support full-speed trace capability. Each traceable machine cycle is sampled, recorded, and stored in the 2K-word trace buffer so that it can later be recalled for display or printing.

Time-stamping is a feature in which a time value is associated with each trace sample so that the time between breakpoints can be calculated. This lets you determine the amount of time spent in a certain portion (e.g., a loop) of code.

TMS320 XDS/22 Equipment List

The following items are required for use with the TMS320 emulator:

Terminal	**Description**
RS-232C compatible	25-pin RS-232C DB25P connection
Cables	
For terminal/host or printer	Two standard RS-232C cables with male connectors on the XDS end, one nine-wire cable for connection between IBM PCs, and a 9-pin-to-25-pin adapter supplied with the XDS/22. The adapter is used to connect between the standard RS-232C cable and the nine-wire cable.
Host Computer	
(Optional)	Data-terminal communications program connected to an RS-232C port

4.4.2.2 TMS320C10/C15 Emulator

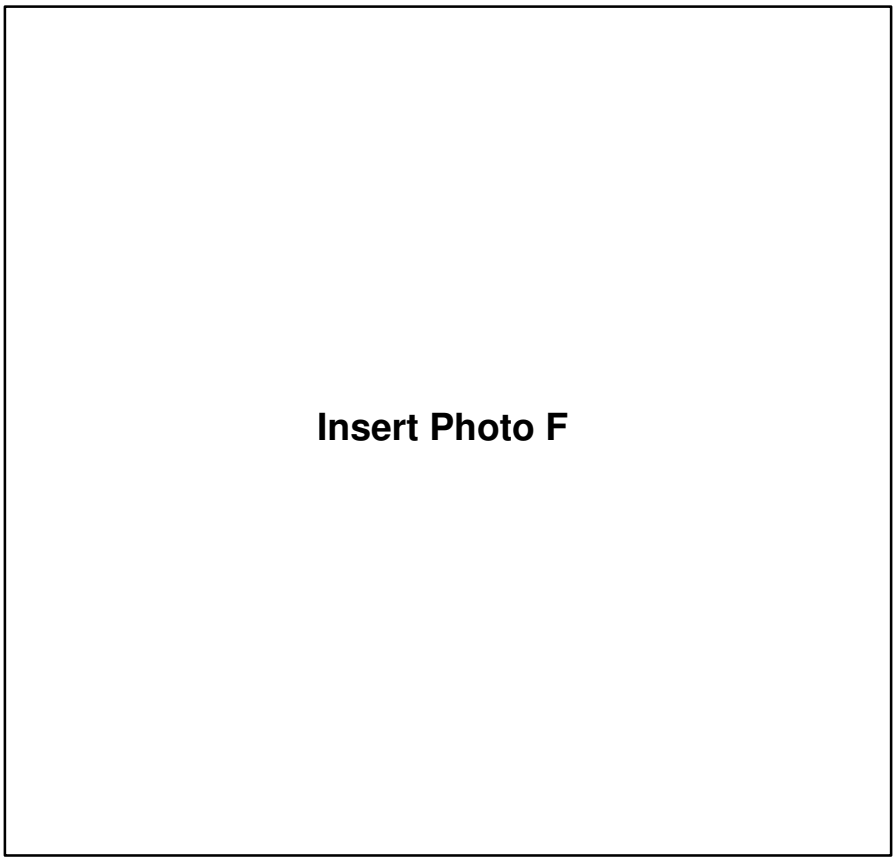
The TMS320C10/C15 emulator (XDS/22) has been designed to emulate operation of the following TMS320C1x devices: TMS320C10, TMS320C10-14,

TMS320C10-25, TMS320C15/E15, TMS320C15-25, and TMS320E15-25 (see Figure 4–11 on page 4-30). Other models are available to support the TMS320C14 and TMS320C17. To emulate any one of these devices requires the insertion of the appropriate device into the emulator board.

These items are packaged with the TMS320C10/C15 emulator:

- XDS/22 chassis
- Emulator board—TMS320C1x base and the TMS320C10/C15 processor module (PM)—with target connector cable
- Breakpoint, trace, and timing board with logic analyzer interface cable
- Communications board
- RS-232C cable for connection between the XDS and either a PC or a stand-alone monitor
- Trace probe cable to connect the breakpoint, trace, and timing board to the target system

Figure 4–11. TMS320C10/C15 Emulator (XDS/22)



The TMS320C10/C15 emulator board contains 4K words of fast static RAM. This allows for operation in one of three memory modes:

- Software development mode — 4K words reside entirely within the emulator
- Microcomputer mode — on the TMS320C10, 1.5K words of program memory reside within the emulator, and 2.5K words of program memory reside within the target system; on the TMS320C15, the mode is identical to the software development mode, and all 4K words of program memory reside within the emulator
- Microprocessor mode — 4K words of program memory reside entirely within the target system

The breakpoint, trace, and timing board (BTT) monitors various hardware activities. It can be programmed to take various actions triggered by the occur-

rence of specified qualifiers, depending on the state of the board. This allows multilevel or sequenced breakpoints to be used for complex debugging solutions. The BTT board's trace sample function offers "snapshot" storage of bus cycle activity. Up to 2,047 samples can be stored in a circular trace buffer. Timing statistics are provided, thus facilitating your analysis of a program's performance by displaying either the actual execution time of a particular routine or the time spent accessing selected memory locations. The BTT has the option of stopping an application program after collecting a selected number of cycles.

The communications board receives all keyboard or host computer input and generates RS-232C-compatible output signals for the host computer, printer, EPROM programmer, and screen display.

These key features distinguish the TMS320C10/C15 emulator:

- 25-MHz, full-speed, in-circuit emulation
- Dual in-line target connector with optional PLCC target connector
- Software development mode (4K words reside entirely within the emulator)
- Microcomputer mode (on the TMS320C10, 1.5K words of program memory reside within the emulator, and 2.5K words of program memory reside within the target system; on the TMS320C15, the mode is identical to the software development mode, and all 4K words of program memory reside within the emulator)
- Microprocessor mode (4K words of program memory reside entirely within the target system)
- Single-step execution
- Line-by-line, reverse, or patch assembler
- Enhanced decimal parameter entry and display
- All levels of stack available to user
- Target system's oscillator (with DIP target connector) or an internal crystal
- Host-independent uploading/downloading of program memory and data memory
- Ability to inspect/modify all internal registers (program/data)
- BTT capabilities with logic analyzer interface cable

- ❑ Logic tracing with extended data/address probes

4.4.2.3 TMS320C14 Emulator (XDS/22)

The TMS320C14 emulator has all the capabilities of the TMS320C10/C15 emulator with the addition of a TMS320C14 processor module (PM) to the TMS320C1x base emulator board. The TMS320C14 emulator performs emulation of all aspects of the TMS320C14/E14 device. Monitor-type commands supported by the TMS320C14 XDS/22 display or modify any of the TMS320C14 I/O registers.

4.4.2.4 TMS320C17 Emulator (XDS/22)

The TMS320C17 emulator has all the capabilities of the TMS320C10/C15 emulator with the addition of a TMS320C17 processor module (PM) to the TMS320C1x base emulator board. The TMS320C17 emulator permits emulation of all aspects of the TMS320C17/E17 device. This includes serial port and coprocessor port operations.

The TMS320C17 emulator functions in either the coprocessor mode or microcomputer mode; all 4K words of memory are available in each mode.

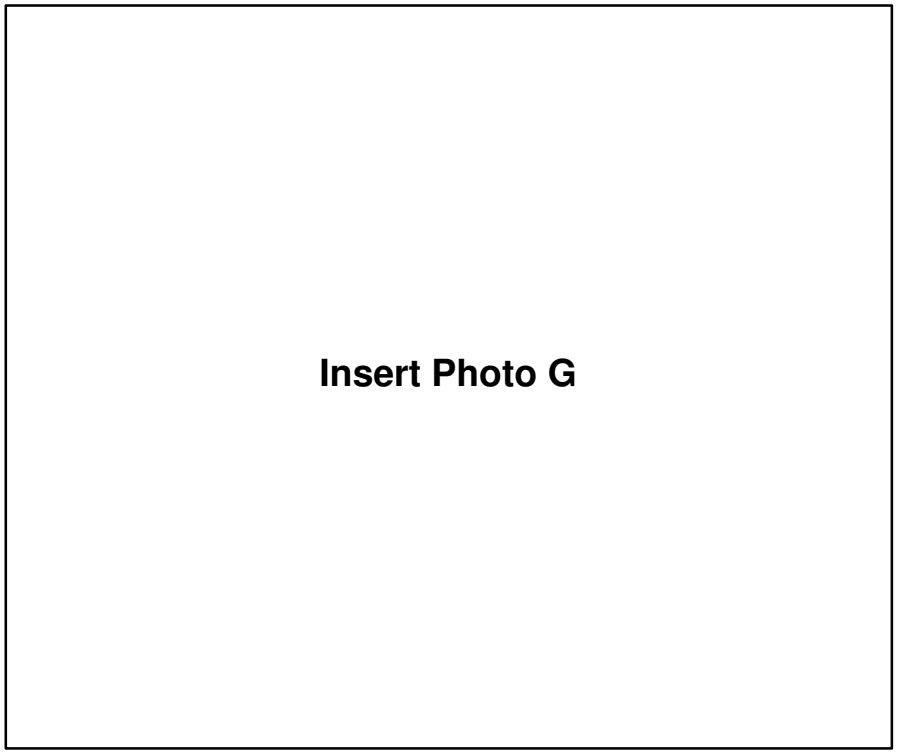
4.4.2.5 TMS320C2x Emulator (XDS/22)

The TMS320C2x emulator (see Figure 4–12) has been designed to emulate operation of the following TMS320C2x generation devices: TMS320C25, TMS320E25, and TMS320C26. To emulate any one of these devices requires the insertion of the appropriate device in the emulator board. The TMS320C2x emulator does not support the TMS320C25-50. Macrochip Research, Inc., a TMS320 third-party supplier, offers full-speed/in-circuit emulation up to 50 MHz to support the TMS320C25-50. Hyperception, another third-party supplier, offers a high-level language debugger package for the TMS320C2x emulator. The debugger interface is the same C/assembly source debugger interface that is found in the newer TI XDS510 emulators for the ’C3x, ’C4x, and ’C5x families. (For more third-party information, please refer to the *TMS320 Third-Party Support Reference Guide*, literature number SPRU052B).

These items are packaged with the TMS320C2x emulator:

- ❑ XDS/22 chassis
- ❑ Emulator board (and appropriate device) with target connector cable
- ❑ Breakpoint, trace, and timing board with a logic analyzer interface cable
- ❑ Memory expansion/communications board
- ❑ RS-232C cable for connection between the XDS and PC
- ❑ Trace probe to connect the breakpoint, trace, and timing board to the target system

Figure 4–12. TMS320C2x Emulator (XDS/22)



The TMS320C2x emulator board contains 8K words (4K-word program and 4K-word data) of high-speed static RAM (if zero wait-state operation) for program and data memory. The memory expansion/communications board offers 64K words of DRAM, which can be configured (with wait states) as all program memory, all data memory, or a combination of both.

The BTT board monitors various hardware activities. It can be programmed to take actions triggered by the occurrence of specified qualifiers, depending upon the state of the board. This makes it possible to use multilevel or sequenced breakpoints for complex debugging solutions. The BTT board's trace sample function offers *snapshot* storage of the bus cycle's activity. Up to 2,047 samples can be stored in a circular trace buffer. Timing statistics are provided, thus facilitating your analysis of a program's performance by displaying either the actual execution time of a particular routine or the time spent accessing selected memory locations. The BTT has the option of stopping an application program after collecting a selected number of cycles.

Key features of the TMS320C2x emulator include:

- 40-MHz, full-speed, nonintrusive, in-circuit emulation
- PLCC target connector with pin-grid-array adapter
- 4K words each of program and data high-speed SRAM memory
- 64K-word DRAM memory expansion/communications board
- Hardware breakpoint, trace, and timing capabilities
- Single-step execution
- Line-by-line, reverse, or patch assembler
- Enhanced decimal parameter entry and display
- Use of target system's CLKIN signal or internal crystal
- Host-independent uploading/downloading of program memory and data memory
- Ability to inspect/modify all internal registers (program/data)
- Multiprocessing support
- Logic tracing with extended data/address probes
- Logic analyzer interface cable

4.5 TMS320 XDS/22 Upgrade Packages

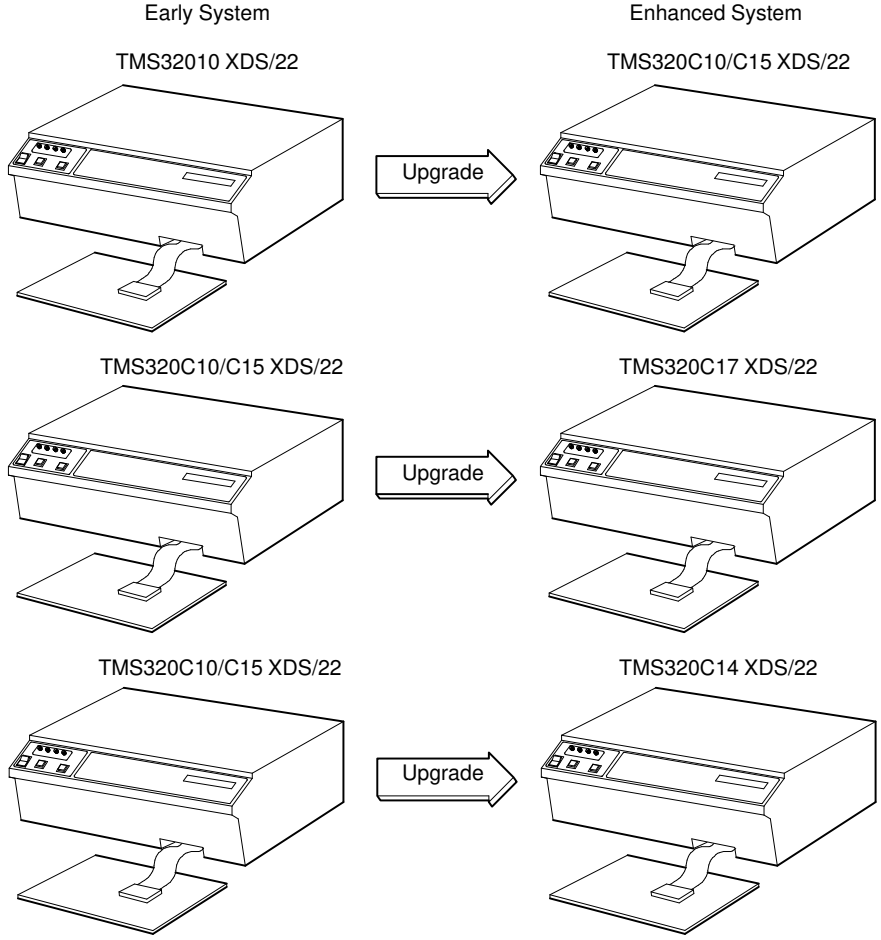
TMS320 XDS upgrade kits extend the functionality of existing development systems at a minimum cost by enhancing the customer's current equipment. For example, an upgrade kit can enable a TMS32010 XDS/22 to emulate operation of all versions of the TMS320C10 and TMS320C15/E15. Upgrade kits are compatible only with devices of the same generation, not between devices of different generations (e.g., a TMS320C1x-generation XDS system cannot be upgraded into a TMS320C2x-generation XDS system.

Figure 4–13 shows the addition of the upgrade kit to the earlier system to give an enhanced-XDS system. Figure 4–13 lists the part numbers for the earlier system, the upgrade kit, and the enhanced system.

Standard upgrade procedures require that the customer make the necessary modifications to the system with the TMS320 XDS upgrade kit. There is a 30-day warranty on upgrades performed by the customer. A board-level test is performed by the factory on each board prior to shipment of the upgrade kit.

Please note that the TI standard warranty does not allow customers to upgrade a system that is still under the 90-day warranty period. According to that warranty, any change or modification to a system still under warranty voids that warranty. Cost for repairs to customer-modified systems will depend upon the factory's current repair prices (see Section B.2 *Nonwarranty Exchange and Repair*, page B-3).

Figure 4–13. XDS Upgrade Configurations



Chapter 5

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**

5.1 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 2000 pages of application notes, and textbooks offered by Prentice-Hall, John Wiley and Sons, and Computer Science Press. The latest product and documentation updates are given in the TMS320 quarterly newsletter *Details on Signal Processing* and the TMS320 DSP Bulletin Board Service (BBS). To inquire about available TMS320 literature, call the TI Literature Response Center:

(800) 477-8924

The following list describes the general contents of each major category of technical documentation available through the TI Literature Response Center:

- Product and preview bulletins and product briefs give an overview of the devices and development support within the TMS320 family, presenting capabilities, diagrams, and hardware/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 electrical specifications, timing characteristics, and mechanical data for each device.
- Application books/reports describe theory and implementation of selected TMS320 applications, including algorithms, code, and block/schematic/logic diagrams. Currently, there are over 2000 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.
- Technology brochures provide an overview of various implementations of DSP technology in applications such as 3-D graphics, modems, and integrated service digital network (ISDN).

5.2 DSP Applications Books

TI engineers continuously write application reports to assist customers in designing TMS320 DSP applications. New application reports are available through the TMS320 DSP Bulletin Board Service (BBS); earlier reports are published in the TMS320 application books. Table 5–1 lists those application subjects that are covered on the BBS and in the application books:

- *Digital Signal Processing Applications With the TMS320 Family,* volume 1 (literature number SPRA012A)
- *Digital Signal Processing Applications With the TMS320 Family,* volume 2 (literature number SPRA016)
- *Digital Signal Processing Applications With the TMS320 Family,* volume 3 (literature number SPRA017). This book is dedicated to floating-point processors.
- *Digital Control Applications With the TMS320 Family* (literature number SPRA019)
- *Parallel Processing With the TMS320C4x Family* (literature number SPRA031)
- *Digital Telecommunication Applications With the TMS320 Family* (literature number SPRA033)

A number of application reports focus on specific DSP issues, including:

- *Calculation of the TMS320C30 Power Dissipation* (literature number SPRA020)
- *DSP Applications With the TMS320C30 EVM* (literature number SPRA021)
- *Implementation of the Kaish Circuit Lockout System* (literature number SPRA022)
- *TMS320C3x DSPs Supercharge 3-D Graphics* (literature number SPRA024)
- *TMS320-SCSI Target Controlled Application Report* (literature number SPRA025)
- *A Parallel Approach for Solving Matrix Multiplication on the TMS320C4x DSP* (literature number SPRA026)
- *TMS320C4x Parallel 2-D FFT Application Report* (literature number SPRA027)

Application books give source code for the application discussed. This code can be used to reduce design time and to bring TMS320-based products to market faster. Source code is also available; refer to Section 5.7 *TMS320 DSP Bulletin Board Service* on page 5-19 for more information.

Table 5–1. Application Reports

Device	Subject	Location†
	Data Communications	
TMS32010	*Theory and Implementation of a Splitband Modem Using the TMS32010*	2
TMS320C17	*An All-Digital Automatic Gain Control*	2
TMS320C17	*Implementation of an FSK Modem Using the TMS320C17*	2
TMS320C5x	*Viterbi Implementation on TMS320C5x for V.32 Modems* (Mansoor Chishtie)	‡
TMS320C5x	*Digital Line Echo Canceller Implementation on TMS320C5x DSP* (Kevin McCoy and Mansoor Chishtie)	‡
	Digital Cellular	
TMS320C5x	*Cellular Phone: A Functional Analysis* (B.I. Pawate and Mansoor Chishtie)	‡
TMS320C5x	*IS-54 Simulation Package* (John Crockett/Steve Popik/ Elliot Hoole)	‡
TMS320C5x	*U.S. Digital Cellular Error Correction Coding Alogorithm Implementation on TMS320C5x* (Mansoor Chishtie)	‡
TMS320C5x	*A Performance Study of two TMS320C53-Based Viterbi Algorithms for U.S. Digital Cellular Radio* (Mansoor Chishtie)	‡
TMS320C5x	*A TMS320C53-Based Advanced FEC Scheme for U.S.D.C. Radio* (Mansoor Chishtie)	‡
TMS320C5x	*IS-54 Digital Cellular Modem Implementation on TMS320C5x* (Balaji Srinivasan)	‡
TMS320C5x	*Mobitex Modem Implementation Using the TMS320C6x* (Etienne Resweber)	‡
TMS320C5x	*Equalization Concepts* (David Smalley)	‡
TMS320C5x	*C5x-Based Equalizer Implementation for IS-54* (Elliot Hoole)	‡
	Digital Control	
TMS320	*Implementation of PID and Deadbeat Controllers With the TMS320 Family*	2
TMS32010	*Control System Compensation and Implementation With the TMS32010*	1

(Continued on the next page)

† The numbers shown in this column refer to the three-volume *Digital Signal Processing With the TMS320 Family*, where **1** = volume 1, **2** = volume 2, and **3** = volume 3.

‡ Refer to *Telecommunications Applications With the TMS320C5x* (literature number SPRA033).

Table 5–1. Application Reports (Continued)

	DSP Interface	
TMS320	*Interfacing the TMS320 Family to the TLC32040 Family*	2†
TMS32010	*Interfacing to Asynchronous Inputs With the TMS32010*	1
TMS32010	*Interfacing External Memory to the TMS32010*	1
TMS320C17	*TMS320C17 and TMS370C0010 Serial Interface*	2
TMS320C2x	*Hardware Interfacing to the TMS320C2x*	2
TMS32020	*Hardware Interfacing to the TMS32020*	1
TMS32020	*TMS32020 and MC68000 Interface*	1
TMS320C25	*I_{CC} Requirements of the TMS320C25*	2
TMS320C25	*An Implementation of a Software UART Using the TMS320C25*	2
TMS320C3x	*Interfacing the TMS320C3x to the TLC3204x Analog Interface Chip*	§
TMS320C30	*TMS320C30 Hardware Applications*	3
TMS320C30	*TMS320C30 – IEEE Floating-Point Format Converter*	3
TMS320C30	*A Low-Cost TMS320C30 Host Interface*	§
TMS320C5x	*A DSP-Based PCMCIA card Design* (Raj Chirayil)	‡

(Continued on the next page)

† The numbers shown in this column refer to the three-volume *Digital Signal Processing With the TMS320 Family*, where **1** = volume 1, **2** = volume 2, and **3** = volume 3.

‡ Refer to *Telecommunications Applications With the TMS320C5x* (literature number SPRA033)

§ Refer to *Selected Application Notes Using the TMS320C30 Evaluation Module* (literature number SPRA021), by Nat Seshan.

Table 5–1. Application Reports (Continued)

	DSP Routines	
TMS320	*Digital Filter Design Programs (FIR/IIR)*	BBS
TMS320C1x	*EDN Magazine 1st-Generation (TMS320C1x) Benchmarks*	BBS
TMS32010	*Floating-Point Arithmetic With the TMS32010*	1
TMS32010	*Precision Digital Sine-Wave Generation With the TMS32010*	1
TMS320C10	*TMS320C10 FFT Routines*	BBS, JW¶
TMS320C14	*TMS320C14 Examples* (from *TMS320C14 User's Guide*)	BBS
TMS320C1x/C2x	*Self Test*	BBS
TMS32010/20	*Companding Routines for the TMS32010/TMS32020*	1
TMS32010/20	*Implementation of FIR/IIR Filters With the TMS32010/ TMS32020*	1
TMS32010/20	*Matrix Multiplication With the TMS32010 and TMS32020*	1
TMS320C2x	*EDN Magazine 2nd-Generation (TMS320C2x) Benchmarks*	BBS
TMS320C2x	*TMS320C2x-Based Adaptive Line Enhancer Design*	BBS
TMS32020	*Floating-Point Arithmetic With the TMS32020*	1
TMS32020	*Implementation of Fast Fourier Transform Algorithms With the TMS32020*	1
TMS320C25	*C25 Complex 256-Point FFT*	BBS
TMS320C25	*C25 Real 256-Point FFT*	BBS
TMS320C25	*TMS320C25 Examples* (from *TMS320C25 User's Guide*)	BBS
TMS320C26	*Preliminary Documentation/Source Listing for C26 Boot ROM*	BBS
TMS320C25/C30	*An 8x8 Discrete Cosine Transform Implementation on the TMS320C25 or the TMS320C30*	BBS, 3
TMS320C25/C30	*Implementation of Adaptive Filters With the TMS320C25 or the TMS320C30*	3
TMS320C3x	*EDN Magazine 3rd-Generation (TMS320C3x) Benchmarks*	BBS
TMS320C30	*C30 Serial Port Example Program*	BBS
TMS320C30	*C Callable Functions to Initialize TMS320C30 Cache and Wait State Control*	BBS
TMS320C30	*C Callable Matrix Multiply Functions for the TMS320C30*	BBS
TMS320C30	*A Collection of Functions for the TMS320C30*	3
TMS320C30	*Doublelength Floating-Point Arithmetic on the TMS320C30*	3
TMS320C30	*Example Programs for Operation of TMS320C30 Serial Ports*	BBS
TMS320C30	*Fast (2.42* ms*) Radix-2 1024-Point FFT Routine*	BBS
TMS320C30	*An Implementation of FFT, DCT, and Other Transforms on the TMS320C30*	3
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*	BBS
TMS320C4x	*A Parallel Approach for Solving Matrix Multiplication on the TMS320C4x DSP*	#

(Continued on the next page)

¶ Refer to *DFT/FFT and Convolution Algorithms* by C. S. Burrus and T. W. Parks, published by John Wiley and Sons.
# Refer to *A Parallel Approach for Solving Matrix Multiplication on the TMS320C4x DSP* (literature number SPRA026).

Table 5–1. Application Reports (Continued)

	Image/Graphics	
TMS32020	*A Graphics Implementation Using the TMS32020 and TMS34061*	1
TMS320C30	*A DSP-Based Three-Dimensional Graphics System*	3
TMS320C3x	*TMS320C3x DSPs Supercharge 3-D Graphics*	‖
TMS320C5x	*DSP-Based Handprinted Character Recognition* (Alan Josephson)	□
	Miscellaneous	
TMS320	*The TMS320 Family and Book Overview*	1, 2, 3
TMS320	*The TMS320 Family of Digital Signal Processors*	2, 3
TMS320	*TMS32010 Macro Files*	BBS
TMS320	*TMS32010 More Macros*	BBS
TMS320	*Implementation of the Kaish Circuit Lockout System With the TMS320 Family*	#
TMS320C25	*The Texas Instruments TMS320C25 Digital Signal Microcomputer*	2
TMS320C30	*The TMS320C30 Floating-Point Digital Signal Processor*	3
TMS320C30TMS320	*Calculation of TMS320C30 Power DissipationSoftware Coding*	☆
TMS320C5x	*Guidelines for C5x Developers* (Mansoor Chishtie)	□
	Speech Coding/Recognition	
TMS320	*Firmware-Programmable μC Aids Speech Recognition*	1
TMS32010	*The Design of an Adaptive Predictive Coder Using a Single-Chip Digital Signal Processor*	1
TMS32010	*A Single-Processor LPC Vocoder*	1
TMS320C30	*A TMS320C30-Based LPC Vocoder*	¶
TMS320C5x	*Theory and Implementation of the Digital Cellular Standard Voice Coder: VSELP on the TMS320C5x* (Jason Macres)	□
TMS320C5x	*Implementation of Speaker-Independant Speech Recognition on TMS320C2x/C5x* (Raj Pawate & Peter Robinson)	□
TMS320C5x	*Automated Dialing of Cellular Telephones Using Speech Recognition*	□

(Continued on the next page)

¶ Refer to *Selected Application Notes Using the TMS320C30 Evaluation Module* (literature number SPRA021), by Nat Seshan.
# Refer to *Implementation of the Kaish Circuit Lockout System With the TMS320 Family* (literature number SPRA022).
‖ Refer to *TMS320C3x DSPs Supercharge 3-DGraphics* (literature number SPRA024).
☆ Refer to *Calculation of TMS320C30 Power Dissipation Application Report* (literature number SPRA020), by Leor Brenman and Jon Bradley.
□ Refer to *Telecommunications Applications With the TMS320C5x* (literature number SPRA033).

Table 5–1. Application Reports (Concluded)

	Telecommunications	
TMS32010	*Add DTMF Generation and Decoding to DSP-μP Designs*	1
TMS32010	*Implementation of Data Encryption Standard Using the TMS32010*	1
TMS32010	*A Real-Time Speech Subband Coder Using the TMS32010*	1
TMS32010	*Telecommunications Interfacing to the TMS32010*	1
TMS32010	*32-kbit/s ADPCM With the TMS32010*	1
TMS320C17/E17	*General-Purpose Tone Decoding and DTMF Detection*	2
TMS32020	*Digital Voice Echo Canceller With the TMS32020*	1
TMS320C30	*Implementation of a CELP Speech Coder for the TMS320C30 Using SPOX*	3
	Tools	
TMS320	*TMS320 Algorithm Debugging Techniques*	2
TMS320C30	*The TMS320C30 Applications Board Functional Description*	3
TMS320C30	*C-Coding Tips for Application Specific Processors*	¶
TMS320C30	*TMS320C30 Evaluation Module Overview*	¶

¶ Refer to *Selected Application Notes Using the TMS320C30 Evaluation Module* (literature number SPRA021), by Nat Seshan.

5.3 TMS320 DSP Designer's Notebooks

The designer's notebooks are short application notes written by TI's engineering teams. The application notes provide helpful tips for designing and programming with the TMS320 DSPs. The designer's notebook issues are available to download from the TMS320 BBS at (713) 274-2323. Table 5–2 lists the designer's notebooks currently available.

Table 5–2. Currently Available Designer's Notebooks from Texas Instruments

No.	Topic	No.	Topic
1	'C3x Block Repeat	19	Dual-Access Into Single-Access RAM on a 'C5x Device
2	Avoiding False Interrupts on the 'C3x	20	A Simple Way to Terminate Unused TMS320C40 Comm Ports
3	Bit-Reversed Addressing w/o Data Alignment on the 'C3x	21	TMS320C5x Interrupts
4	Optimizing Control Algorithms on 'C5x	22	Fast Logarithms on a Floating-Point Device
5	TMS320C30 Addressing up to 68 Gigawords	23	Switching from Bootloader to MP Mode with the TMS320C31
6	'C5x EVM Provides for Audio Processing	24	TMS320C5x Interrupt Response Time
7	Circular Buffering in Second Generation DSPs	25	TMS320C2x/'C5x EVM AIC Intialization and Configuration
8	Bit-Reversed Addressing in C on the 'C3x	26	A Novel Way of Using TMS320C40 Cache
9	Sharing Header Files in C and Assembly	27	Hardware UART for TMS320C3x
10	Initializing the Fixed-Point EVM's AIC	28	Using VRAMs and DSPs for System Performance
11	TMS320C25 Logical Shifts in Parallel With ALU Operations	29	Using the RBIT on the TMS320E25
12	TMS320C40 Boot Loader Selection	30	Addressing Peripherals as Data Structures in C
13	Reducing System Power Requirements	31	Interrupts in C on the TMS320C3x
14	Interfacing the TMS320C31 to A/D and D/A Devices	32	TMS320C40 Emulator Tips
15	Efficient Coding on the TMS320C5x	33	Floating-Point C Compiler: Tips and Tricks - Part 1
16	TMS320C40 DMS Memory Transfer Timing	34	Guidelines for Decoupling Capacitors on DSP Designs
17	Designing with the TMS320C40 Comm Ports: Part 1	35	TMS320C5x Interrupts and the Pipeline
18	Creating a Delay Buffer on a TMS320C2x EVM		

5.4 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**
  200 Old Tappan Rd.
  Old Tappan, NJ 07675
  (201) 767–5973

  - ***A Digital Signal Processing Laboratory Using the TMS32010*** (D.L. Jones and T.W. Parks) is a comprehensive, self-study manual that introduces students to realtime signal processing through programming of the TMS32010 DSP and use of the TMS32010 evaluation module (EVM) and analog interface board (AIB). Examples and exercises (with solutions) are included. A demonstration disk from Atlanta Signal Processors Inc. (included) has been specially prepared to complement the text.
  - ***A Digital Signal Processing Laboratory Using the TMS320C25*** (B.A. Hutchins and T.W. Parks) is a self-study manual that introduces students to realtime signal processing through programming the TMS320C25 DSP and using the TMS320C25 software development system (SWDS) and analog interface board (AIB). Examples and exercises (with solutions) are included.
  - ***Practical Approaches to Speech Coding*** (P. Papamichalis) presents a conceptual approach to speech coding techniques through practical examples of speech coding applications. Speech coding design considerations are discussed throughout the text.
  - ***Servo Motor and Motion Control Using Digital Signal Processors*** (Y. Dote) is a comprehensive reference guide to designing and implementing digital servo control systems. The book discusses complete system design, including motors, sensors, digital control theory, controller design and implementation. Also included is a design example using the TMS320, with the source code given in the text and on a floppy disk. The book emphasizes practical issues in realizing control systems and avoiding unnecessary mathematics.
  - ***Digital Signal Processing Applications With the TMS320 Family***, volume 1 (K.S. Lin, editor) is a reference guide for practicing engineers developing applications. The guide consists of application reports, published articles, and technical reports covering a wide range

of DSP applications. Source code for application algorithms is given in the text as well as on two floppy disks (included).

- ***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).
- ***Digital Signal Processing Applications With the TMS320 Family***, volume 3 (P. Papamichalis, editor) focuses primarily on TMS320C3x applications, such as implementation of 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).
- ***TMS320C1x User's Guide*** (Texas Instruments) is a reference for the TMS320C1x digital signal processors. The guide introduces the TMS320 family and describes the TMS320C1x devices' pinouts, signals, architecture, assembly language instructions, and software and hardware applications. Data sheets are included.
- ***TMS320C2x User's Guide*** (Texas Instruments) is a reference for the TMS320C2x digital signal processors. The guide introduces the TMS320 family and describes the TMS320C2x devices' pinouts, signals, architecture, assembly language instructions, and software and hardware applications. Data sheets are included.

❑ **John Wiley and Sons**
One Wiley Dr.
Somerset, NJ 08875
(800) 526–5368

- ***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 (Cooley-Tukey, prime-factor, and Winograd) to Fast Fourier Transforms (FFT) are also described in detail. TMS320 coding examples are included.
- ***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.
- ***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.

- ***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.
- ***Digital Signal Processing With C and the TMS320C30*** (Ralph Chassaing) describes the architecture and instruction set of the TMS320C30. Progamming examples using both C and TMS320C30 are include throughout the text. A disk of programming examples is included.

❑ **Computer Science Press**
W. H. Freeman and Company
4419 West 1980 South
Salt Lake City, UT 84104
(801) 973–4660

- ***Digital Signal Processing Design*** (A. Bateman and W. Yates) provides a comprehensive survey of the rapidly expanding field of DSP. Material is presented in the form of specific algorithms with an explanation of how to apply the algorithm in hardware and software design. The book contains many examples of implementation of DSP applications on TMS320 devices.
- ***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.

5.5 Technical Articles Bibliography

Since the TMS32010 DSP was introduced in 1982, the TMS320 family has received an ever-increasing amount of recognition. Many technical articles are being written about TMS320 applications.

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 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 SPRA012A, SPRA016, and SPRA017, respectively) and in *Digital Control Applications with the TMS320 Family* (SPRA019).

- **General-Purpose DSP**
  - Papamichalis, P.E., and C.S. Burrus, "Conversion of Digit-Reversed to Bit-Reversed Order in FFT Algorithms," *Proceedings of ICASSP 89*, USA, pages 984–987, May 1989.
  - Papamichalis, P., and R. Simar, Jr., "The TMS320C30 Floating-Point Digital Signal Processor," *IEEE Micro Magazine*, USA, pages 13–29, December 1988.
  - Papamichalis, P.E., "FFT Implementation on the TMS320C30," *Proceedings of ICASSP 88*, USA, Volume D, page 1399, April 1988.
  - Peterson, C., Zervakis, M., Shehadeh, N., *"Adaptive Filter Design and Implementation Using the TMS320C25 Microprocessor"*, Computers in Education Journal, USA, Volume 3, Number 3, pages 12–16, July–September 1993.
  - Simar, Jr., R., and A. Davis, "The Application of High-Level Languages to Single-Chip Digital Signal Processors," *Proceedings of ICASSP 88*, USA, Volume D, page 1678, April 1988.
  - Gass, W., R. Tarrant, T. Richard, B. Pawate, M. Gammel, P. Rajasekaran, R. Wiggins, and C. Covington, "Multiple Digital Signal Processor Environment for Intelligent Signal Processing," *Proceedings of the IEEE, USA*, Volume 75, Number 9, pages 1246–1259, September 1987.
  - Chassaing, R., Horning, D.W., *"Digital Signal Processing with Fixed and Floating-Point Processors"* , CoED, USA, Volume 1, Number 1, pages 1–4, March 1991.
  - Lin, K., G. Frantz, and R. Simar, Jr., "The TMS320 Family of Digital Signal Processors," *Proceedings of the IEEE*, USA, Volume 75, Number 9, pages 1143–1159, September 1987.
  - Simar, Jr., R., T. Leigh, P. Koeppen, J. Leach, J. Potts, and D. Blalock, "A 40 MFLOPS Digital Signal Processor: the First Supercomputer on

a Chip," *Proceedings of ICASSP 87*, USA, Catalog Number 87CH2396–0, Volume 1, pages 535–538, April 1987.

- Prado, J., and R. Alcantara, "A Fast Square-Rooting Algorithm Using a Digital Signal Processor," *Proceedings of IEEE*, USA, Volume 75, Number 2, pages 262–264, February 1987.
- Lovrich, A., Reimer, J., *"An Advanced Audio Signal Processor"* , Digest of Technical Papers for 1991 International Conference on Consumer Electronics, June 1991.
- Frantz, G., K. Lin, J. Reimer, and J. Bradley, "The Texas Instruments TMS320C25 Digital Signal Microcomputer," *IEEE Microelectronics*, USA, Volume 6, Number 6, pages 10–28, December 1986.
- Simar, Jr., R., and J. Reimer, "The TMS320C25: a 100 ns CMOS VLSI Digital Signal Processor," *1986 Workshop on Applications of Signal Processing to Audio and Acoustics*, September 1986.
- Essig, D., C. Erskine, E. Caudel, and S. Magar, "A Second-Generation Digital Signal Processor," *IEEE Journal of Solid-State Circuits*, USA, Volume SC–21, Number 1, pages 86–91, February 1986.
- Magar, S., D. Essig, E. Caudel, S. Marshall and R. Peters, "An NMOS Digital Signal Processor with Multiprocessing Capability," *Digest of IEEE International Solid-State Circuits Conference*, USA, February 1985.
- Erskine, C., and S. Magar, "Architecture and Applications of a Second-Generation Digital Signal Processor," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, USA, 1985.

❑ **Graphics/Imagery**

- Reimer, J., and A. Lovrich, "Graphics with the TMS32020," *WESCON/85 Conference Record*, USA, 1985.

❑ **Speech/Voice**

- DellaMorte, J., and P. Papamichalis, "Full-Duplex Real-Time Implementation of the FED-STD-1015 LPC-10e Standard V.52 on the TMS320C25," *Proceedings of SPEECH TECH 89*, pages 218–221, May 1989.
- Pawate, B.I., and G.R. Doddington, "Implementation of a Hidden Markov Model-Based Layered Grammar Recognizer," *Proceedings of ICASSP 89*, USA, pages 801–804, May 1989.
- Reimer, J.B. and K.S. Lin, "TMS320 Digital Signal Processors in Speech Applications," *Proceedings of SPEECH TECH '88*, April 1988.
- Reimer, J.B., M.L. McMahan, and W.W. Anderson, "Speech Recognition for a Low-Cost System Using a DSP," *Digest of Technical Papers*

for 1987 International Conference on Consumer Electronics, June 1987.

- Frantz, G.A., and K.S. Lin, "A Low-Cost Speech System Using the TMS320C17," *Proceedings of SPEECH TECH '87*, pages 25–29, April 1987.
- Papamichalis, P., and D. Lively, "Implementation of the DOD Standard LPC–10/52E on the TMS320C25," *Proceedings of SPEECH TECH '87*, pages 201–204, April 1987.

❑ **Control**

- Ahmed, I., "16-Bit DSP Microcontroller Fits Motion Control System Application," *PCIM*, October 1988.
- Ahmed, I., "Implementation of Self Tuning Regulators with TMS320 Family of Digital Signal Processors," *MOTORCON '88*, pages 248–262, September 1988.
- Allen, C. and P. Pillay, *"TMS320 Design for Vector and Current Control of AC Motor Drives"*, Electronics Letters, UK, Volume 28, Number 23, pages 2188–2190, November 1992.
- Panahi, I. and R. Restle, *"DSPs Redefine Motion Control"* , Motion Control Magazine, December 1993.
- Lovrich, A., G. Troullinos, and R. Chirayil, "An All-Digital Automatic Gain Control," *Proceedings of ICASSP 88*, USA, Volume D, page 1734, April 1988.
- Ahmed, I., and S. Meshkat, "Using DSPs in Control," *Control Engineering*, February 1988.
- Meshkat, S., and I. Ahmed, "Using DSPs in AC Induction Motor Drives," *Control Engineering*, February 1988.
- Matsui, N. and M. Shigyo, *"Brushless DC Motor Control Without Position and Speed Sensors"* , IEEE Transactions on Industry Applications, USA, Volume 28, Number 1, Part 1, pages 120–127, January–February 1992.
- Hanselman, H., "LQG-Control of a Highly Resonant Disc Drive Head Positioning Actuator," *IEEE Transactions on Industrial Electronics*, USA, Volume 35, Number 1, pages 100–104, February 1988.
- Bose, B.K., and P.M. Szczesny, "A Microcomputer-Based Control and Simulation of an Advanced IPM Synchronous Machine Drive System for Electric Vehicle Propulsion," *Proceedings of IECON '87*, Volume 1, pages 454–463, November 1987.
- Ahmed, I., and S. Lindquist, "Digital Signal Processors: Simplifying High-Performance Control," *Machine Design*, September 1987.

❑ **Multimedia**

- Reimer, J., *"DSP-Based Multimedia Solutions Lead Way Enhancing Audio Compression Performance"* , Dr. Dobbs Journal, December 1993.
- Reimer, J., G. Benbassat, and W. Bonneau Jr., *"Application Processors: Making PC Multimedia Happen"* , Silicon Valley PC Design Conference, July 1991.

❑ **Military**

- Papamichalis, P., and J. Reimer, "Implementation of the Data Encryption Standard Using the TMS32010," *Digital Signal Processing Applications*, 1986.

❑ **Telecommunications**

- Casale, S., R. Russo, and G. Bellina, "Optimal Architectural Solution Using DSP Processors for the Implementation of an ADPCM Transcoder," *Proceedings of GLOBECOM '89*, pages 1267–1273, November 1989.
- Lovrich, A., and J. Reimer, "A Multi-Rate Transcoder," *Transactions on Consumer Electronics*, USA, November 1989.
- Lovrich, A. and J. Reimer, *"A Multi-Rate Transcoder"* , Digest of Technical Papers for 1989 International Conference on Consumer Electronics, June 7–9, 1989.
- Lu, H., D. Hedberg, and B. Fraenkel, "Implementation of High-Speed Voiceband Data Modems Using the TMS320C25," *Proceedings of ICASSP 87*, USA, Catalog Number 87CH2396–0, Volume 4, pages 1915–1918, April 1987.
- Ahmed, I., and A. Lovrich, "Adaptive Line Enhancer Using the TMS320C25," *Conference Records of Northcon/86*, USA, 14/3/1–10, September/October 1986.
- Troullinos, G., and J. Bradley, "Split-Band Modem Implementation Using the TMS32010 Digital Signal Processor," *Conference Records of Electro/86 and Mini/Micro Northeast*, USA, 14/1/1–21, May 1986.
- Cole, C., A. Haoui, and P. Winship, "A High-Performance Digital Voice Echo Canceller on a SINGLE TMS32020," *Proceedings of ICASSP 86*, USA, Catalog Number 86CH2243–4, Volume 1, pages 429–432, April 1986.
- Cole, C., A. Haoui, and P. Winship, "A High-Performance Digital Voice Echo Canceller on a Single TMS32020," *Proceedings of IEEE Inter-*

national Conference on Acoustics, Speech and Signal Processing, USA, 1986.

- Reimer, J., M. McMahan, and M. Arjmand, "ADPCM on a TMS320 DSP Chip," *Proceedings of SPEECH TECH 85*, pages 246–249, April 1985.
- Mock, P., "Add DTMF Generation and Decoding to DSP– µP Designs," *Electronic Design*, USA, Volume 30, Number 6, pages 205–213, March 1985.

❑ **Automotive**

- Lin, K., "Trends of Digital Signal Processing in Automotive," *International Congress on Transportation Electronic (CONVERGENCE '88*), October 1988.

❑ **Consumer**

- Frantz, G.A., J.B. Reimer, and R.A. Wotiz, "Julie, The Application of DSP to a Product," *Speech Tech Magazine*, USA, September 1988.
- Reimer, J.B., and G.A. Frantz, "Customization of a DSP Integrated Circuit for a Customer Product," *Transactions on Consumer Electronics*, USA, August 1988.
- Reimer, J.B., P.E. Nixon, E.B. Boles, and G.A. Frantz, "Audio Customization of a DSP IC," *Digest of Technical Papers for 1988 International Conference on Consumer Electronics*, June 8–10 1988.

❑ **Medical**

- Knapp and Townshend, "A Real-Time Digital Signal Processing System for an Auditory Prosthesis," *Proceedings of ICASSP 88*, USA, Volume A, page 2493, April 1988.
- Morris, L.R., and P.B. Barszczewski, "Design and Evolution of a Pocket-Sized DSP Speech Processing System for a Cochlear Implant and Other Hearing Prosthesis Applications," *Proceedings of ICASSP 88*, USA, Volume A, page 2516, April 1988.

❑ **Development Support**

- Simar, Jr., R., and A. Davis, "The Application of High-Level Languages to Single-Chip Digital Signal Processors," *Proceedings of ICASSP 88*, USA, Volume 3, pages 1678–1681, April 1988.
- Mersereau, R., R. Schafer, T. Barnwell, and D. Smith, "A Digital Filter Design Package for PCs and TMS320," *MIDCON/84 Electronic Show and Convention*, USA, 1984.

5.6 TMS320 Newsletter, *Details on Signal Processing*

The TMS320 newsletter, *Details on Signal Processing*, is published quarterly to update TMS320 customers on product information and industry trends. It covers TMS320 products, documentation, third-party support, application boards, technical information about TMS320 products, miniapplication reports, development tool updates, contacts for support, design workshops, seminars, conferences, and the TMS320 university program.

To be added to the mailing list, write to:

Texas Instruments Incorporated
Market Communications Manager, MS736
P.O. Box 1443
Houston, Texas 77251-1443

5.7 TMS320 DSP Bulletin Board Service

The TMS320 Digital Signal Processor Bulletin Board Service (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.

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 BBS with a terminal or PC and a modem. The modem must be able to communicate at a data rate of either 300, 1200, 2400, or 9600 bps. A character length of eight bits is required, with one stop bit and no parity. The telephone number of the bulletin board is (713) 274–2323. There is a 90-minute access limit per day on the bulletin board. The BBS is open 24 hours a day. ROM-code algorithms may be submitted by secure electronic transfer via the TMS320 BBS.

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 **(713) 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 BBS). The BBS can also be accessed through *Internet* at the anonymous FTP address **evans.ee.adfa.oz.au**.

The first time you access the BBS, you have very limited capabilities. Before the system operator (SYS OP) can upgrade your ID to higher access capabilities, you must answer completely the questionnaire that is automatically displayed when your password is defined. Because no further capabilities are available to you until your ID access is upgraded, you should log off after answering the questions. Once the questionnaire has been completed, your ID will be upgraded within one business day for nonrestricted access to the BBS, including downloading of useful TMS320 application programs, updates, and device and development tool information. Additional help on any menu can be obtained by using the **?** 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 pro-

tocol by selecting ***Y****our 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.

5.8 TMS320 DSP Technical Hotline

The TMS320 group at Texas Instruments maintains a DSP Hotline to answer TMS320 technical questions. Specific questions regarding TMS320 device problems, development tools, third-party support, consultants, documentation, upgrades, and new products are answered.

The TMS320 DSP Technical Hotline is open five days a week from 8:00 AM to 5:30 PM Central Time. It is staffed with DSP personnel ready to provide the support needed for your TMS320 design or evaluation.

To assure the maximum support from this service, first consult your product documentation. If your question is not answered there, gather all of the information that applies to your problem. With your information, manuals, and products close at hand, call:

TMS320 DSP Technical Hotline (713) 274–2320

For realtime transmission of information, a facsimile machine is available:

FAX (713) 274–2324

or you may submit information via electronic mail:

The Hotline email address is

4389750@mcimail.com

The MCI mail address is

4389750 or **TMS320 Hotline**

The European FAX line is

+ 33-1-3070-1032

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 (214) 644–5580

5.9 TMS320 Software Cooperative

The TMS320 Software Cooperative is designed to benefit TI customers and third parties by offering licensable software. The TMS320 Software Cooperative provides quick access to DSP applications for customers who require short development time and fast time to market. The program provides marketing support and exposure to participating third parties.

The Software Cooperative offers a data sheet folder that contains data sheets for each software algorithm available. Table 5–3 lists the third party algorithms currently available and a list of third party company names and phone numbers is presented at the end of the algorithm listing. If you would like more information regarding the TMS320 Software Cooperative algorithms or how to become a third party member, please call the TI Product Information Center. To receive the TMS320 Software Cooperative Data Sheet packet, request literature number SPRT111.

Table 5–3. Algorithms Available Through TI or Third Parties

Vocoder Algorithms

Algorithm	Source	Device	Data Rate (kbps)	Program Memory† (Words)	Data Memory† (Words)	Processor Loading (MIPS)†	Quality	Page Number
AMPS / TACS Analog Cellular	DSP Telecom	'C1x		8 K	256	6		VOC-1-1
Code Excited Linear Prediction (CELP) – USFS 1016	DSP–SE	'C3x	4.8	7.2 K	6K	12.5 / 2.4	DAM: Quiet – 60 Office – 55 E4B – 50	VOC-2-1
	Ensigma	'C3x	4.8	8K	10K	19		VOC-2-2
	Signals SW	'C5x	4.8	8K	1.5K	19		VOC-2-3
	Sonitech	'C3x	4.8	4K	8K	16.6	DAM: Quiet – 65 Office –55	VOC-2-4
CELP w/ Enhanced Codebook Search	Sonitech	'C3x	4.8	4K	8K	12	DAM: Quiet – 65 Office –55	
CELP	DSP Telecom	'C5x	6.4, 7.2, 8, 9.6	8K	2K	20		VOC-2-5
	Signals SW	'C5x	7.2	8K	1.5K	19		VOC-2-3
	Lemout & Hauspie	'C3x	4.8, 7.2, 9.6	6K	1.6K	15 / 1.7		VOC-2-6
	Lernout & Hauspie	'C25	4.8	5K	1.6K	10		VOC-2-6
HLTP-CELP	CNET	'C3x	8					VOC-2-7
Variable-Rate (LVR) CELP	Ensigma	'C3x	4 – 8	48K	8K	18		VOC-2-8
Continuously Varying Slope Delta Mod. (CVSD)	ASPI	'C3x	16	490	215	1.8 / 1.7		VOC-3-1

† Analysis / Synthesis or Transmit / Receive

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Vocoder Algorithms

Algorithm	Source	Device	Data Rate (kbps)	Program Memory† (Words)	Data Memory† (Words)	Processor Loading (MIPS)†	Quality	Page Number
G.711 Compander	ASPI	'C3x	64	131	16	0.5		VOC-4-1
	DSP-SE	'C3x	48, 56, 64	39	0	0.5 (μ-law)	S/N: 39dB	VOC-4-2
	DSP-SE	'C3x	48, 56, 64	43	0	0.6 (A-law)	S/N: 39dB	VOC-4-2
G.711 μ-Law/A-Law Detector	DSP-SE	'C3x	48, 56, 64	0.4 K	0.35 K	0.3		VOC-4-3
G.721	ASPI	'C3x	32	1950	719	3.7 / 4.2		VOC-5-1
	CNET	'C1x				5	Half-Duplex	VOC-2-7
	Oros	'C25	32					VOC-5-2
	Ensigma	'C3x	16	8K	8 K	19.6		VOC-8-2
	Signal SW	'C5x	16	8K	0	25 / 16		VOC-2-3
	TELOGY	'C3x	16	32K	0			VOC-8-3
G.722 SB-ADPCM	ASPI	'C3x	64	2402	320	5 / 4.2		VOC-6-1
	CNET	'C3x						VOC-2-7
	CNET	'C1x				5	Half-Duplex	VOC-2-7
	CNET	'C2x				10		VOC-2-7
	CNET	'C5x				7		VOC-2-7
	DSP-SE	'C3x	48, 56, 64	603 / 473	264 / 266	5.2 / 5.1		VOC-6-2
	Oros	'C25	64					VOC-5-2
	Signal SW							VOC-2-3
G.723 ADPCM	ASPI	'C3x	24, 32, 40	1950	719	3.7 / 4.2		VOC-5-1
G.726 ADPCM (μ-Law)	DSP-SE	'C3x	16, 24, 32, 40	757 / 882	128 / 71	3.6/ 4.2		VOC-7-1
(PCM)	DSP-SE	'C3x	16, 24, 32, 40	757 / 882	128 / 71	3.7 / 4.5		VOC-7-1
G.728 Low Delay - CELP	CNET	'C3x	16					VOC-2-7
	DSP-SE	'C3x	16	6,600 / 6,268	913 / 1,127	11 / 6.6		VOC-8-1
	Ensigma	'C3x	16	8K	8K	19.6		VOC-8-2
	Signal SW	'C5x	16	8K	0	25 / 16		VOC-8-3
	TELOGY	'C3x	16	32K	0	11 / 6.6		VOC-8-3

† Analysis / Synthesis or Transmit / Receive

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Vocoder Algorithms

Algorithm	Source	Device	Data Rate (kbps)	Program Memory† (Words)	Data Memory† (Words)	Processor Loading (MIPS)†	Quality	Page Number
General Speciale Moblite (GSM)	CNET	'C2x	13			4.5		VOC-2-7
	CNET	'C5x	13					VOC-2-7
	Oros	'C25	13					VOC-5-2
	Signals SW	'C2x	13	8K	2K	5		VOC-2-3
Improved Multiband Excitation (IBME) Vocoder	Digital Voice Systems	'C3x	2.4 – 9.6	9K	0	5.8 / 4.5		VOC-9-1
	Digital Voice Systems	'C5x	2.4 – 9.6	16K	0	6.8 / 7		VOC-9-1

† Analysis / Synthesis or Transmit / Receive

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Vocoder Algorithms

Algorithm	Source	Device	Data Rate (kbps)	Program Memory† (Words)	Data Memory† (Words)	Processor Loading (MIPS)†	Quality	Page Number
	TELOGY	'C3x	6.4	32K	0			VOC-9-2
Inmarsat-B (APC)	TELOGY	'C3x	9.6, 16	32K	0			VOC-10-1
Linear Predictive Coding (LPC-10)	ASPI	'C3x	2.4	2.6K	2K	1.2 / 1.3		VOC-11-1
	EIA	'C2x	2.4, 9.6			10		VOC-11-2
	EIA	'C5x	2.4, 9.6			10		VOC-11-2
	Oros	'C25	2.4					VOC-5-2
LP'C Multipulse	Oros	'C25	4.8, 8, 9.6, 16					VOC-5-2
LPC-10e (USFS 1015)	DSP-SE	'C3x	2.4	7K	4K	5 / 5.8	DAM: Quiet – 54 Office – 46 E4B – 43	VOC-11-3
FIR Excited Sample Selective (FIRESS) LPC	Recosyst	'C3x	9.6	6K	3K	13.3		VOC-12-1
Mixed Excitation Linear Predictive (MELP)	ASPI	'C3x	1.6 – 2.4	6.5K	22.1K	10.7 / 8.8		VOC-13-1
Self-Excited Vocoder (SEV)	ASPI	'C3x	9.6	18,000	12,000	15.8		VOC-14-1
	ASPI	'C3x	4.8	18,000	30,000	13.3		VOC-14-1
Sinusoidal Transform Coder (STC)	Aware	'C3x	> 1.2					VOC-15-1
Subband Coder	ASPI	'C3x	16	1,668	960	6.4 / 3.2		VOC-16-1
VSELP Digital Cellular (TIA IS-54)	DSP–SE	'C3x	7.95	7.8K	3K	13 / 2.5		VOC-17-1
PDC Baseband Processing	DSP Telecom	'C5x		16K	4K	25		VOC-17-2

† Analysis / Synthesis or Transmit / Receive

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Speech Recognition / Synthesis Algorithms

Algorithm	Source	Device	Data	Program Memory (Words)	Data Memory (Words)	Processor Loading (MIPS)	Language	Page Number
Speaker Dependent Recognition	DSP Telecom	'C1x		32K	256	6		SRS-1-1
	Voice Control Systems	'C3x	1 – 8	29K	52K + 13K/chan	16.3 (8 chan)	language indepen-dent	SRS-1-2
Speaker Independent Recognition	Lernout & Hauspie	'C3x	1	64K	64 K			SRS-2–1
	Voice Control Systems	'C3x	1 – 8	21.8K	20K + 13K/chan	16.3 (8 chan)	28 languages	SRS-2-2
	Ensigma	'C3x						
Continuous Speaker Independent Recognition	Voice Control Systems	'C3x	1 – 2	84K	64 K/chan	15 (2 chan)	7 languages	SRS-2-3
Speaker Independent Alphabet & Alphanu-meric Recognition	Voice Control Systems	'C3x	1 – 8	21.8K	20K + 13K/chan	16.3 (8 chan)	12 languages	SRS-2-4
Speaker Verification	Voice Control Systems	'C3x	1 – 8	30.8K	20K + 13K/chan	16.3 (8 chan)	language indepen-dent	SRS-3-1
Text-to-Speech	Berkeley Speech	'C25	1 – 4	2K	50/chan			SRS-4-1
	Lernout & Hauspie	'C3x	1	20K	512K	5.8	6 languages	SRS-4-2
	Lernout & Hauspie	'C25	1	6K	2K	10.6	6 languages	SRS-4-2

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Audio Algorithms

Algorithm	Source	Device	Sampling Rate	Program Memory (Words)	Data Memory (Words)	Processor Loading (MIPS)	Page Number
10-Band Graphic Equalizer	ASPI	'C3x	16, 32, 44.1, 48-khz	4K	3K	9.2 (stereo)	AUD-1-1
Automatic Gain Control	DSP-SE	'C3x	8-khz	300	50	0.15	AUD-2-1
Echo and Reberveration	ASPI	'C3x	32, 44.1, 48-khz	665	8,192	16.7 (6 walls)	AUD-3-1
Loudness Measurement	DSP-SE	'C3x	8-khz	150	25	0.07	AUD-2-1
MPEG Audio Encoder	ASPI	'C3x	32, 44.1, 48-khz	12,594	6,346	18 @ 48K	AUD-4-1
MPEG Audio Decoder	ASPI	'C3x	32–448 kb/s	3,891	3,211	9.2 @ 48K	AUD-4-2
Music Synthesis	ASPI	'C3x	22.05, 44.1-khz	47,527	0	16.7	AUD-5-1
Multirate Filtering Time Scaler	DSP-SE	'C3x	8 – 48 kb/s	0.5K	0.25K	3.0 – 5.7	AUD-6-1
Pitch Restoring Time Scaler	DSP-SE	'C3x	8 – 48 kb/s	1.5K	1.25K	2.0 – 4.9	AUD-6-1
Sample Rate Conversion	ASPI	'C3x	8, 10, 16, 20, 32, 44.1, 48-khz	3K	4K	8.3 – 33.3	AUD-6-2
Time Scale Modification	ASPI	'C3x	8-khz	36K	6K	16.7	AUD-6-3
Voice Activity Detection	DSP-SE	'C3x	8-khz	350	75	0.04	AUD-2-1
Volume Controller	DSP-SE	'C3x	8-khz	100	10	0.02	AUD-2-1

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Telecommunication Algorithms

Algorithm	Source	Device	Data Rate (kbps)	Program Memory† (Words)	Data Memory† (Words)	Processor Loading (MIPS)†	Page Number
Call Progress Tone Detector	DSP-SE	'C3x		461 / 623	389 / 123	0.82 / 0.52	TEL-1-1
DTMF Generator/Decoder	DSP-SE	'C3x	24 chan	1,112 / 712	467 / 711	1.25 / 0.6	TEL-2-1
	Ensigma	'C3x	30 chan		1K	15.8	TEL-2-2
	Signal SW						TEL-2-3
Group 3 Fax (V.29, T.30, V.21, T.4, V.27ter)	Dallas SP	'C3x, 'C4x	.3, 2.4, 4.8, 7.2, 9.6	8000	1024	2.5 / 4.7	TEL-3-1
	DSP-SE	'C3x	2.4, 4.8, 7.2, 9.6	8K	4K	3.3 / 13.3	TEL-3-2
	ILLICO!	'C3x	7.2, 9.6	8K	540	6.75	TEL-3-3
V.21, V.22, V.23, V.27, V.29	Signal SW	'C2x, 'C5x					TEL-3-4
QFax Group 4 Fax (T.62bis)	DGM&S	'C3x	64			4.2	TEL-4-1
Line Echo Canceller	DSP-SE	'C3x		1.25K	750	7.9	TEL-5-1
V.22bis Modem	Dallas SP	'C3x	0.3, 1.2, 2.4	8000	2048	8.3	TEL-6-1
	DSP-SE	'C3x	1.2, 2.4	8K	3K	5	TEL-6-2
V.32bis Modem	DSP-SE	'C3x	4.8, 7.2, 9.6, 12, 14.4	10K	5K	13	TEL-7-1
	Signals SW	'C2x				10	TEL-3-4
	Signals SW	'C5x				15	TEL-3-4

† Analysis / Synthesis or Transmit / Receive

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Image Algorithms

Algorithm	Source	Device	Color	Program Memory (Words)	Data Memory (Words)	Processor Loading (MIPS)	Page Number
JPEG Still Image (8–R8)	ASPI	’C3x	8 / 16 / 24 bit	3,226	6,061	4 µsec/color pixel (33 MHz ’C31)	IMG-1-1
	Sonitech	’C3x, ’C4x				5.5 µsec/ color pixel (33 MHz ’C30)	IMG-1-2
Optical Character Recognition	EIA	’C2x				10	IMG-2-1
(OCR)	EIA	’C5x				10	IMG-2-1

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Operating Systems

Name	Source	Device	Max Memory	Min Memory	Host Platform	Page Number
Debugger Support Kit	3L					OS-1-1
Helios	Perihelion	’C4x	4M	16K	PC Sun	OS-2-1
MX	TELOGY	’C3x, ’C5x	5.5K	2.5K	PC	OS-3-1
Nucleus RTX	Accelerated Technology	’C3x, ’C4x, ’C25, ’C5x	4856	1036	PC	OS-4-1
Nucleus Plus	Accelerated Technology	’C3x, ’C4x	4856	1036	PC	OS-4-2
OS320	Enea Data	’C5x			PC Sun	OS-5-1
QRux	DGM&S	’C3x			PC	OS-6-1
Real-Time Interface to SIMULINK	dSPACE	’C3x, ’C4x			PC Sun	OS-6-2
SPOX-KNL	Spectron / Objectif	’C3x, ’C4x, ’C5x		1,532	PC Sun/4	OS-7-1
SPOX-DEBUG	Spectron / Objectif	’C3x, ’C4x, ’C5x			PC Sun/4	OS-7-2
SPOX	Spectron / Objectif	’C3x, ’C4x, ’C5x		1,532	PC Sun/4	OS-7-3
Virtuoso Classico	ISI	’C3x, ’C4x		6K	PC Sun	OS-8-1
Virtuoso Micro	ISI	’C3x, ’C4x		4.5K	PC Sun	OS-8-2
Virtuoso Molto	ISI	’C3x, ’C4x			PC Sun	OS-8-3
Virtuoso Nano	ISI	’C3x, ’C4x		200	PC Sun	OS-8-4

Table 5–3. Algorithms Available Through TI or Third Parties (Continued)

Run-Time Support Libraries

Name	Source	Device	Description	Host Platform	Page Number
3L DSP Library	3L	'C4x	Basic DSP functions	PC Sun	RTS-1-1
DSPLib	Ensigma	'C3x, 'C4x	Signal processing functions	PC Sun	RTS-1-2
DSP Library	Sonitech	'C3x/'C4x	Optimized DSP functions	PC Sun	RTS-1-3
DSP/Vector Library	Sinectonalysis	'C4x	Signal processing functions	PC	RTS-1-4
Virtuoso Molto	ISI	'C3x, 'C4x	Optimized DSP and mathematical libraries	PC Sun	RTS-1-5
C-BLAS 1, 2, 3	Sinectonalysis	'C3x, 'C4x	Linear algebra functions	PC	RTS-2-1
EISPACK	Sinectonalysis	'C3x, 'C4x	Eigenvalue/Eigenvector functions	PC	RTS-2-2
FasTar	Tartan	'C3x, 'C4x	Mathematical functions	PC Sun	RTS-2-3
FloTar	Tartan	'C3x, 'C4x	64-bit precision mathematical functions	PC Sun	RTS-2-4
MATLAB Interface	Sonitech	'C3x, 'C4x	MATLAB drivers for SPIRIT-30/40	PC	RTS-2-5
SPOX – MATH	Spectron / Objectif	'C3x, 'C4x	Signal processing and mathematical functions	PC Sun/4	RTS-2-6
STD/Mathlib	Sinectonalysis	'C3x, 'C4x	Mathematical functions	PC Sun HP RS6000	RTS-2-7
WCI	Wideband Computers	'C3x, 'C4x	Mathematical functions	PC Sun	RTS-2-8
EYE-LIB	Sinectonalysis	'C4x	Image processing functions	PC	RTS-3-1
GIPS Vision	EIA	'C2x, 'C5x	Image processing library	PC VME	IMG-2-1
Spectral Analysis Library	Sonitech	'C3x, 'C4x	Spectrum estimation functions	PC Sun	RTS-4-1

Table 5–3. Algorithms Available Through TI or Third Parties (Concluded)

Miscellaneous

Algorithm	Source	Device	Color	Program Memory (Words)	Data Memory (Words)	Processor Loading (MIPS)	Page Number
RF Identification	Recosyst	'C1x				4.5	MSC-1-1

5.10 Algorithm Software Sources

This listing provides the name, country, and the contact telephone number for each of the referenced software packages. The **bold** type indicates the actual corporate name associated with the software package.

Third Party Contacts	Telephone
3L - **3L, LTD.**, Edinburgh, Scotland	44 31 66 24 333
Accelerated Technology, Inc., Mobile, AL USA	(205) 661-5770
ASPI - **Atlanta Signal Processors, Inc.**, Atlanta, GA USA	(404) 892-7265
AWARE - **AWARE, Inc.,** Cambridge, MA USA	(617) 577-1700
Berkeley Speech - **Berkeley Speech Tech, Inc.**, Berkeley, CA USA	(510) 841-5083
CNET - **CNET LAA/TSS/CMC**, Cedex, France	33 96 05 39 41
Dallas SP - **Dallas Signal Processing**, Plano, TX USA	(214) 985-5130
DVS - **Digital Voice Systems, Inc.,** Burlington, MA USA	(617) 270–1030
DGM&S - **DGM&S**, Mt. Laurel, NJ USA	(609) 866-1212
DSP-SE - **DSP Software Engineering, Inc.,** Bedford, MA USA	(617) 275-3733
DSP Telecom, Givat Shmuel, Israel	972 3 531 3300
dSpace, Paderborn, Germany	49 5251 16380
EIA, Cedex, France	33 16 1 69 41 37 70
Enea - **Enea Data**, Täby, Sweden	46 8 638 5000
Ensigma - **Ensigma, Ltd.**, Gwent, UK	44 291 625 422
ILLICO - **ILLICO!**, Santa Clara, CA USA	(408) 980-8170
ISI, Linden, Belgium	32 16 62 15 85
Lernout & Hauspie **- Lernout & Hauspie Speech Products**, Woburn, MA USA and Ieper, Belgium	(617) 932-4118 32 24 60 33 97
Objectif, Cedex, France	33 16 1 47 35 30 31
Oros, Meylan, France	33 76 90 62 36
Perihelion, Somerser, UK	44 749 344 345
Recosyst - Recosyst R & D, Emmerich, Germany	49 2822 18731
Sinectonalysis - **Sinectonalysis, Inc.**, West Newton, MA USA	(617) 894-8296
Signals SW - **Signals & Software, Ltd.**, Middlesex, UK	44 81 426 9533
Sonitech - Sonitech International, Inc., Wellesley, MA USA	(617) 235-6824
Spectron - **Spectron Microsystems**, Mountain View, CA USA	(415) 903-2247
Tartan - **Tartan, Inc.**, Monroeville, PA USA	(412) 856-3600
Teknic - **Teknic, Inc.**, Rochester, NY USA	(716) 546-3212
TELOGY - **TELOGY Networks, Inc.**, Gaithersburg, MD USA	(301) 948-5204
Voice Control - **Voice Control Systems**, Dallas, TX USA	(214) 386-0300
Wideband - **Wideband Computers, Inc.**, Mountain View, CA USA	(415) 962-8722

Chapter 6

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 Regional Technology Centers (RTCs).

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**

6.1 Technical Training Organization (TTO) Services

The Technical Training Organization (TTO) offers three- or four-day DSP design workshops, a digital control workshop, and C-programming 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. Five different DSP design workshops are available, each covering a different generation of the TMS320 DSP family.

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.

6.1.1 TMS320C2x Design Workshop

The TMS320C2x DSP design workshop is tailored for design engineers who are involved with the early stages of TMS320C2x application development. The workshop enables the designer to more effectively use the TMS320C2x digital signal processors through hands-on experience. The designer is introduced to numerous hardware and software techniques for application of current DSP algorithms. Previous experience with assembly language programming and DSP knowledge is recommended.

Topics covered in this workshop include:

- Introduction to the ’C2x DSP
- Program development basics
- Addressing
- Multiplication
- Program control
- Numerical issues
- DSP fundamentals
- Developing an algorithm
- Subroutines, macros, and interrupts
- On-chip resources: timer and block 0
- Hardware interfaces
- Multiprocessing interfaces
- Design support

6.1.2 TMS320C3x Design Workshop

The TMS320C3x DSP design workshop introduces design engineers to the powerful TMS320C3x generation of DSPs. Hands-on, EVM-based exercises throughout the course give the designer a rapid start in utilizing TMS320C3x design skills. Experience with digital design techniques is desirable. Assembly language experience is required. C language programming experience is desirable.

Topics covered in the TMS320C3x DSP design workshop include:

- Introduction to the ’C3x DSP
- Introduction to the C source debugger
- Software development
- Memory addressing
- CPU operations and floating-point format
- Enhanced CPU operations
- Device pipeline and cache
- Direct memory access
- Resets, interrupts, and traps
- Special addressing modes
- Memory interface
- Timers and serial ports
- C complier and run time environment
- Mixing C and assembly language
- Example architectures
- System design

6.1.3 TMS320C4x Design Workshop

TMS320C4x products and parallel processing are the main topics of this workshop. Engineers are introduced to the ’C40 architecture and instruction set through interactive, hands-on use of simulators, compilers, and assemblers. This develops the skills needed to start ’C4x product designs quickly. Class participants are required to have experience in assembly programming; it is preferred that they also have experience in C.

Main concepts covered in the class include:

- Introduction to the ’C4x DSP
- Software development tools
- Simulator walk-through
- Memory addressing
- Basic CPU operations
- Floating-point and parallel instructions
- Device pipeline and cache
- Reset, interrupts, and traps
- Special addressing modes
- Memory interface
- Boot loader
- Communication ports
- DMA coprocessor
- C complier
- Parallel application development
- Design support

6.1.4 TMS320C5x Design Workshop

The TMS320C5x DSP design workshop introduces the student to the architecture, instruction set, and development tools for the TMS320C5x generation. The workshop offers hands-on labs and other activities. The course focuses on the unique features of the TMS320C5x generation and typical applications. Previous experience with digital techniques and knowledge of assembly language is assumed.

Topics covered in this workshop include:

- Introduction to the ’C5x DSP
- Development tools
- Addressing modes
- Arithmetic operations
- Program control
- Numerical issues
- DSP fundamentals
- Algorithm development
- Logical operators
- Interrupts and on-chip peripherals
- Hardware interfacing
- Multiprocessing interfacing
- Design support

6.1.5 Digital Control Design Workshop

The digital control design workshop covers all the fundamental issues involved in the design and implementation of control systems using TMS320 DSPs. The workshop is divided into two major parts. The first part covers theory and design of control systems and discusses practical aspects that a control design engineer should be aware of before attempting to implement a controller. The second part is devoted to hands-on experience with TMS320C25 DSPs to demonstrate and practice control implementation examples. A design and implementation software package is used to test algorithms on an actual motor positioning system.

Topics covered in the digital control design workshop include:

- System modeling
- Stability analysis
- Analysis of numerical problems
- Quantization effects
- Truncation, rounding, and scaling issues
- Sampling rate selection
- Algorithm structural optimization

6.1.6 Applications in C Design Workshop

The Applications in C Design workshop is an advanced, C programming course, which is tailored for practical, hands-on applications using Turbo C and the TI TMS320C3x C compiler. This course is for hardware and software engineers with a background in programming and an introductory knowledge of C. The course centers around data structure concepts illustrated with application examples. Program examples include file filters, sorting, Huffman coding for data compression, memory management, graphics algorithms, and other utilities.

Topics covered in the Applications in C Design workshop include:

- Review of C language (syntax and conventions)
- Data structures, constructs, and concepts
- Optimization and efficiency techniques
- Arrays and pointers
- Portability issues
- Algorithms (FFT, discrete transforms, bit manipulation, etc.)

6.1.7 Registration

Class size for these workshops is limited to twelve students. Further information on courses and schedules in North America or, to register for a course offered within the United States, contact the TTO Central Registration office at (800) 336-5236, ext. 3904 or (214) 917-3894 or a TI-authorized distributor. Workshops in Europe and Asia are offered through local Regional Technology Centers (RTCs); for more information, contact the nearest RTC (see Section 6.3).

For information on courses outside the United States, contact the nearest RTC (see Section 6.3).

6.2 Design Services

The TI technical staff can offer applications assistance with customer designs through local Regional Technology Centers. Services include:

- Design assistance
- Simulation
- Emulation

Each Regional Technology 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 to 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

Once 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.

6.3 RTC Locations

Table 6–1 and Table 6–2 give the worldwide locations of the TI Regional Technology Centers.

Table 6–1. RTC 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 950 Winter Street, Suite 2800 Waltham, MA 02154–1263 (617) 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 (214) 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 6–2. RTC International Locations

International Locations	
AUSTRALIA Texas Instruments Australia Ltd. 6–10 Talavera Road, North Ryde New South Wales, Australia 2113 Tel: (61) (2) 8789000	**JAPAN** (Tokyo) Texas Instruments Japan Ltd Ms Shibaura Building 9F 4–13–23 Shibaura Minato-Ku, Tokyo, JAPAN 108 Tel: (81) (3) 3769–8700
BRAZIL Texas Instruments Electronicos Rua Paez Leme, 524–7 Andar 05424 Sao Paulo, Brazil Tel: (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 Tel: (81) (6) 204–1881
FEDERAL REPUBLIC OF GERMANY Texas Instruments Deutschland GMBH Kirchhorster Strasse 2 3000 Hannover 51, FR Germany Tel: (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 Tel: (82) (2) 5512800
FEDERAL REPUBLIC OF GERMANY Texas Instruments Deutschland GMBH Haggertystrasse 1 8050 Freising, FR Germany Tel: (49) (8161) 80–0	**SINGAPORE** Texas Instruments Singapore (Pte) Ltd. Asia Pacific Division 101 Thomson Road #23–01 United Square Singapore 1130 Tel: (65) 2519818
FRANCE (Paris) Texas Instruments France 8–10 Avenue Morane Saulnier Borte Postale 67 Velizy Villcoublay Cedex, France Tel: (33) (13) 0701001	**SWEDEN** Texas Instruments International Trade Corporation Box 30 S–164 93 Kista Isafjordsgatan 7, Sweden Tel: (8) 752–5800

(Continued on the next page)

Table 6–3. RTC International Locations (Concluded)

International Locations	
HONG KONG Texas Instruments Hong Kong Ltd. 8th Floor, World Shipping Centre 7 Canton Road Kowloon, Hong Kong Tel: (852) 7351223	**TAIWAN** Texas Instruments Taiwan Ltd. Taipei Branch 10 Floor, Bank Tower 205 Tung Hua N. Road Taipei, Taiwan 105 Republic of China Tel: (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 Tel: (39) (39) 63221	**UNITED KINGDOM** Texas Instruments Ltd. Regional Technology Center Manton Lane Bedford, England MK41 7PA Tel: (44) (234) 270111

Chapter 7

TMS320 University Program

Texas Instruments believes it is important to train future engineers and encourages universities to do advanced research in the area of DSP. TI has established the TMS320 University Program to make its resources available to universities and to assist in the incorporation of the TMS320 family 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.

Through the TMS320 University Program, TI offers discounts on development tools and DSP laboratory packages, software donations, newsletter distribution, expert engineering assistance for university research, and third-party contacts. TMS320 documentation, a technical hotline, and a 24-hour bulletin board service are also available.

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 5.4 *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.

Topic **Page**

7.1 Development Tools Available to Universities

TMS320 DSPs (TMS320C1x/C2x/C3x/C4x/C5x) and associated development tools are available to universities at a discount. If a TI integration/debug tool is purchased, Texas Instruments donates the associated assembler/linker.

The code-generation tools include the TMS320 macro assembler, linkers, and C compiler packages (includes assembler/linker).

The system integration and debug tools include emulators (XDSs), simulators, evaluation modules (EVMs), DSP Starter Kits (DSKs), and adapters.

7.2 DSP Lab Workstation Recommendations

The DSP lab workstation, designed for use at the third- or fourth-year 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 realtime DSP simulations. A DSP lab usually consists of 4 or 5 workstations. Workstations are based upon TMS320C1x, TMS320C2x, TMS320C3x, TMS320C4x, or TMS320C5x devices and typically include the following:

- **'C1x Lab Workstation:**
  - PC or terminal
  - TMS320C1x simulator
  - TMS320C1x EVM
  - TMS320C1x assembler/linker
  - Digital filter design package (DFDP), optional
- **'C2x Lab Workstation**
  - PC or terminal
  - TMS320C2x simulator
  - TMS320C2x EVM (or C2x DSK)
  - TMS320C2x/C5x optimizing ANSI C compiler package
  - TMS320C2x/C5x assembler/linker
  - Digital filter design package (DFDP), optional
- **'C3x Lab Workstation**
  - PC or terminal
  - TMS320C3x simulator
  - TMS320C3x EVM
  - TMS320C3x/C4x optimizing ANSI C compiler package
  - TMS320C3x/C4x assembler/linker
- **'C4x Lab Workstation**
  - PC or terminal
  - TMS320C4x simulator
  - TMS320C3x/C4x optimizing ANSI C compiler
  - TMS320C3x/C4x assembler/linker
- **'C5x Lab Workstation**
  - PC or terminal
  - TMS320C5x simulator
  - TMS320C5x EVM (or C5x DSK)
  - TMS320C2x/C5x optimizing ANSI C compiler package
  - TMS320C2x/C5x assembler/linker

The following textbooks are recommended for the indicated workstation environments:

- **'C1x Workstation**
  - *TMS320C1x User's Guide* from Texas Instruments (literature number SPRU013C) (Prentice-Hall)
  - Lab Manual: *A Digital Signal Processing Laboratory Using the TMS32010*, by Parks and Jones (Prentice-Hall)
- **'C2x Workstation**
  - *TMS320C2x User's Guide* from Texas Instruments (literature number SPRU014B) (Prentice-Hall)
  - Lab Manual: *A Digital Signal Processing Laboratory Using the TMS320C25*, by Hutchins and Parks (Prentice-Hall)
  - *Digital Signal Processing With C and the TMS320C25,* by Chassaing and Horning (John Wiley and Sons)
- **'C3x Workstation**
  - *TMS320C3x User's Guide* from Texas Instruments (literature number SPRU031C)
  - *Digital Signal Processing Applications with the TMS320C30 Evaluation Module*
  - *Digital Signal Processing with C and the TMS320C30,* Chassaing (John Wiley and Sons)
- **'C4x Workstation**
  - *TMS320C4x User's Guide* from Texas Instruments (literature number SPRU063)
- **'C5x Workstation**
  - *TMS320C5x User's Guide* from Texas Instruments (literature number SPRU056A)

Additional TMS320 DSP textbooks are available (refer to Section 5.4, *University Textbooks*, for a complete listing).

7.3 DSP Research Workstation

A DSP research lab workstation is created by adding the following equipment to the DSP lab workstation:

- **'C1x Research Workstation**
  - TMS320C1x XDS/22 emulator
  - DSP software library
- **'C2x Research Workstation**
  - TMS320C2x XDS/22 emulator
  - DSP software library
- **'C3x Research Workstation**
  - TMS320C3x XDS500 emulator, or TMS320C3x XDS1000 development environment
  - DSP software library
- **'C4x Research Workstation**
  - TMS320C4x XDS510 emulator
  - TMS320C4x PPDS
- **'C5x Research Workstation**
  - TMS320C5x XDS510 emulator

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 Bulletin Board Service (BBS), hotline, newsletter, and upgrading of TMS320 tools with the latest version. TI also offers TTO workshop materials, which can be incorporated into the DSP courses. 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, contact the nearest TI Field Sales Office or write to:

TMS320 University Program
Texas Instruments Incorporated
P.O. Box 1443, M/S 701
Houston, TX 77251–1443

In Europe, contact:

Texas Instruments France
Boite Postale 5
06270 Villeneuve-Loubet
Nice, France
Attn: TMS320 University Program

In the Far East, contact:

Texas Instruments Japan Limited
MOS Logic Product Marketing
MS Shibaura Bldg
13 – 23, Shibaura 4–Chome
Minato-Ku
Tokyo 108, Japan

Appendix 1

TMS320 Product Information

This appendix provides ordering information for TMS320 devices and development tools.

The TMS320 device prefix and suffix designators used in the TMS320 product numbering system are explained in the last two sections.

1.1 DSP Device and Tool Part Numbers

Table 1 and Table 2 list important information about each DSP chip. Table 3 and Table 4 supply information for all available TMS320 support tools.

Table 1. TMS320 Digital Signal Processor Commercial Part Numbers

Device Name	Operating Frequency	Package Type	Typical§ Dissipation	Temp Range
TMS320C10NL†	20 MHz	Plastic 40-pin DIP	165 mW	0°/70°C
TMS320C10NL-14†	14 MHz		140 mW	0°/70°C
TMS320C10NL-25†	25 MHz		200 mW	0°/70°C
TMS320C10FNA	20 MHz	Plastic 44-lead PLCC	165 mW	–40°/85°C
TMS320C10FNL-14	14-MHz		144 mW	0°/70°C
TMS320C10FNL-25†	25 MHz		200 mW	0°/70°C
TMS320C10FNL†	20 MHz	Plastic 44-lead PLCC	165 mW	0°/70°C
TMS320C10FNL-14	14-MHz		144 mW	0°/70°C
TMS320C10FNL-25†	25 MHz		200 mW	0°/70°C
TMS320C10FNA	20 MHz	Plastic 44-lead PLCC	165 mW	–40°/85°C
TMS320C10FNL-14	14-MHz		144 mW	0°/70°C
TMS320C10FNL-25†	25 MHz		200 mW	0°/70°C
TMS320C14FNA	20 MHz	Plastic 68-lead PLCC	165 mW	–40°/85°C
TMS320C14FNL	25 MHz		275 mW	0°/70°C
TMS320C14FZA	24 MHz		350 mW	–40°/85°C
TMS320E14FZA	24 MHz	Ceramic 68-lead CERQUAD	325 mW	–40°/85°C
TMS320E14FZL	25 MHz	Ceramic 68-lead CERQUAD	325 mW	0°/70°C
TMS320P14FNL	25 MHz	Plastic 68-lead PLCC	325 mW	0°/70°C
TMS320C15NL†	20 MHz	Plastic 40-pin DIP	225 mW	0°/70°C
TMS320C15NL-25†	25 MHz		250 mW	0°/70°C
TMS320C15FNL†	20 MHz	Plastic 44-lead PLCC	225 mW	0°/70°C
TMS320C15FNL-25†	25 MHz		250 mW	0°/70°C
TMS320E15FZL	20 MHz	Ceramic 44-lead CERQUAD	275 mW	0°/70°C
TMS320E15JDL‡	20 MHz	Ceramic 40-pin DIP	275 mW	0°/70°C
TMS320E15JDL-25	25 MHz		325 mW	0°/70°C
TMS320E15JDA	20 MHz	Ceramic 40-pin DIP	275 mW	–40°/85°C
TMS320C15PEL	20 MHz	Plastic 44-pin PQFP	225 mW	0°/70°C
TMS320C15NA	20 MHz	Plastic 44-pin DIP	225 mW	–40°/85°C

† A military version is available.
‡ A military version is planned; contact the nearest TI field sales office for availability.
§ Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

Table A-1. TMS320 Digital Signal Processor Part Commercial Numbers (Continued)

Device Name	Operating Frequency	Package Type	Typical§ Dissipation	Temp Range
TMS320C15FNA	20 MHz	Plastic 44-pin PLCC	225 mW	–40°/85°C
TMS320E15FZL-25	25 MHz	Ceramic 44-lead CERQUAD	325 mW	0°/70°C
TMS320LC15NA	16 MHz	Plastic 44-pin DIP	50 mW	–40°/85°C
TMS320LC15FNA	16 MHz	Plastic 44-pin PLCC	50 mW	–40°/85°C
TMS320LC15FNL	16 MHz	Plastic 44-pin PLCC	50 mW	0°/70°C
TMS320LC15NL	16 MHz	Plastic 40-pin DIP	50 mW	0°/70°C
TMS320P15FNL	20 MHz	Plastic 44-pin PLCC	275 mW	0°/70°C

TMS320P15FNL-25	25 MHz	Plastic 44-pin DIP	325 mW	0°/70°C
TMS320P15NA	20 MHz	Plastic 40-pin DIP	275 mW	−40°/85°C
TMS320P15NL	20 MHz	Plastic 40-pin DIP	275 mW	0°/70°C
TMS320P15NL-25	25 MHz	Plastic 40-pin DIP	325 mW	0°/70°C
TMS320C16PGL	35 MHz	Plastic 64-pin QFP	300 mW	0°/70°C
TMS320LC16PGL	16 MHz	Plastic 64-pin PQFP	100 mW	0°/70°C
TMS320C17FNA	20 MHz	Plastic 44-pin PLCC	250 mW	−40°/85°C
TMS320C17FNL	20 MHz	Plastic 44-pin PLCC	250 mW	0°/70°C
TMS320C17FNL-25	25 MHz	Plastic 44-pin PLCC	300 mW	0°/70°C
TMS320C17NA	20 MHz	Plastic 40-pin DIP	250 mW	−40°/85°C
TMS320LC17FNA	14 MHz	Plastic 44-pin PLCC	50 mW	−40°/85°C
TMS320LC17NA	14 MHz	Plastic 40-pin DIP	50 mW	−40°/85°C
TMS320C17NL	20 MHz	Plastic 40-pin DIP	250 mW	0°/70°C
TMS320C17NL-25	25 MHz	Plastic 40-pin DIP	300 mW	0°/70°C
TMS320E17FZA	20 MHz	Ceramic 44-lead CERQUAD	275 mW	−40°/85°C
TMS320E17FZL	20 MHz	Ceramic 44-lead CERQUAD	275 mW	0°/70°C
TMS320E17FZL-25	25 MHz	Ceramic 44-lead CERQUAD	340 mW	0°/70°C
TMS320E17JDA	20 MHz	Ceramic 40-pin DIP	275 mW	−40°/85°C
TMS320E17JDL	20 MHz	Ceramic 40-pin DIP	275 mW	0°/70°C
TMS320E17JDL-25	25 MHz	Ceramic 40-pin DIP	340 mW	0°/70°C

§ Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

Table A-1. TMS320 Digital Signal Processor Part Commercial Numbers (Continued)

Device Name	Operating Frequency	Package Type	Typical§ Dissipation	Temp Range
TMS320LC17FNL	14 MHz	Plastic 44-pin PLCC	50 mW	0°/70°C
TMS320LC17NL	14 MHz	Plastic 40-pin DIP	50 mW	0°/70°C
TMS320P17FNL	20 MHz	Plastic 44-pin PLCC	275 mW	−40°/85°C
TMS320P17FNL	20 MHz		275 mW	0°/70°C
TMS320P17FNL-25	25 MHz		340 mW	0°/70°C
TMS320P17FNA	20 MHz	Plastic 44-pin DIP	275 mW	−40°/85°C
TMS320P17NL	20 MHz	Plastic 40-pin DIP	275 mW	0°/70°C
TMS320P17NL-25	25 MHz		340 mW	0°/70°C
TMS320C25FNL	40 MHz	Plastic 68-pin PLCC	550 mW	0°/70°C
TMS320C25FNL-33	33 MHz		450 mW	0°/70°C
TMS320C25GBA	40 MHz	Ceramic 68-pin PGA	550 mW	−40°/85°C
TMS320C25GBL†	40 MHz		550 mW	0°/70°C
TMS320C25FNL-50†	50 MHz	Plastic 68-pin PLCC	700 mW	0°/70°C
TMS320C25PHL	40 MHz	Plastic 80-pin PQFP	550 mW	0°/70°C
TMS320C25PHL-33	33 MHz		475 mW	0°/70°C
TMS320P25FNA	40 MHz	Plastic 68-pin PLCC	550 mW	−40°/85°C
TMS320C26FNL†	40 MHz		550 mW	0°/70°C
TMS320C28FNL	40 MHz		550 mW	0°/70°C
TMS320C28FNL-50	50 MHz		700 mW	0°/70°C
TMS320C28PHL	40 MHz	Plastic 80-pin PQFP	550 mW	0°/70°C

TMS320C28PHL-50	50 MHz		700 mW	0°/70°C
TMS320C30GEL-27	27 MHz	Ceramic 181-pin PGA	875 mW	0°/70°C
TMS320C30GEL†	33 MHz		1000 mW	0°/70°C
TMS320C30GEL-40	40 MHz		1250 mW	0°/70°C
TMS320C30GEL-50	50 MHz		1500 mW	0°/70°C
TMS320C30PPML-27	27 MHz	Plastic 208-pin PQFP	575 mW	0°/70°C
TMS320C30PPML	33 MHz		700 mW	0°/70°C
TMS320C30PPML-40	40 MHz		850 mW	0°/70°C
TMS320C30PPML-50	50 MHz		1000 mW	0°/70°C

† A military version is available.
§ Calculated from typical I_{CC} current and nominal V_{CC} supply voltage.

Table A-1. TMS320 Digital Signal Processor Part Commercial Numbers (Continued)

Device Name	Operating Frequency	Package Type	Typical§ Dissipation	Temp Range
TMS320C31PQL-27	27 MHz	Plastic 132-pin PQFP	600 mW	0°/70°C
TMS320LC31PQL	33 MHz		500 mW	0°/70°C
TMS320C31PQL-40	40 MHz		900 mW	0°/70°C
TMS320C31PQL-50	50 MHz		1100 mW	0°/70°C
TMS320C31PQA-27	27 MHz		625 mW	−40°/85°C
TMS320C31PQA	33 MHz		750 mW	−40°/85°C
TMS320C40GFL	50 MHz	Ceramic 325-pin PGA	1500 mW	0°/70°C
TMS320C40GFL-40	40 MHz		1200 mW	0°/70°C
TMS320C50PQ	40 MHz	Plastic 132-pin PQFP	500 mW	0°/70°C
TMS320C50PQ-57	57 MHz	Plastic 132-pin PQFP	650 mW	0°/70°C
TMS320C50PQ-80	80 MHz		750 mW	0°/70°C
TMS320C50PQA	40 MHz		500 mW	−40°/85°C
TMS320C50PQA-57	57 MHz		650 mW	−40°/85°C
TMS320C51PQ	40 MHz		500 mW	0°/70°C
TMS320C51PQ-57	57 MHz		650 mW	0°/70°C
TMS320BC51PQ	40 MHz		500 mW	0°/70°C
TMS320BC51PQ-57	57 MHz		650 mW	0°/70°C
TMS320C51PQ-80	80 MHz	Plastic 132-pin PQFP	750 mW	0°/70°C
TMS320C51PQA	40 MHz		500 mW	−40°/85°C
TMS320C51PQA-57	57 MHz		650 mW	−40°/85°C
TMS320C51PZ	40 MHz	Plastic 100-pin TQFP	500 mW	0°/70°C
TMS320C51PZ-80	80 MHz		750 mW	0°/70°C
TMS320C51PZA	40 MHz		500 mW	−40°/85°C
TMS320C51PZA-57	57 MHz		650 mW	−40°/85°C
TMS320BC51PQ-57	57 MHz	Plastic 132-pin PQFP	650 mW	−40°/85°C
TMS320BC51PQ-80	80 MHz		750 mW	0°/70°C
TMS320BC51PQA	40 MHz		500 mW	−40°/85°C

† A military version is planned.
§ Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

Table A-1. TMS320 Digital Signal Processor Part Commercial Numbers (Continued)

Device Name	Operating Frequency	Package Type	Typical§ Dissipation	Temp Range
TMS320C51PQL	40 MHz	Plastic 132-pin QFP¶	250 mW	0°/70°C
TMS320BC51PZ	40 MHz	Plastic 100-pin TQFP	500 mW	0°/70°C
TMS320BC51PZ-57	57 MHz		650 mW	0°/70°C
TMS320C51PZ-80	80 MHz		750 mW	0°/70°C
TMS320C51PZA	40 MHz		500 mW	−40°/85°C
TMS320C51PZA-57	57 MHz		650 mW	−40°/85°C
TMS320C52PJ	40 MHz	Plastic 100-pin PQFP	500 mW	0°/70°C
TMS320C52PJ-57	57 MHz		650 mW	0°/70°C
TMS320BC52PJ	40 MHz		500 mW	0°/70°C
TMS320BC52PJ-57	57 MHz		650 mW	0°/70°C
TMS320C52PJ-80	80 MHz		750 mW	0°/70°C
TMS320C52PJA	40 MHz		500 mW	−40°/85°C
TMS320C52PJA-57	57 MHz		650 mW	−40°/85°C
TMS320BC52PJA	40 MHz		500 mW	−40°/85°C
TMS320C52PZ	40 MHz	Plastic 100-pin TQFP	500 mW	0°/70°C
TMS320C52PZ-57	57 MHz		650 mW	0°/70°C
TMS320C52PZ-80	80 MHz		750 mW	0°/70°C
TMS320C52PZA	40 MHz		500 mW	−40°/85°C
TMS320C52PZA-57	57 MHz		650 mW	−40°/85°C
TMS320BC52PQ	40 MHz	Plastic 100-pin PQFP	500 mW	0°/70°C
TMS320BC52PQ-57	57 MHz		650 mW	0°/70°C
TMS320BC52PQ-80	80 MHz		750 mW	0°/70°C
TMS320BC52PQA	40 MHz		500 mW	−40°/85°C
TMS320BC52PQA-57	57 MHz		650 mW	−40°/85°C
TMS320BC52PZ	40 MHz	Plastic 100-pin TQFP	500 mW	0°/70°C
TMS320BC52PZ-57	57 MHz		650 mW	0°/70°C
TMS320BC52PZ-80	80 MHz		750 mW	0°/70°C
TMS320BC52PZA	40 MHz		500 mW	−40°/85°C
TMS320BC52PZA-57	57 MHz		650 mW	−40°/85°C

§ Calculated from typical I_{CC} current and nominal V_{CC} supply voltage
¶ Proposed package; initially in ceramic 132-pin QFP packages (part number TMS320C51HTL)

Table A-1. TMS320 Digital Signal Processor Part Commercial Numbers (Concluded)

Device Name	Operating Frequency	Package Type	Typical§ Dissipation	Temp Range
TMS320C53PQ	40 MHz	Plastic 132-pin PQFP	500 mW	0°/70°C
TMS320C53PQ-57	57 MHz		650 mW	0°/70°C
TMS320C53PQ-80	80 MHz		750 mW	0°/70°C
TMS320BC53PQ	40 MHz		500 mW	0°/70°C
TMS320BC53PQ-57	57 MHz		650 mW	0°/70°C
TMS320C53PQA	40 MHz		500 mW	−40°/85°C
TMS320C53PQA-57	57 MHz		650 mW	−40°/85°C
TMS320C53PZ	40 MHz	Plastic 100-pin TQFP	500 mW	0°/70°C
TMS320C53PZ-57	57 MHz		650 mW	0°/70°C
TMS320C53PZ-80	80 MHz		750 mW	0°/70°C

TMS320C53PZA	40 MHz		500 mW	–40°/85°C
TMS320C53PZA-57	57 MHz		650 mW	–40°/85°C
TMS320BC53PQA	40 MHz	Plastic 132-pin PQFP	500 mW	–40°/85°C
TMS320BC53PQ-57	57 MHz	Plastic 100-pin PQFP	650 mW	0°/70°C
TMS320BC53PQ-80	80 MHz		750 mW	0°/70°C
TMS320BC53PZ	40 MHz	Plastic 100-pin TQFP	500 mW	0°/70°C
TMS320BC53PZ-57	57 MHz		650 mW	0°/70°C
TMS320BC53PZ-80	80 MHz		750 mW	0°/70°C
SMJ320C10JDM	20.5 MHz	Ceramic 40-pin DIP	165 mW	–55°/125°C
SMJ320C10-25JDM	25.6 MHz		200 mW	–55°/125°C
SMJ320C15JDM	20.5 MHz		165 mW	–55°/125°C
SMJ320C15-25JDM	25.6 MHz		200 mW	–55°/125°C
SMJ320E15JDM	20.5 MHz		275 mW	–55°/125°C
SMJ320E14GBM	20.5 MHz	68-pin PGA	325 mW	–55°/125°C
SMJ320E14FJM	20.5 MHz	68-pin JLCC	325 mW	–55°/125°C
SMJ320C25GBM	40.0 MHz	68-pin PGA	550 mW	–55°/125°C
SMJ320C25FJM	40.0 MHz	68-pin JLCC	550 mW	–55°/125°C
SMJ320C25FDM	40.0 MHz	68-pin LCCC	550 mW	–55°/125°C

§ Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

Table 2. TMS320 Digital Signal Processor Military Part Numbers

Device Name	Operating Frequency	Package Type	Typical§ Dissipation	Temp Range
SMJ320C25-50GBM	50.0 MHz	68-pin PGA	700 mW	–55°/125°C
SMJ320C25-50FJM	50.0 MHz	68-pin JLCC	700 mW	–55°/125°C
SMJ320C26BGBM	40.0 MHz	68-pin PGA	550 mW	–55°/125°C
SMJ320C26BFJM†	40.0 MHz	68-pin JLCC	550 mW	–55°/125°C
SMJ320C30GBM-25	25.0 MHz	181-pin PGA	1100 mW	–55°/125°C
SMJ320C30HFG-25	25.0 MHz	196-pin CQFP	1100 mW	–55°/125°C
SMJ320C30GBM-28	28.6 MHz	181-pin PGA	1000 mW	–55°/125°C
SMJ320C30HFG-28	28.6 MHz	196-pin CQFP	1000 mW	–55°/125°C
SMJ320C30GBM-33	33.3 MHz	181-pin PGA	1100 mW	–55°/125°C
SMJ320C30HFG-33	33.3 MHz	196-pin CQFP	1100 mW	–55°/125°C
SMJ320C31GFA-27	27 MHz	141-pin PGA	600 mW	–55°/125°C
SMJ320C31GFA-33	33 MHz	141-pin PGA	750 mW	–55°/125°C
SMJ320C31HFG-27	27 MHz	132-pin CQFP	600 mW	–55°/125°C
SMJ320C31HFG-33	33 MHz	132-pin CQFP	750 mW	–55°/125°C
SMJ320C40GF-40†	40.0 MHz	325-pin PGA	1750 mW	–55°/125°C
SMJ320C40GFM-33	33.3 MHz	325-pin PGA	1000 mW	–55°/125°C
SMJ320C40HFH-40†	40.0 MHz	352-pin CQFP	1750 mW	–55°/125°C
SMJ320C40HFHM-33	33.3 MHz	352-pin CQFP	1000 mW	–55°/125°C
SMJ320C50AGFAM-40	40 MHz	141-pin PGA	350 mW	–55°/125°C
SMJ320C50AGFAM-50	50 MHz	141-pin PGA	450 mW	–55°/125°C
SMJ320C50AHFGM-40	40 MHz	132-pin CQFP	350 mW	–55°/125°C
SMJ320C50AHFGM-50	50 MHz	132-pin CQFP	450 mW	–55°/125°C

‡ A military version is planned; contact the nearest TI field sales office for availability.

§ Calculated from typical I_{CC} current and nominal V_{CC} supply voltage

Table 3. TMS320 Code-Generation Tools

Generation Name	Product	Part Number
TMS320C1x, TMS320C2x and TMS320C5x	Assembler/Linker (supports TMS320C1x/C2x/C5x) ■ PC (MS-DOS)	TMDS3242850–02
	Digital Filter Design Package (DFDP) ■ PC (MS-DOS)	DFDP
TMS320C3x and TMS320C4x	C Compiler (supports TMS320C3x/C4x) ■ PC (MS-DOS or OS/2) ■ Sun-3/-4 (UNIX)‡	TMDS3243855–02 TMDS3243555–08
	’C3x Tartan C++ Compiler ■ ’C3x Tartan C++(MS-DOS) ■ ’C3x Tartan C++(Sun-3/4) ■ Tartan Floating Point Library (MS-DOS) ■ Tartan Floating Point Library (Sun-3/4)	 TAR–CCM–PC–3X TAR–CCM–SP–3X 320FLO–PC–C30 320FLO–SUN–C30
	’C3x Tartan C++ Compiler and Simulator ■ ’C3x Tartan C++(MS-DOS) ■ ’C3x Tartan C++(Sun-3/4)	 TAR–SIM–PC–C3X TAR–SIM–SP–C3X
	’C4x Tartan C++ Compiler ■ ’C4x Tartan C++(MS-DOS) ■ ’C4x Tartan C++(Sun-3/4) ■ Tartan Floating Point Library (MS-DOS) ■ Tartan Floating Point Library (Sun-3/4)	 TAR–CCM–PC–4X TAR–CCM–SP–4X 320FLO–PC–C40 320FLO–SUN–C40
	’C4x Tartan C++ Compiler and Simulator ■ ’C4x Tartan C++(MS-DOS) ■ ’C4x Tartan C++(Sun-3/4)	 TAR–SIM–PC–C4X TAR–SIM–SP–C4X
	Assembler/Linker (supports TMS320C3x/C4x) ■ PC (MS-DOS)	TMDS3243850–02
	Digital Filter Design Package (DFDP) ■ PC (MS-DOS)	DFDP

‡ In TAR format

Table 4. TMS320 System Integration and Debugging Tools

Generation Name	Product	Part Number
TMS320C1x	Software Simulator ■ PC (MS-DOS)	TMDS3240811–02
	Evaluation Board ■ ’C16 Evaluation Module (EVM)	TMDS3260016
	Emulator ■ TMS320C10/C15 XDS/22 ■ TMS320C14 XDS/22 ■ TMS320C17 XDS/22	 TMDS3262215 TMDX3262214 TMDS3262217
	EPROM Programming Adapter Socket ■ 40-pin-to-28-pin conversion (supports TMS320E15/E17/P15/P17) ■ 68-pin-to-28-pin conversion (supports TMS320E14/P14) ■ 44-pin-to-28-pin conversion (supports TMS320E15/E17/P15/P17)	 RTC/PGM320A–06 TMDX3270110 RTC/PGM320C–06
	Additional Target Cable ■ 44-pin PLCC (supports TMS320C10/C15/C17)	TMDS3288810

TMS320C2x	Software Simulator ■ PC (MS-DOS) ■ Sun-3/4	 TMDS3242851–02 TMDS3242551–09
	Evaluation Board ■ ’C26 Evaluation Module (EVM) ■ ’C26 DSP Starter Kit (DSK)	 TMDS3260026 TMDS3200026
	Emulator ■ TMS320C25/C26 XDS/22‡	 TMDS3262221

(Continued on the next page)

‡ 50-MHz emulation is supported by Macrochip Research Inc.; refer to the *TMS320 Third-Party Support Reference Guide* (literature number SPRU052)

NO TAG. TMS320 System Integration and Debugging Tools (Concluded)

Generation Name	Product	Part Number
TMS320C2x	EPROM Programming Adapter Socket ■ 68-pin-to-28-pin conversion (supports TMS320E25)	TMDX3270120
TMS320C3x	Software Simulator ■ PC (MS-DOS, MS-Windows) ■ Sun-3/4	 TMDS3243851–02 TMDS3243551–09
	’C30 Evaluation Board ■ Evaluation Module (EVM)‡	 TMDS3260030
	Emulator ■ XDS510 (MS-DOS, MS-Windows) ■ XDS510WS (Sun-3/4) ■ XDS510 PC/SPARC Emulation Cable (Conversion of ’C4x, ’C5x, to ’C3x) ■ Emulation Porting Kit (’C3x)§	 TMDS3260130 TMDS3260630 TMDS3080003 TMDX3240030
TMS320C4x	Software Simulator ■ PC (MS-DOS, MS-Windows) ■ Sun-3/4	 TMDS3244851–02 TMDS3244551–09
	TMS320C4x Quad Target Board ■ Parallel Processing Development System (PPDS)	 TMDX3261040
	Emulator ■ XDS510 (MS-DOS, MS-Windows, OS/2) ■ XDS510WS (Sun-3/4) ■ XDS PC/SPARC Emulation Cable (Conversion of ’C3x to ’C4x) ■ Emulation Porting Kit (’C4x)§	 TMDS3260140 TMDS3260640 TMDS3080002 TMDX3240040
TMS320C5x	Software Simulator ■ Workstation (Sun-3/4) ■ PC (MS-DOS, MS-Windows)	 TMDS3245551–09 TMDS3245851–02

(Continued on the next page)

† In backup format
‡ Applies to ’C30 only
§ Requires a license

NO TAG. TMS320 System Integration and Debugging Tools (Concluded)

Generation Name	Product	Part Number
TMS320C5x	Emulator ■ XDS510 (MS-DOS, OS/2, MS-Windows) ■ XDS510 (Sun-3/4) ■ XDS PC/SPARC Emulation Cable (Conversion of ’C3x to ’C5x) ■ Emulation Porting Kit (’C5x)§	 TMDS3260150 TMDS3260650 TMDS3080002 TMDX3240050
	Evaluation Board ■ ’C50 Evaluation Module (EVM) ■ ’C50 DSP Starter Kit (DSK)	 TMDS3260050 TMDS3200051

§ Requires a license

1.2 Device and Development Support Tool Prefix Designators

To classify the stages in the product development cycle, Texas Instruments assigns prefix designators in the part number nomenclature. There are three device prefix designators—TMX, TMP, and TMS—and three development support tool prefix designators—TMDX, TMDS, and TMDC. These prefixes represent the evolutionary stages of product development from engineering prototypes (TMX/TMDX) through fully qualified production devices (TMS/TMDS). This development flow is defined below.

Device Development Evolutionary Flow

TMX Experimental device that is not representative of the final device's electrical specifications

TMP Final silicon die that conforms to the device's electrical specifications but has not completed quality and reliability verifications

TMS Fully qualified production device

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 against the following disclaimer:

Developmental product intended for internal evaluation purposes.

Note:

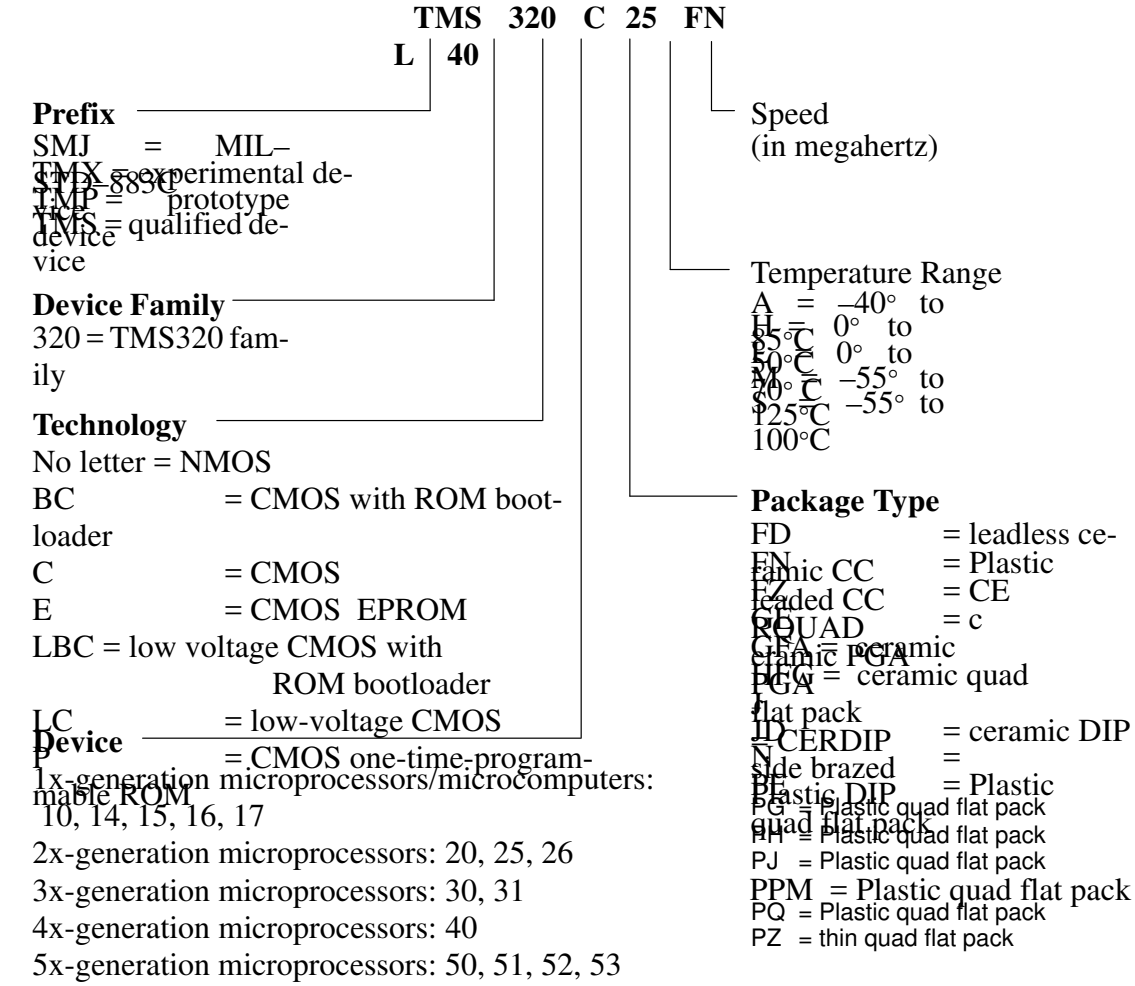
Texas Instruments recommends that prototype devices (TMX or TMP) not be used in production systems, because their expected end-use failure rate is undefined but predicted to be greater than the failure rate of standard, qualified production devices.

TMS devices and 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.

1.3 Device Nomenclature

In addition to the prefix *TMS*, the device nomenclature includes a two-part suffix that follows the device's family name. This suffix indicates the package type (e.g., FD) and temperature range (e.g., A). NO TAG provides a legend for reading the complete nomenclature of any TMS320 family member.

Figure 1. TMS320 Device Nomenclature



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.

Note:

If TI accepts your unit for repair and you asked 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.

Topic **Page**

2.1 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.

Note:

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 customer's expense, any product that cannot be tested to its specifications because of customer's modifications.

2.2 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 that the product will be promptly returned to you.

2.3 System Updates

The standard TI practice is to update each Microprocessor Development Systems product that is returned for repair to the current revision of make/model. Extra charges may apply to some updates.

2.4 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 (713) 274–2285 and ask for a *Return Material Authorization* (RMA) number.

If you reside outside of the United States or Canada, contact the nearest Regional Technology Center 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:

- Factory Repair's *Return Material Authorization* 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 ESD 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.

2.5 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 will exceed the standard fixed-price rates. 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 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.

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-disc 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 & 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

ROM Codes

Size of a printed circuit board must be considered in many DSP applications. To fully utilize the board space, Texas Instruments offers three options that reduce the chip count and provide a single-chip solution to its customers. These options are a mask-programmable ROM, an EPROM, and a one-time-programmable (OTP) EPROM. This allows you to use a code-customized processor for a specific application while taking advantage of the following:

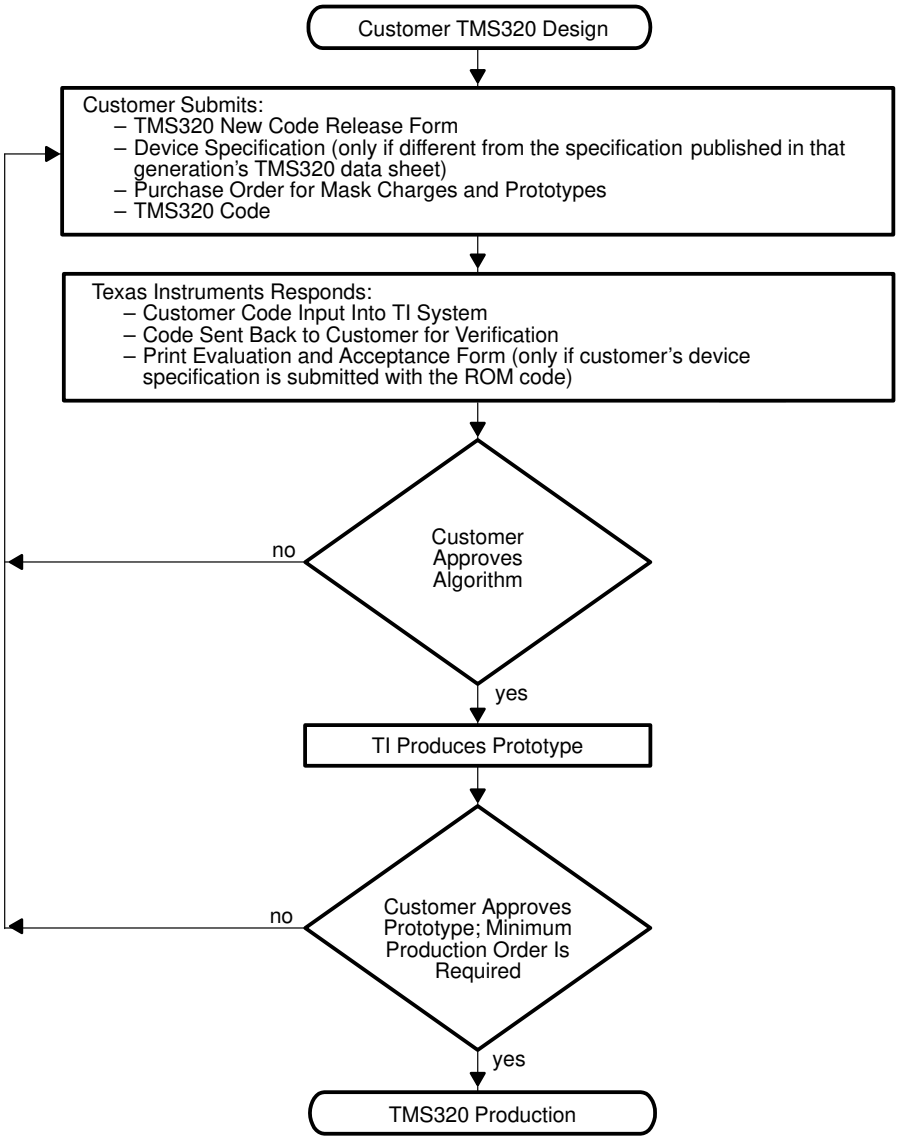
- Greater memory expansion
- Lower system cost
- Less hardware and wiring
- Smaller PCB

A repetitive routine or an entire algorithm can be programmed into the on-chip ROM of a TMS320 DSP. TMS320 programs can also be expanded via external memory; this reduces the chip count and allows for a more flexible program memory. Multiple functions are easily implemented by a single device, thus enhancing the system's capabilities.

TMS320 development tools are used to develop, test, refine, and finalize the algorithms. The microprocessor/microcomputer mode (MC/$\overline{\text{MP}}$ for TMS320C1x, TMS320C3x, and TMS320C4x; MP/$\overline{\text{MC}}$ for TMS320C2x and TMS320C5x) is available on all ROM-coded TMS320 DSP devices when accessing either on-chip or off-chip memory 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, either the TMS320 may once again be used in the microprocessor mode or the windowed/OTP EPROM version of that TMS320 device may be inserted with a code to manage the transition to a new ROM code. This shortens the field upgrade time and avoids the possibility of inventory obsolescence.

Figure 1 illustrates the procedural flow for TMS320 masked parts. When ordering, there is a one-time non-refundable charge for mask tooling. A minimum production order per year is required for any masked-ROM device. A ROM code will be deleted from the TI system one year after the last delivery date.

Figure 1. TMS320 ROM Code Flowchart



Digital signal processors with the windowed or the one-time-programmable EPROM option offer a solution toward low-/high-volume orders, respectively. These EPROM options allow for form-factor emulation. Field upgrades and changes are possible with any EPROM option.

A TMS320 ROM code may be submitted in one of the following formats:

- ❑ 5.25" or 3.5" floppy: TI-tagged Intel or common object file format (COFF) from cross-assembler/linker
- ❑ EPROM (TMS320): TMS320E14/P14, TMS320E15/P15, TMS320E17/P17, or TMS320E25
- ❑ EPROM (others): TMS27C64
- ❑ Modem (BBS): TI-tagged Intel or COFF format from cross-assembler/linker

Note:

When submitting 'C3x or 'C5x code, please include the COFF file only.

When a code is submitted to Texas Instruments for masking, the code is reformatted to accommodate the TI mask-generation system. System-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. Those formatting changes involve both the removal of address relocation information (the code address begins at the base address of the ROM in the TMS320 device and progresses without gaps to the last address of the ROM on the TMS320 device) and the addition of data into the reserved locations of the ROM for a device ROM test. Note that because these changes have been made, a check-sum comparison is not a valid means of verification.ROM code algorithms may also be submitted by secure electronic transfer via a modem. Contact the nearest TI Field Sales Office for further information.

With each masked-device order, the customer must sign a disclaimer stating:

The units to be shipped against this order were assembled, for expediency purposes, on a prototype (i.e., nonproduction qualified) manufacturing line, the reliability of which is not fully characterized. Therefore, the anticipated inherent reliability of these prototype units cannot be expressly defined.

and a release stating:

Any masked ROM device may be resymbolized as a TI standard product and resold as though it were an unprogrammed version of the device at the convenience of Texas Instruments.

Contact the nearest TI Field Sales Office for more information on procedures, lead times, and cost.

Appendix 5

Quality and Reliability

Texas Instruments has developed a leadership reliability qualification system, based upon years of experience with leading-edge memory technology and upon years of research to better meet its customers' requirements. In order to achieve constant improvement, programs that support this system respond to customer inputs and internal information.

5.1 TI's Quality Commitment

The quality and reliability of Texas Instruments microprocessor and microcontroller products, which include the TMS320 digital signal processors, rely upon feedback from:

- Customers
- Overall manufacturing operation from front-end wafer fabrication to final shipping inspection to improve product yield and quality
- Product quality and reliability monitoring

Our customers' perception of quality is the governing criterion for judging performance. This concept is the basis for Texas Instruments Corporate Quality Policy, which is as follows:

For every product or service we offer, we shall define the requirements that solve the customer's problems, and we shall conform to those requirements without exception.

5.2 Reliability Stress Tests

Accelerated stress tests are performed on new semiconductor products (except TMD products and devices that are used on those products). Process changes are made to qualify the products and to ensure excellence in product reliability. These test environments are typical:

- ❑ High-temperature operating life
- ❑ Storage life
- ❑ Temperature cycling
- ❑ Biased humidity
- ❑ Autoclave
- ❑ Electrostatic discharge
- ❑ Package integrity
- ❑ Electromigration
- ❑ Channel-hot electrons (performed on geometries of less than 2.0 μm)

Typical events or changes that require internal requalification of a product include:

- ❑ New die design, shrink, or layout
- ❑ Wafer process (baseline/control systems, flow, mask, chemicals, gases, dopants, passivation, or metal systems)
- ❑ Packaging assembly (baseline control systems or critical assembly equipment)
- ❑ Piece parts (lead frame, mold compound, mount material, bond wire, or lead finish)
- ❑ Manufacturing site

TI reliability control systems extend beyond qualification. Total reliability controls and management include product reliability monitoring as well as final product release controls. MOS memories, utilizing high-density active elements, serve as the leading indicator in wafer-process integrity at TI MOS fabrication sites, enhancing all MOS logic device yields and reliability. TI places several thousand MOS devices per month through stress tests to ensure and sustain built-in product excellence.

Table 1 lists the microprocessor and microcontroller reliability tests, duration of specific tests, and sample size. Some definitions and descriptions relating to those tests follow.

- **AOQ (Average Outgoing Quality)** — amount of defective products in a population, usually expressed in terms of parts per million (PPM)
- **FIT (Failure In Time)** — estimated field failure rate in number of failures per billion power-on device hours; 1000 FITS equals 0.1% failure per 1000 device hours
- **Operating Life** — device dynamically exercised at a high ambient temperature (usually 125°C) to simulate field usage that exposes the device to a much lower ambient temperature (such as 55°C); using a derived high temperature, a 55°C ambient failure rate can be calculated
- **High-Temperature Storage** — device exposed to 150°C, unbiased conditions; bond integrity is stressed within this environment
- **Biased Humidity** — moisture and bias used to accelerate corrosion-type failures in plastic packages; conditions include 85°C ambient temperature with 85% relative humidity (RH); typical bias voltage is +5 V_{DC}, grounded on alternating pins
- **Autoclave (pressure cooker)** — plastic-packaged devices exposed to moisture at 121°C, using a pressure rating of one atmosphere above normal pressure; the pressure forces moisture permeation of the package and accelerates corrosion mechanisms (if present) on the device; also, external package contaminants can be activated and caused to generate inter-pin current leakage paths
- **Temperature Cycle** — device exposed to severe temperature extremes in an alternating fashion (–65°C for 15 minutes and 150°C for 15 minutes per cycle) for at least 1000 cycles; package strength, bond quality, and consistency of assembly process are tested within this environment
- **Electrostatic Discharge** — Device exposed to electrostatic discharge pulses (calibration per MIL STD 883C, Method 3015.6); device is stressed to determine the failure threshold of the design
- **Thermal Shock** — test similar to the temperature cycle test, but involves liquid-to-liquid transfer (per MIL-STD-883C, Method 1011)
- **PIND (Particle Impact Noise Detection)** — a nondestructive test to detect loose particles inside the device's cavity

❑ **Mechanical Sequence**

Fine and gross leak	per MIL-STD-883C, Method 1014
Mechanical shock	per MIL-STD-883C, Method 2002, 1500 g, 0.5 ms, Condition B
PIND (optional)	per MIL-STD-883C, Method 2020
Vibration, variable frequency	per MIL-STD-883C, Method 2007, 20 g, Condition A
Constant acceleration	per MIL-STD-883C, Method 2001, 20 kg, Condition D, *Y1* Plane min
Electrical test	per data sheet limits

❑ **Thermal Sequence**

Fine and gross leak	per MIL-STD-883C, Method 1014
Solder heat (optional)	per MIL-STD-750C, Method 1014
Temperature cycle (10 cycles minimum)	per MIL-STD-883C, Method 1010, –65°C to +150°C, Condition C
Thermal shock (10 cycles minimum)	per MIL-STD-883C, Method 1011, –55°C to +125°C, Condition B
Moisture resistance	per MIL-STD-883C, Method 1004
Electrical test	per data sheet limits

❑ **Thermal/Mechanical Sequence**

Fine and gross leak	per MIL-STD-883C, Method 1014
Temperature cycle (10 cycles minimum)	per MIL-STD-883C, Method 1010, –65°C to +150°C, Condition C
Constant acceleration	per MIL-STD-883C, Method 2001, 30 kg, *Y1* Plane
Electrical test	per data sheet limits
Electrostatic discharge	per MIL-STD-883C, Method 3015
Solderability	per MIL-STD-883C, Method 2003
Solder heat	per MIL-STD-750C, Method 2031, 10 sec
Salt atmosphere	per MIL-STD-883C, Method 1009, Condition A, 24 hrs min
Lead pull	per MIL-STD-883C, Method 2004, Condition A
Lead integrity	per MIL-STD-883C, Method 2004, Condition B1
Electromigration	Accelerated stress testing of conductor patterns to ensure acceptable life-time of power-on operation
Resistance to solvents	per MIL-STD-883C, Method 2015

Table 1. Microprocessor and Microcontroller Tests

Test	Duration	Sample Size	
		Plastic	Ceramic
Operating life, 125°C, 5.0 V	1000 hrs	129	129
Storage life, 150°C	1000 hrs	77†	77
Biased 85°C / 85% RH, 5.0 V	1000 hrs	129	–
Autoclave, 121°C, 1 ATM	240 hrs	77	–
Temperature cycle, –65°C to 150°C	1000 cyc‡	129	129
Temperature cycle, 0°C to 125°C	3000 cyc	129	129
Thermal shock, –65°C to 150°C	200 cyc	129	129
Electrostatic discharge, ±2 kV		12	12
Latch-up (CMOS devices only)		5	5
Mechanical sequence		–	38
Thermal sequence		–	38
Thermal/mechanical sequence		–	38
PIND		–	45
Internal water vapor		–	3
Solderability		22	22
Solder heat		22	22
Resistance to solvents		15	15
Lead integrity		15	15
Lead pull		22	–
Lead-finish adhesion		15	15
Salt atmosphere		15	15
Flammability (UL94-V0)		3	–
Thermal impedance		5	5

† If junction's temperature does not exceed plasticity of package
‡ For severe environments; for office environments, number of cycles will be less

TMS320 EPROM Programming

The EPROM versions of the TMS320 family include a 4K-word×16-bit EPROM implemented with an industry-standard EPROM 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. EPROM adapter sockets are available that provide 40-pin-to-28-pin conversion for programming the TMS320E15/P15 and TMS320E17/P17. Adapter sockets are also available to provide the 68-pin-to-28-pin conversion required for programming the TMS320E14/P14 and TMS320E25. Refer to NO TAG in Appendix A, *TMS320 Product Information,* for part numbers.

Key features of the EPROM cell include standard programming and verification. The EPROM 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 EPROM contents. This appendix describes programming, erasure, version verification, and EPROM security.

Topic **Page**

6.1 Programming

The TMS320 EPROM 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 EPROM 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 EPROM 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.

The EPROM 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 EPROM 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.

6.2 Erasure

Before it is programmed, the TMS320 EPROM device must be erased by exposing the chip (through the transparent lid) to a high-intensity, ultraviolet-light source (wavelength of 2537 angstroms); after erasure, all bits are in a logic **1** state. Note that normal ambient light contains ultraviolet. Therefore, when a TMS320 EPROM device is used, its transparent window should be covered with an opaque tape or label after erasure. This prevents the degradation and/or erasure of the programmed data. Information on recommended exposure time and lid-to-light distance for a particular TMS320 EPROM device is furnished in the appropriate TMS320 user's guide.

6.3 Fast and SNAP! Pulse Programming

Two programming algorithms are available for TMS320 EPROM devices. The *fast* programming algorithm is normally used to program the entire EPROM contents, although individual locations may be programmed separately. Fast programming is supported on the TMS320E14/P14, TMS320E15/P15, TMS320E17/P17, and TMS320E25. 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 TMS320E14/P14 and TMS320E25. For more information on these two programming algorithms, consult the appropriate TMS320 user's guide.

6.4 Version Verification

Information on verification of the EPROM versions of the TMS320 family is contained in the appropriate TMS320 user's guide.

6.5 EPROM Security

The EPROM protection mechanism completely disables a TMS320 device and prevents reading of the EPROM contents. This guarantees the security of proprietary algorithms. This facility is implemented through a unique EPROM cell called the RBIT (ROM protect bit) cell. Once the contents are programmed into the EPROM, the RBIT can be programmed, preventing access to the EPROM contents and disabling the microprocessor mode on the TMS320 device. Once programmed, the RBIT can only be cleared 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.

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X