EmbeddedBIOS® 2000

Version 5.3

Firmware USER'S GUIDE

AMD Geode™ DBSC1200 Development Board

The Best Firmware for

Embedded x86 Development

GENERAL® SOFTWARE

11000 NE 33rd Place, Suite 102, Bellevue, WA 98004
425.576.8300 (Outside U.S.)
800.850.5755 (U.S. & Canada)
support@gensw.com
www.gensw.com

IMPORTANT NOTICES

Please complete and return your **Product Registration** card immediately. This will help us to notify you of updates, and make you eligible to receive technical support and access to on-line services.

Important Licensing Information

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introducing Embedded BIOS® 2000

General Software's Embedded BIOS® 2000 brand BIOS (Basic Input/Output System) pre-boot firmware is the industry's standard product used by most designers of embedded x86 computer equipment in the world today. Its superior combination of configurability and functionality enables it to satisfy the most demanding ROM BIOS needs for embedded designers. Its modular architecture and high degree of configurability make it **the most flexible BIOS in the world**.

General Software understands the need for expandability and serviceability by the OEM in their embedded designs. That's why General Software is proud to introduce **Firmbase®** technology for AMD™ Geode™ solutions. Firmbase's full 32-bit capabilities replace AMD's proprietary 16-bit SMM handler, previously known as VSA2 (Virtual System Architecture™ technology), on these designs. With Firmbase technology, your AMD Geode processor design can now take full advantage of these features and more:

- Full Legacy USB support for Keyboard and Mouse emulation
- Full USB Mass Storage support for boot from floppy and hard disk drives as well as CDROM, DiskOnKey and Compact Flash readers (licensed separately)
- Full Power Management including ACPI (S0 – S5 including S3 Suspend to RAM support) and APM (full-on, standby, suspend to RAM, soft-off)
- Full Hardware Emulation (Virtual PCI)
- Soft VGA for both CRT and TFT devices
- Integrated Audio
- Full support for AMD video and audio device drivers
- Firmware level TCP/IP stack (sold separately)
- Fully expandable 32-bit environment allows OEMs to develop custom firmware using 32-bit Windows development tools.
- Full support from your BIOS vendor for BIOS components

1.1.1 Customizing Firmware with Embedded BIOS 2000

With your Embedded BIOS 2000 Adaptation Kit, we send you virtually the same source code used by General Software for validation and release purposes so you know you will be start from known working source for your baseline thus reducing your risk and keeping you only one step away from a working BIOS.

1.1.2 AMD and General Software

General Software's commitment to technological innovation and customer service is seen throughout the industry making General Software and Embedded BIOS 2000 the best firmware of choice for your next Geode solutions-based design.

> *"AMD has come to expect outstanding service from General Software, both internally when we work on new products together ..., and in the field, when they work with our embedded customers," said Mark Morneault, Division Manager, Systems Engineering and Strategic Initiatives at AMD. "General Software's commitment to helping all AMD customers, going well out of their way to ensure a successful design, makes them a valued long-term strategic partner for AMD."*

General Software also offers full deployment and support services with the industry's top x86 BIOS engineers to reduce your risk and development overhead so you can bring your design to market quickly and cost efficient manner.

1.1.3 Focus on Embedded

Although Embedded BIOS 2000 can be made to behave like a PC BIOS by the company that adapts it for a given hardware platform, it has a fundamentally different role from that of a PC BIOS. The goal of a PC BIOS is to provide the same standard user experience for the systems it runs on. This allows a wide range of users to intuitively use any notebook computer, regardless of brand, for example. The goal of Embedded BIOS 2000 is completely different—it allows the embedded equipment manufacturer to differentiate its product from competing products in a way that is quick and cost effective—allowing the manufacturer to pass savings on to the end user.

This operating guide describes the possible user features of Embedded BIOS 2000. The manufacturer of a given piece of equipment may choose to incorporate some or all of these features, or modify them as necessary to fit a given application. Therefore, some of the features discussed in this operating guide may not apply to your system.

1.2 Overview of Embedded BIOS 2000 Features

Serving the entire 32-bit embedded 80x86-based market, Embedded BIOS 2000 offers special-purpose features not provided by typical PC BIOS implementations.

CPU Support
With embedded CPU support, virtually any type of CPU can be supported, provided it is reasonably compatible with the Intel 80386 instruction set. For example, the AMD Geode SC1200 processor and other CPUs are supported by Embedded BIOS 2000 when the associated

CPU-specific personality module is selected for the project. This allows support of high-integration processors that have on-board timers, DMA controllers, serial ports, watchdog timers, power management, and other features. Embedded BIOS 2000 supports the following Geode processor family products:

- NX 1750@14W, NX 1500@6W
- GX 533@1.1W, GX 500@1.0W, GX 466@0.9W
- SC1100
- SC1200/1201, SC2200, SC3200
- GX1

Additional AMD CPU-level products supported by Embedded BIOS 2000:

- SC300/310
- SC400/410
- SC520
- K6

Chipset Support
With chipset support, virtually any add-on chipset, or CPU with on-board chipset can be supported by Embedded BIOS 2000. Traditionally, chipsets provide DRAM memory management, bus control, and address space management. The Embedded BIOS 2000 architecture provides for Chipset Personality Modules that can be selected for a project. Embedded BIOS 2000 supports the following chipset and System-On-Chip solutions:

- GX 533@1.1W, GX 500@1.0W, GX 466@0.9W with CS5535
- SC1100
- SC1200/1201, SC2200, SC3200
- GX1 with CS5530A

Additional AMD chipset-level products supported by Embedded BIOS 2000:

- SC300/310
- SC400/410
- SC520

Board-Level Support
Embedded BIOS 2000's board-level support provides for the OEM to control the BIOS's access to chipset and CPU modules in major or subtle ways. Essentially a routing module, the board module contains routines that call associated routines in the Chipset and CPU Personality Modules. The board module routines can be modified as needed to replace the calls to the underlying CPU and chipset modules with custom code, as needed for hardware designs that work differently than standard reference designs supported by General Software. Embedded BIOS 2000 supports the following Geode development boards:

- DBSC1200 development board: SC1200
- GX DB533-TC (CRT): GX533@1.1W/CS5535
- GX DB533-TT (TFT): GX533@1.1W/CS5535
- SP4GX10: GX1/CS5530A
- SP4SC40: SC1100
- SP4SC31: SC1200/SC1201, SC2200, or SC3200

Additional AMD board-level products supported by Embedded BIOS 2000:

- Elan Am386SC300
- Elan Am486SC400
- Elan SC520

Optimized Core
Embedded BIOS 2000 is implemented in hand-optimized 8086 assembly language, with special code paths for a variety of different x86 compatible processors. The code paths have been hand-tuned to minimize the interrupt latency commonly found in desktop BIOS implementations, and many of the “hot paths” of the BIOS have been straight-line optimized for the common case.

ROM Disk File System
ROM disk software is integrated directly with the system BIOS itself, eliminating the need to populate the ROM scan area with ROM BIOS extensions to simulate one or more floppy or hard disks in ROM. Instead, with the ROM disk configuration feature enabled, an image of a floppy or hard disk can be stored in ROM anywhere in the address space of the Geode and treated as a solid-state drive. If the ROM disk feature is enabled, the ROM disk can be selectively turned on or off in the Setup screen.

RAM Disk File System
RAM disk software is also integrated directly into the system BIOS to support PCMCIA SRAM cards and other RAM areas as floppy or hard disk emulators. SETUP even has a formatting screen for the RAM disk.

Reflash Utility
The BIOS reflash utility allows BIOS Flash chips to be updated in a live system from a DOS command prompt. When the utility is run directly on the target system, Flash media services from the replacement BIOS are used as a Flash driver. This allows a system shipped with a BIOS that does not support Flash media services to be updated or replaced in the field.

Resident Flash Disk
The system BIOS supports a Resident Flash Disk (RFD) that provides read/write access to sectored Flash devices as though they were a floppy or hard disk of up to 32MB in size. The inclusion of this software makes it simple and easy to support Flash in embedded and hand-held consumer electronics. Multiple RFDs can be supported in the same Geode solutions-based platform.

BIOS Debugger
The integrated BIOS debugger gives the engineer bringing up new hardware the capability of debugging the hardware with powerful tools like a disassembler, breakpoints, CMOS editing, A20-line gating commands, cache control commands, PCI bus management commands, and Super I/O controls. The debugger is very useful for debugging chipset modules, CPU class modules, and initialization of user ROM extensions and hardware. Like the Setup screen, the integrated BIOS debugger can run directly on a PC keyboard and video screen, or it can be redirected over an RS232 serial link.

Watchdog Timer
Embedded systems deployed into more inaccessible areas need watchdog timer support, so that they can automatically restart in the event that application or system software fails. Embedded BIOS 2000 provides watchdog timer control functions to allow operating systems and

application programs to use watchdog timer hardware found in chipsets and certain CPU classes.

Console Redirection
Keyboard and video output may be selectively redirected over RS232 serial links for different system components. For example, standard console I/O, such as that used by DOS and DOS applications, can be redirected over any COM port, including those built-into high-integration CPUs. Debugger I/O and Setup screen I/O can also be redirected over the same or different RS232 serial links.

Manufacturing Mode
A special Manufacturing Mode feature provides the necessary provisions for programming electronics products through a high-speed serial link, and then testing and repairing the same items in the field at service centers. The OEM can write custom software that uses Embedded BIOS 2000 Manufacturing Mode functions to perform virtually any maintenance or programming task on the DBSC1200 development board under host control.

System Registration
The BIOS incorporates a System Registration facility that provides a method for users to obtain technical support from General Software. As the supplier of the BIOS for this platform, and as a supplier of Embedded BIOS 2000 Adaptation Kit products that ODMs and OEMs can use to build perfect-fit BIOSes for their own designs, General Software is ideally positioned to provide the technical support that embedded ODM/OEM user's need.

Legacy USB
Embedded BIOS 2000 provides support for keyboard and mouse device emulation for certain target hardware. This is accomplished through emulation of the 8042 keyboard controller using a Firmware Application that runs inside the SMM operating environment provided by Firmbase, together with a USB stack also implemented with Firmware Applications.

USB Mass Storage
Embedded BIOS 2000 provides the ability to boot from a large variety of USB mass storage devices using hard, floppy, or no-emulation (El-Torito) drives. This is accomplished through Firmware Applications that run inside the SMM operating environment provided by Firmbase. These Firmware Applications include a USB stack, a mass storage driver, and a driver to process INT13 requests from the BIOS. Currently supported USB Mass Storage devices include floppy and hard disk drives, CD/DVD-ROM drives, CompactFlash readers, DiskOnKey memory sticks, Zip drives, and more. Embedded BIOS 2000 USB boot is sold separately. Please contact your General Software account manager for additional details.

System Management Mode
Embedded BIOS 2000 harnesses the hardware support for System Management Mode (SMM) implemented by nearly all chipsets and embedded x86 CPUs, and makes it available through programming services in the Firmbase operating environment. Licensed separately, the Firmbase SDK enables OEMs to develop 32-bit firmware applications for SMM that run alongside all industry-standard operating systems, whether they are just booting, are running, have crashed, or are missing. The Firmbase high-availability operating environment can be used to integrate crash protection and recovery, remote administration, custom thermal management, and many other solutions that run from software stored directly in the BIOS ROM.

1.3 PC BIOS Features

POST
Embedded BIOS 2000 provides a comprehensive Power-On Self-Test (POST) algorithm that is automatically configured for the peripherals and capabilities selected by the adaptation engineer. During POST, hardware is initialized and tested, including the CPU, RAM, and peripherals. POST provides “beep code” diagnostics for errors when a display is not available, as well as error message diagnostics on the display when available. POST can also be configured to output status report codes to a manufacturing port (typically, port 80h) so that automated Q/A equipment can determine the status of a system during POST. A special set of ASCII POST status codes are also available through a serial port, for flexibility in the debugging process when new hardware is being brought up. Either POST code system, or both, can be used during debugging.

Setup Screen System
The Embedded BIOS 2000 setup screen system is configurable at the source level by the adaptation engineer to contain any combination of subscreens, including Basic CMOS Configuration, Feature Configuration, Plug-n-Play Configuration, Custom Configuration, Shadow Configuration, Diagnostics screens, Manufacturing Mode, Debugger access, and formatting of drive emulators such as RAM and RFD drives. SETUP screens can also be customized at the source level (in the Board Personality Module) to contain custom fields as required by the application.

HTML Browser
The BIOS Browser is also available for your use. Selectable as a boot action or from within Setup, it can present text only HTML files to your end users to document the platform, provide contact information for your company, etc. This provides a convenient means of providing information to help users recover from hardware problems, such as a failed hard drive.

Password Protection
Also available is a password protection system, so that a password must be provided by the end-user before POST allows booting of an operating system. The password is stored in CMOS, is one-way encrypted, and can be modified in a Setup screen.

Shadowing
The ability to shadow slower ROM devices with DRAM or SRAM is selectable in the Shadow Setup screen and calls chipset-specific code to enable shadowing for the BIOS ROM itself or for feature ROMs on a 16Kb region basis. DRAM may take the form of FPM, EDO, SDRAM, RDRAM, or other technologies.

Cache Control
Embedded BIOS 2000 provides extensive support for both internal CPU cache control (i486 and above) and external cache control (typically chipset-controlled). Internal cache is managed by the CPU class personality modules, whereas external cache is managed by the cache manager, which directs peripherals (chipset, 8042, custom I/O ports, or CPU integrated peripherals) to manage the cache. Keyboard controls on the PC/AT keyboard are implemented for enabling and disabling the cache on-the-fly (while the system is running). The BIOS provides cache control services to applications that allow operating systems and user code to control and inspect the status of the cache.

Processor Speed Control
CPU speed controls are handled by the system BIOS by routing control through the appropriate

logic (chipset, 8042, custom I/O ports, or CPU integrated peripherals). As with cache control, CPU speed is controllable on-the-fly at the keyboard or via programming interfaces.

1.4 Software Compatibility

Embedded BIOS 2000 offers a high degree of compatibility with past and current BIOS standards, allowing it to run off-the-shelf operating system software and application software.

Embedded BIOS 2000 has been systematically tested with the following industry standard operating systems:

- Windows XP (Professional and Home editions)
- Windows 2000 (Professional, Server, and Advanced Server editions)
- Windows 98 (Second edition)
- Windows Millennium
- Windows CE (3.0, 4.1, .NET, and 5.0 editions)
- Windows NT4 (Workstation and Server editions)

- Linux RedHat 9 (kernel 2.4.20-8)
- Linux RedHat 8 (kernel 2.4.18-24)
- Linux RedHat 7.3 (kernel 2.4.18-3)
- Linux RedHat 7.2 (kernel 2.4.7-10)
- Linux RedHat 7.1 (kernel 2.4.2-2)
- Linux Mandrake 9.1 (kernel 2.4.21-0.13mdk)
- Linux Mandrake 9 (kernel 2.4.19-16mdk)
- Linux Mandrake 8.1 (kernel 2.4.8-26mdk)

- VxWorks
- QNX
- MS-DOS 6.22

Embedded BIOS 2000 is rigorously tested with the following programs to ensure its compatibility with established industry application standards:

- AMIDIAG v3.20
- PCI_SIG Test Suite
- PNP Test Suite
- ACPI Test Suite
- SiSoftware Sandra
- Ultra-X Diagnostic Suite
- Bytemark Benchmarks

In addition to its standard data structures and programming interfaces, Embedded BIOS 2000 provides support for industry-standard initiatives, including:

- Advanced Power Management (APM) Specification 1.2
- Advanced Control and Power Interface (ACPI) Specification 2.0
- POST Memory Manager (PMM) Specification 1.01
- Plug-n-Play Specification 1.0A
- DMI 2.3.1 (SMBIOS)
- CD-ROM Boot “El Torito” Specification 1.0
- PCI Specification 2.2

- PXE Specification (through support of Argon’s software modules)
- BIOS32 Specification
- Enhanced Disk Drive Specification 1.1
- Firmbase Specification 1.3

This page intentionally left blank.

CHAPTER 2

POWER ON SELF TEST

2.1 About POST

When your development platform is powered on, Embedded BIOS 2000 tests and initializes the hardware and programs the chipset and other peripheral components. During this time, Power On Self Test (POST) progress codes are written by the system BIOS to I/O port 80h, allowing the user to monitor the progress with a special monitor. Appendix A lists the POST codes and their meanings.

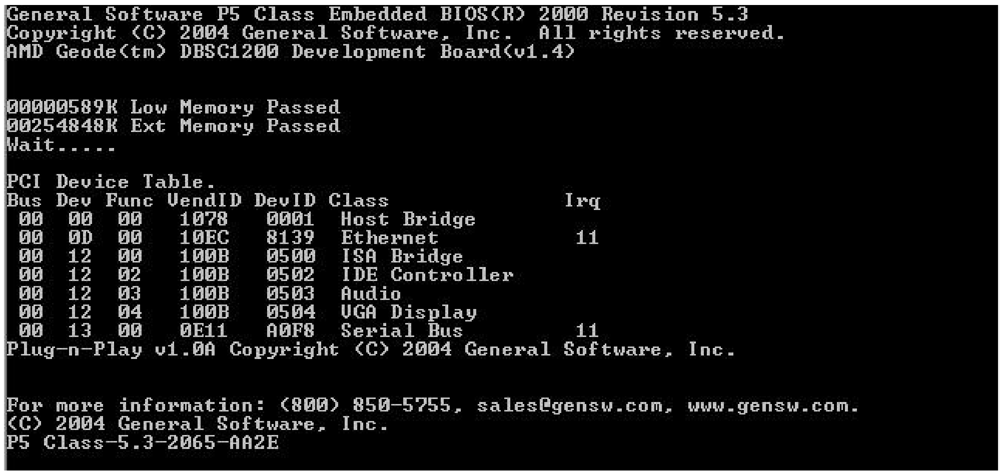
During early POST, no video is available to display error messages should a critical error be encountered; therefore, POST uses beeps on the speaker to indicate the failure of a critical system component during this time. Consult Appendix B for a list of Beep codes used by the BIOS of the DBSC1200 development board.

2.2 The BIOS User Interface

The DBSC1200 development board BIOS can use the standard keyboard and video device, or use console redirection to demonstrate headless operation. For headless operation, remove the standard keyboard and screen devices and the system will boot unattended. If an RS232 cable is attached to COM1, a PC/AT-style character-based POST is available from HyperTerminal, PROCOMM, or any other terminal emulator software that supports VT100 emulation. To use console redirection over an RS232 cable attached to COM1, press Ctrl-C (^C) during POST.

When a keyboard and video device are attached, the DBSC1200 development board can display either a traditional character-based PC BIOS display with memory count-up, or it can display a graphical POST with splash screen and progress icons. Both POST displays accept a Ctrl-C (^C) or <DEL> key press to enter the setup screen, and both display boot-time progress activity. The graphical display shows the status of file system devices and even OEM-defined devices (when the OEM adapts the BIOS to a particular OEM platform), but omits the character-based PCI resource display. The text-based POST displays the memory count-up and the PCI resource assignment table.

Figure 2.1 shows the format of the text-based POST display. The display is very similar if console redirection through a COM port is used instead.



```
General Software P5 Class Embedded BIOS(R) 2000 Revision 5.3
Copyright (C) 2004 General Software, Inc.  All rights reserved.
AMD Geode(tm) DBSC1200 Development Board(v1.4)


00000589K Low Memory Passed
00254848K Ext Memory Passed
Wait.....

PCI Device Table.
Bus Dev Func VendID DevID Class              Irq
 00  00  00   1078   0001  Host Bridge
 00  0D  00   10EC   8139  Ethernet           11
 00  12  00   100B   0500  ISA Bridge
 00  12  02   100B   0502  IDE Controller
 00  12  03   100B   0503  Audio
 00  12  04   100B   0504  VGA Display
 00  13  00   0E11   A0F8  Serial Bus         11
Plug-n-Play v1.0A Copyright (C) 2004 General Software, Inc.


For more information: (800) 850-5755, sales@gensw.com, www.gensw.com.
(C) 2004 General Software, Inc.
P5 Class-5.3-2065-AA2E
```

Figure 2.1. The text-based system, BIOS POST, provides a familiar desktop PC look-and-feel.

Figure 2.2 shows the graphical version of POST. The BIOS decompresses the main image and can display multiple overlaid graphics at various points in POST. The OEM can define the entire sequence and control the timing of the system for an embedded application, and can arrange to have different graphics displayed on each successive boot of the system. This feature is ideal for embedded systems that must show evidence of operation during startup, while the application loads underneath the splash screen. Once the application begins writing to the screen, the splash screen relinquishes control, providing a seamless graphical progression for the end user.

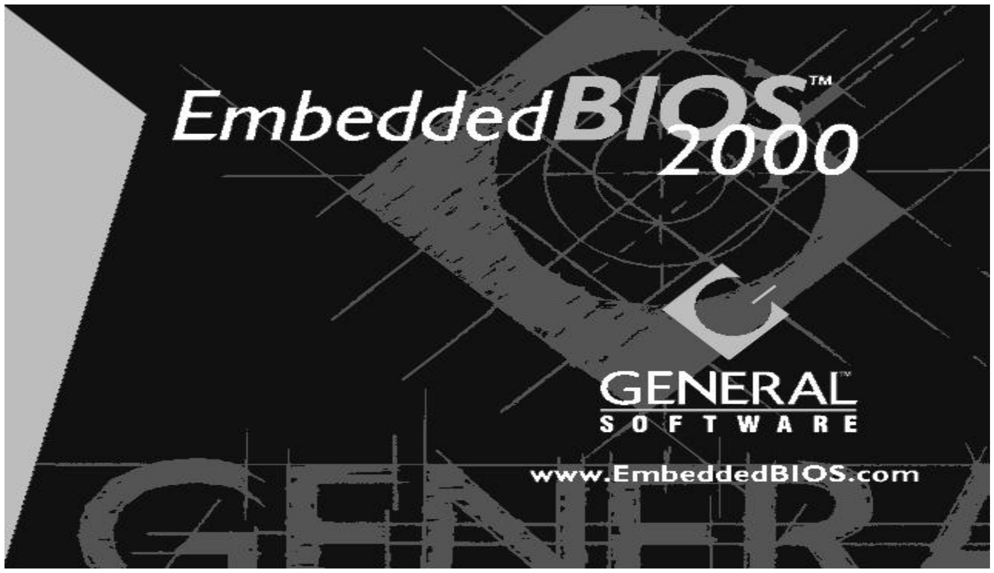


Figure 2.2. The graphical POST provides an instant-on and controlled display for the user to view.

When the DBSC1200 development board is powered on for the first time, you'll need to configure the system through the Setup Screen System (described in Chapter 3) before peripherals, such as disk drives, are recognized by the BIOS. The information is written to battery-backed CMOS RAM on the board's Real Time Clock. Should the board's battery fail, this information will be lost, and the board will need to be reconfigured.

The Feature Setup Screen of the DBSC1200 development board provides an option to disable the graphical/audio POST and switch to the legacy text-based version. This feature may not permanently disable the graphical POST if the BIOS adaptation calls for reverting to the graphical form after so many boots. If you find that the graphical POST comes back after several boots, it is because this option is enabled for this platform. Naturally, the OEM can use the Embedded BIOS 2000 Adaptation Kit to control whether Setup can be used to dictate the policy, and whether it is permanent or temporary.

More information about the BIOS POST Interface, please contact your General Software Account Manager to obtain a copy of the *Embedded BIOS 2000 Technical Reference Manual*.

This page intentionally left blank.

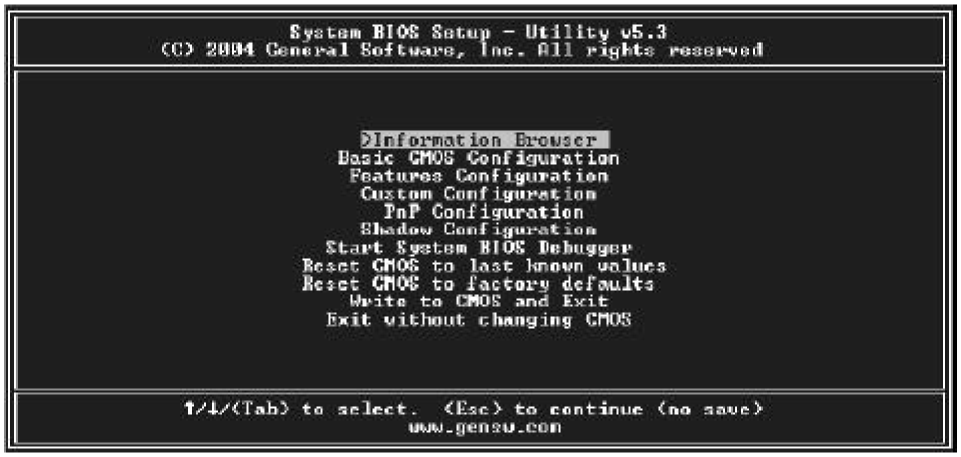
CHAPTER 3

SETUP SCREEN SYSTEM

3.1 The Setup Screen System

NOTE: The screens displayed in this chapter will differ from platform to platform, and may not be present at all, depending on the OEM's build configuration.

The DBSC1200 development board is configured from within the Setup Screen System – a series of menus invoked during POST by pressing Ctrl-C (^C) or the <DEL> key if the main keyboard is being used, or by pressing Ctrl-C (^C) if the console is being redirected to a terminal emulator program.



```
                 System BIOS Setup - Utility v5.3
   (C) 2004 General Software, Inc. All rights reserved

                    >Information Browser
                   Basic CMOS Configuration
                    Features Configuration
                     Custom Configuration
                      PnP Configuration
                     Shadow Configuration
                   Start System BIOS Debugger
                 Reset CMOS to last known values
                 Reset CMOS to factory defaults
                      Write to CMOS and Exit
                   Exit without changing CMOS

          ↑/↓/<Tab> to select.  <Esc> to continue (no save)
                          www.gensw.com
```

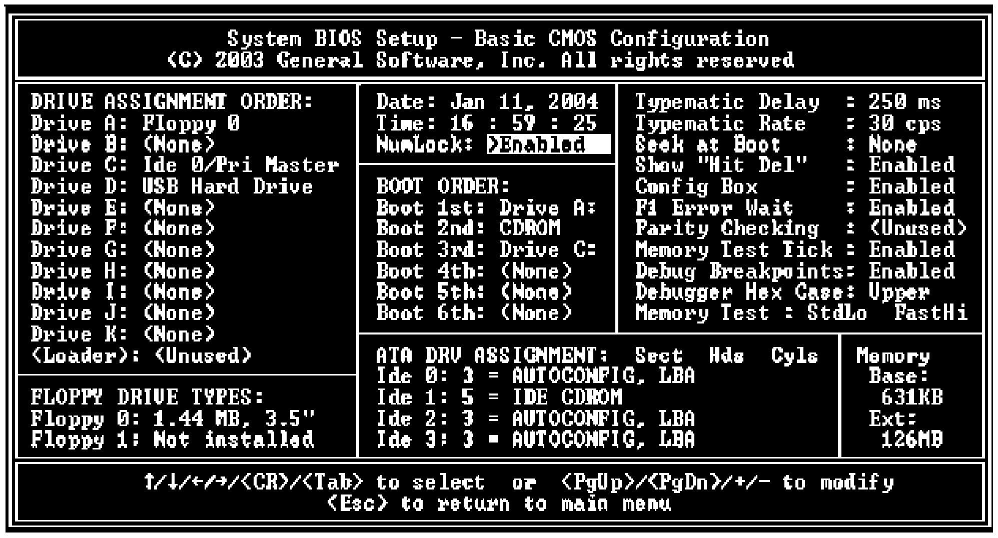
Figure 3.1. The Embedded BIOS 2000 Setup Screen Menu allows access to configuration and embedded features.

Once in the Setup Screen System (Figure 3.1), the user can navigate with the UP and DOWN arrow keys or the Ctrl-E (^E) and Ctrl-X (^X) keys from either the main console or from the remote terminal program. TAB and ENTER are used to advance to the next field, and '+' and '-' keys cycle through values, such as those in the Basic Setup Screen or the Diagnostics Setup Screen.

For a complete discussion of the Setup Screen System, please contact your General Software Account Manager to obtain a copy of the *Embedded BIOS 2000 Technical Reference Manual*.

3.2 Basic CMOS Configuration Screen

The drive types, boot activities, and POST optimizations of the DBSC1200 development board are configured from the Basic Setup Screen (Figure 3.2). In order to use disk drives with your system, you must select appropriate assignments of drive types in the left-hand column. Then, if you are using true floppy and IDE drives (not memory disks that emulate these drives), you need to configure the drive types themselves in the Floppy Drive Types and ATA Drive Assignment sections. Finally, you'll need to configure the boot sequence in the middle of the screen. Once these selections have been made, your system is ready to use.



```
                 System BIOS Setup - Basic CMOS Configuration
              (C) 2003 General Software, Inc. All rights reserved

DRIVE ASSIGNMENT ORDER:     Date: Jan 11, 2004    Typematic Delay   : 250 ms
Drive A: Floppy 0           Time: 16 : 59 : 25    Typematic Rate    : 30 cps
Drive B: (None)             NumLock: Enabled      Seek at Boot      : None
Drive C: Ide 0/Pri Master                         Show "Hit Del"    : Enabled
Drive D: USB Hard Drive     BOOT ORDER:           Config Box        : Enabled
Drive E: (None)             Boot 1st: Drive A:    F1 Error Wait     : Enabled
Drive F: (None)             Boot 2nd: CDROM       Parity Checking   : (Unused)
Drive G: (None)             Boot 3rd: Drive C:    Memory Test Tick  : Enabled
Drive H: (None)             Boot 4th: (None)      Debug Breakpoints: Enabled
Drive I: (None)             Boot 5th: (None)      Debugger Hex Case: Upper
Drive J: (None)             Boot 6th: (None)      Memory Test : StdLo  FastHi
Drive K: (None)
(Loader): (Unused)          ATA DRV ASSIGNMENT:  Sect  Hds  Cyls   Memory
                            Ide 0: 3 = AUTOCONFIG, LBA               Base:
FLOPPY DRIVE TYPES:         Ide 1: 5 = IDE CDROM                      631KB
Floppy 0: 1.44 MB, 3.5"     Ide 2: 3 = AUTOCONFIG, LBA               Ext:
Floppy 1: Not installed     Ide 3: 3 = AUTOCONFIG, LBA                126MB

         ↑/↓/←/→/(CR)/(Tab) to select  or  (PgUp)/(PgDn)/+/- to modify
                         (Esc) to return to main menu
```

Figure 3.2. The Embedded BIOS 2000 Basic Setup Screen is used to configure drives, boot actions, and POST.

3.2.1 Configuring Drive Assignments

Embedded BIOS 2000 allows the user to map a different file system to each drive letter. The BIOS allows file systems for each floppy (Floppy0 and Floppy1), each IDE drive (Ide0, Ide1, Ide2, and Ide3), and memory disks when configured (Flash0, ROM0, RAM0, etc.). Figure 3.2 shows how the first floppy drive (Floppy0) is assigned to drive A: in the system, and then shows how the first IDE drive (Ide0) is assigned to drive C: in the system.

To switch two floppy disks around or two hard disks around, just map Floppy0 to B: and Floppy1 to A: and for hard disks map Ide0 to D: and Ide1 to C:.

CAUTION: Take care to not skip drive A: when making floppy disk assignments, as well as drive C: when making hard disk assignments. The first floppy should be A:, and the first hard drive should be C:. Also, do not assign the same file system to more than one drive letter. Thus, Floppy0 should not be used for both A: and B:. The BIOS permits this to allow embedded devices to alias drives, but desktop operating systems may not be able to maintain cache coherency with such a mapping in place.

A special field in this section entitled **[Boot Method: (Windows CE/Boot Sector)]** is used to configure the CE Ready feature of the BIOS. For normal booting (DOS, Windows NT, etc.), select **[Boot Sector]** or **[Unused]**.

3.2.2 Configuring Floppy Drive Types

If true floppy drive file systems (and not their emulators, such as ROM, RAM, or Flash disks) are mapped to drive letters, then the floppy drives themselves must be configured in this section. Floppy0 refers to the first floppy disk drive on the drive ribbon cable (normally drive A:), and Floppy1 refers to the second drive (drive B:).

3.2.3 Configuring IDE Drive Types

If true IDE disk file systems (and not their emulators, such as ROM, RAM, or Flash disks) are mapped to drive letters, then the IDE drives themselves must be configured in this section. The following table shows the drive assignments for Ide0-Ide3:

FILE SYSTEM NAME	CONTROLLER	MASTER/SLAVE
Ide0	Primary (1f0h)	Master
Ide1	Primary (1f0h)	Slave
Ide2	Secondary (170h)	Master
Ide3	Secondary (170h)	Slave

To use the primary master IDE drive in your system (the typical case), just configure Ide0 in this section, and map Ide0 to drive C: in the Configuring Drive Assignments section.

The ATA Drive Assignment section lets you select the type for each of the four IDE drives: **None**, **User**, **Physical, LBA**, **CHS**, or **CDROM**.

The **User** type allows the user to select the maximum cylinders, heads, and sectors per track associated with the IDE drive. This method is now rarely used since LBA is now in common use.

The **Physical** type instructs the BIOS to query the drive's geometry from the controller on each POST. No translation on the drive's geometry is performed, so this type is limited to drives of 512MB or less. Commonly, this is used with embedded ATA PC Cards.

The **LBA** type instructs the BIOS to query the drive's geometry from the controller on each POST, but then translate the geometry according to the industry-standard LBA convention. This supports up to 137GB drives. **Use this method for all new drives.**

The **CHS** type instructs the BIOS to query the drive's geometry from the controller on each POST, but then translate the geometry according to the Phoenix CHS convention. Using this type on a drive previously formatted with LBA or Physical geometry might show data as being missing or corrupted.

The **CDROM** type instructs the BIOS to query the CDROM drive to search for a bootable image as specified in the "El Torito" boot specification on each POST. The "El Torito" boot specification lists bootable images as being floppy (soft), hard, and linear images. **Use this type for all IDE CDROM drives.**

3.2.4 Configuring Boot Order

Embedded BIOS 2000 supports user-defined steps in the boot sequence. When the entire system has been initialized, POST executes these steps in order until an operating system successfully loads. In addition, other pre-boot features can be run before, after, or between operating system load attempts. The following actions are supported:

Embedded BIOS 2000 supports up to six different user-defined steps in the boot sequence. When the entire system has been initialized, POST executes these steps in order until an operating system successfully loads. In addition, other pre-boot features can be run before, after, or between operating system load attempts. The following actions are supported:

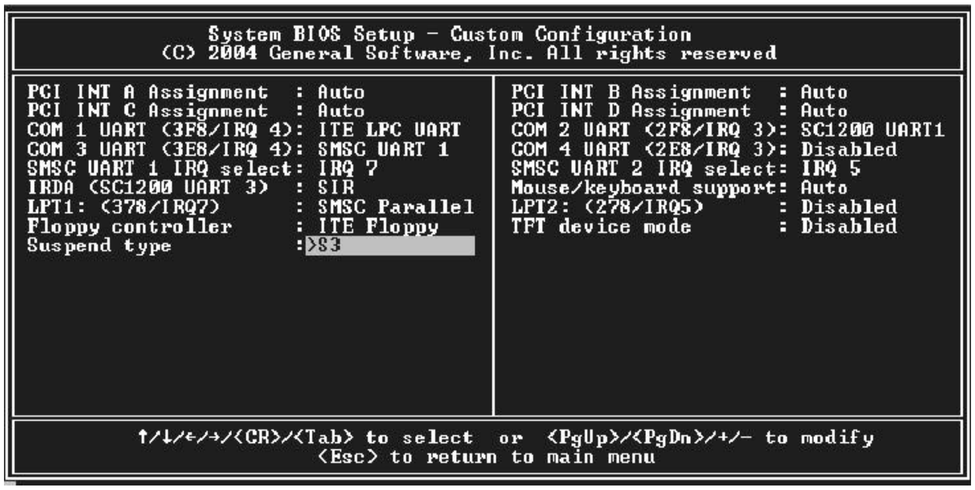
Drive A: – D:	Boot operating system from specified drive. If "Loader" is set to "BootRecord" or "Unused", then the standard boot record will be invoked, causing DOS, Windows95, Windows 98, Windows ME, Windows 2000, Windows NT, Windows XP, Linux, or other industry-standard operating systems to load. If "Boot Method" is set to "Windows CE", then the boot drive's boot record will not be used, and instead the BIOS will attempt to load and execute the Windows CE Kernel file, `NK.BIN`, from the root directory of each boot device.
CDROM	Boot from the first IDE or USB CDROM found that contains an El Torito bootable CDROM.
Debugger	Launch the Integrated BIOS Debugger. To return exit the debugger environment, type "G" at the debugger prompt and press ENTER.
MFGMODE	Initiate Manufacturing Mode, allowing the system to be configured remotely via an RS232 connection to a host computer.
WindowsCE	Execute a ROM-resident copy of Windows CE, if available. This feature is not applicable unless configured by the OEM in the BIOS adaptation.
DOS in ROM	Execute a ROM-resident copy of DOS, if available. This feature is not applicable unless an XIP copy of DOS has been stored in the BIOS boot ROM.
Alarm	Generate an alarm by beeping the speaker and sending a signal to a Firmware Application running in the Firmbase environment. The application can perform whatever processing is necessary to handle the alarm, including taking local action, interacting with other tightly-coupled computers, or even notifying other systems on the network, for

	example. These applications are beyond the scope of the Embedded BIOS 2000 Adaptation Kit.
Maintenance	Enter maintenance mode by sending a signal to a Firmware Application running in the Firmbase environment. The application can perform whatever processing is necessary to implement the maintenance mode, which is largely defined to mean some state where the system is capable of being diagnosed and/or repaired in the field. These applications are beyond the scope of the Embedded BIOS 2000 Adaptation Kit.
RAS	Enter remote access mode or perform network boot. This is done by sending a signal to a Firmware Application running in the Firmbase environment. The application can perform whatever processing is necessary to implement the RAS mode, which is largely defined to mean some state where the system accepts remote connections for normal operation; not specifically for field maintenance. These applications are beyond the scope of the Embedded BIOS 2000 Adaptation Kit. Commonly, RAS is used by the BOOTNET package to perform a remote network boot.
Power Off	Cause the target to switch off its power with a “soft off” feature, and signal Firmware Applications running in the Firmbase environment that power is going down. These applications are beyond the scope of the Embedded BIOS 2000 Adaptation Kit.
Reboot	Reboot the target, and send a signal to a Firmware Application running in the Firmbase environment indicating that the target is rebooting. These applications are beyond the scope of the Embedded BIOS 2000 Adaptation Kit.
CLI	Enter command line mode by calling a special Board Module Function (BoardPostControl) that can be used to implement an OEM-defined Command Language Interpreter. The design of such an interpreter is beyond the scope of the Embedded BIOS 2000 Adaptation Kit.
None	No action; POST proceeds to the next activity in the sequence.

For a complete discussion of the Disk File Management System, please contact your General Software Account Manager to obtain a copy of the *Embedded BIOS 2000 Technical Reference Manual*.

3.3 Custom Configuration Setup Screen

The hardware-specific features of the DBSC1200 development board are configured with the Custom Setup Screen (Figure 3.3). All features are straightforward except for the UART resource assignment options, which control the SMSC SuperI/O resources on the baseboard, the ITE SuperI/O resources on the CPU module, and the integrated SC1200 resources. The UARTs may be configured to combine a maximum of two LPC UARTs and one parallel port device at the same time. Please note that the SC1200 devices are independent of this configuration.

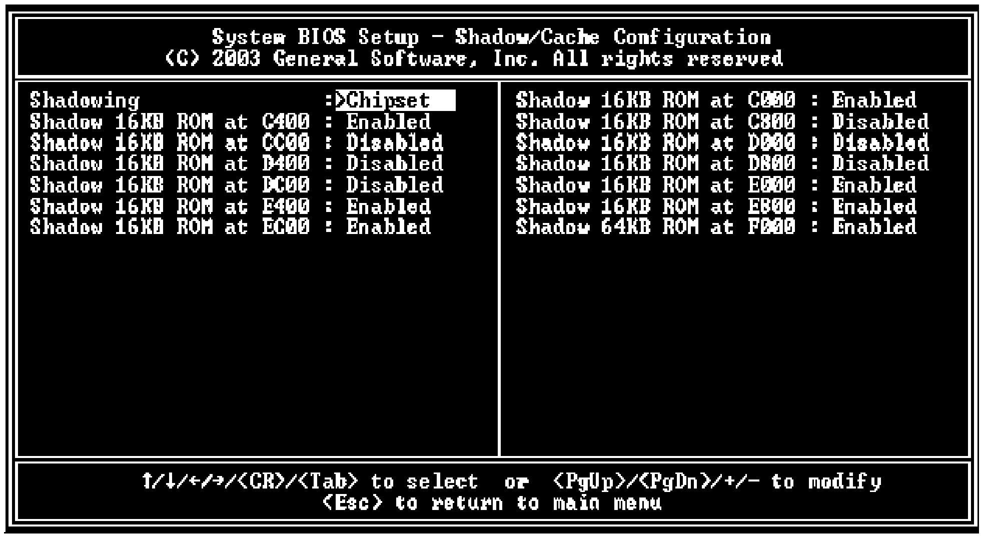


```
                    System BIOS Setup - Custom Configuration
            (C) 2004 General Software, Inc. All rights reserved

PCI INT A Assignment   : Auto            PCI INT B Assignment   : Auto
PCI INT C Assignment   : Auto            PCI INT D Assignment   : Auto
COM 1 UART (3F8/IRQ 4): ITE LPC UART     COM 2 UART (2F8/IRQ 3): SC1200 UART1
COM 3 UART (3E8/IRQ 4): SMSC UART 1      COM 4 UART (2E8/IRQ 3): Disabled
SMSC UART 1 IRQ select: IRQ 7            SMSC UART 2 IRQ select: IRQ 5
IRDA (SC1200 UART 3)   : SIR             Mouse/keyboard support: Auto
LPT1: (378/IRQ7)       : SMSC Parallel   LPT2: (278/IRQ5)       : Disabled
Floppy controller      : ITE Floppy      TFT device mode        : Disabled
Suspend type           :>S3

        ↑/↓/←/→/<CR>/<Tab> to select  or  <PgUp>/<PgDn>/+/- to modify
                          <Esc> to return to main menu
```

Figure 3.3. The Embedded BIOS 2000 Custom Setup Screen is used to configure low-level hardware.

3.4 Shadow Configuration Setup Screen

The Shadow Configuration Setup Screen (Figure 3.4) of the DBSC1200 development board allows the selective enabling and disabling of shadowing in 16Kb sections, except for the top 64Kb of the BIOS ROM, which is shadowed as a unit. Normally, shadowing should be enabled at C000/C400 (to enhance VGA ROM BIOS performance) and then E000-F000 should be shadowed to maximize system ROM BIOS performance.

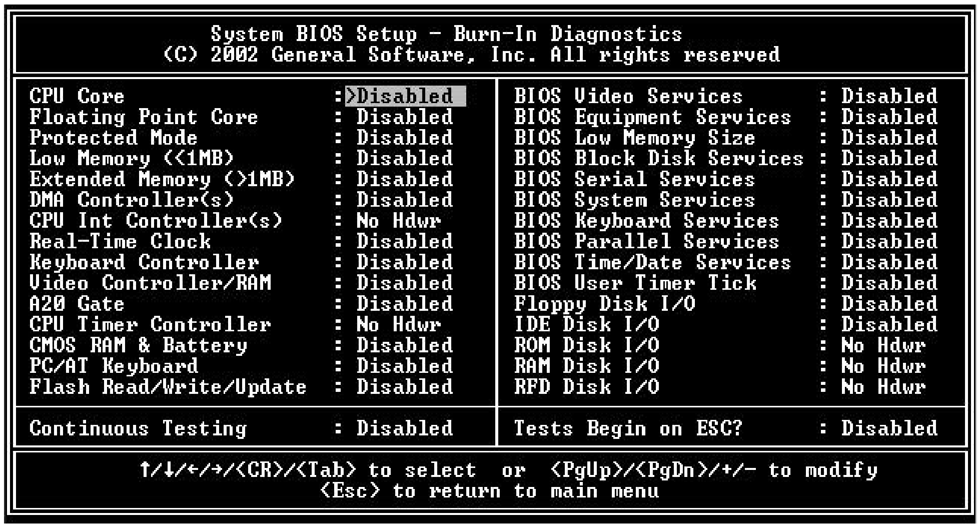


```
                  System BIOS Setup - Shadow/Cache Configuration
            (C) 2003 General Software, Inc. All rights reserved

Shadowing                  :>Chipset     Shadow 16KB ROM at C000 : Enabled
Shadow 16KB ROM at C400 : Enabled        Shadow 16KB ROM at C800 : Disabled
Shadow 16KB ROM at CC00 : Disabled       Shadow 16KB ROM at D000 : Disabled
Shadow 16KB ROM at D400 : Disabled       Shadow 16KB ROM at D800 : Disabled
Shadow 16KB ROM at DC00 : Disabled       Shadow 16KB ROM at E000 : Enabled
Shadow 16KB ROM at E400 : Enabled        Shadow 16KB ROM at E800 : Enabled
Shadow 16KB ROM at EC00 : Enabled        Shadow 64KB ROM at F000 : Enabled

        ↑/↓/←/→/<CR>/<Tab> to select  or  <PgUp>/<PgDn>/+/- to modify
                          <Esc> to return to main menu
```

Figure 3.4. The Embedded BIOS 2000 Shadow Setup Screen is used to configure ROM shadowing.

3.5 Standard Diagnostics Routines Setup Screen

Embedded systems may require automated burn-in testing in the development cycle. This facility can be provided directly in the system BIOS of the DBSC1200 development board through the Standard Diagnostics Routines Setup Screen (Figure 3.5). To use the system, selectively enable or disable features to be tested, and then enable the **[Tests Begin on ESC?]** option to cause the system test suite to be invoked. To repeat the system test battery continuously, you should also enable the **[Continuous Testing]** option. When continuous testing is started, the system will continue until an error is encountered.



```
               System BIOS Setup - Burn-In Diagnostics
          (C) 2002 General Software, Inc. All rights reserved

CPU Core                   :>Disabled   BIOS Video Services      : Disabled
Floating Point Core        : Disabled   BIOS Equipment Services  : Disabled
Protected Mode             : Disabled   BIOS Low Memory Size     : Disabled
Low Memory (<1MB)          : Disabled   BIOS Block Disk Services : Disabled
Extended Memory (>1MB)     : Disabled   BIOS Serial Services     : Disabled
DMA Controller(s)          : Disabled   BIOS System Services     : Disabled
CPU Int Controller(s)      : No Hdwr    BIOS Keyboard Services   : Disabled
Real-Time Clock            : Disabled   BIOS Parallel Services   : Disabled
Keyboard Controller        : Disabled   BIOS Time/Date Services  : Disabled
Video Controller/RAM       : Disabled   BIOS User Timer Tick     : Disabled
A20 Gate                   : Disabled   Floppy Disk I/O          : Disabled
CPU Timer Controller       : No Hdwr    IDE Disk I/O             : Disabled
CMOS RAM & Battery         : Disabled   ROM Disk I/O             : No Hdwr
PC/AT Keyboard             : Disabled   RAM Disk I/O             : No Hdwr
Flash Read/Write/Update    : Disabled   RFD Disk I/O             : No Hdwr

Continuous Testing         : Disabled   Tests Begin on ESC?      : Disabled

          ↑/↓/←/→/<CR>/<Tab> to select  or  <PgUp>/<PgDn>/+/- to modify
                          <Esc> to return to main menu
```

Figure 3.5. The Standard Diagnostic Setup Screen provides burn-in tests for manufacturing.

CAUTION: The disk I/O diagnostics perform write operations on those drives; therefore, only use spare drives as data could be harmed by the test.

ADVISORY: The keyboard test may fail when in fact the hardware is operating within reasonable limits. This is because although the device may produce occasional errors, the BIOS retries operations when failures occur during normal operation of the system.

3.6 BIOS Debugger Setup Screen

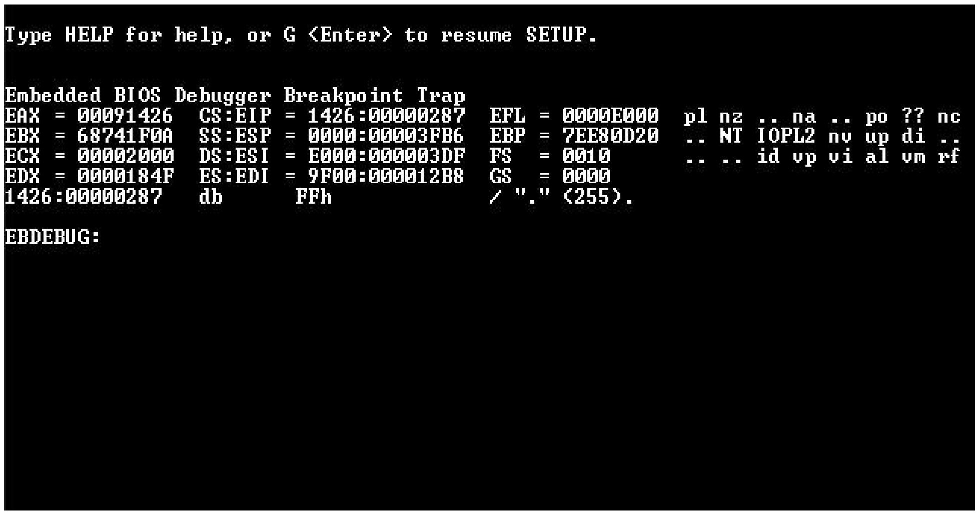
The Embedded BIOS 2000 Integrated Debugger may be invoked from the Setup Screen main menu, as well as a boot activity (Figure 3.6).

Once invoked, the debugger will display the debugger prompt…

```
EBDEBUG:
```

…and await debugger commands. To resume back to the Setup Screen main menu, type the following command, which instructs the debugger to “go”:

```
EBDEBUG: G (ENTER)
```



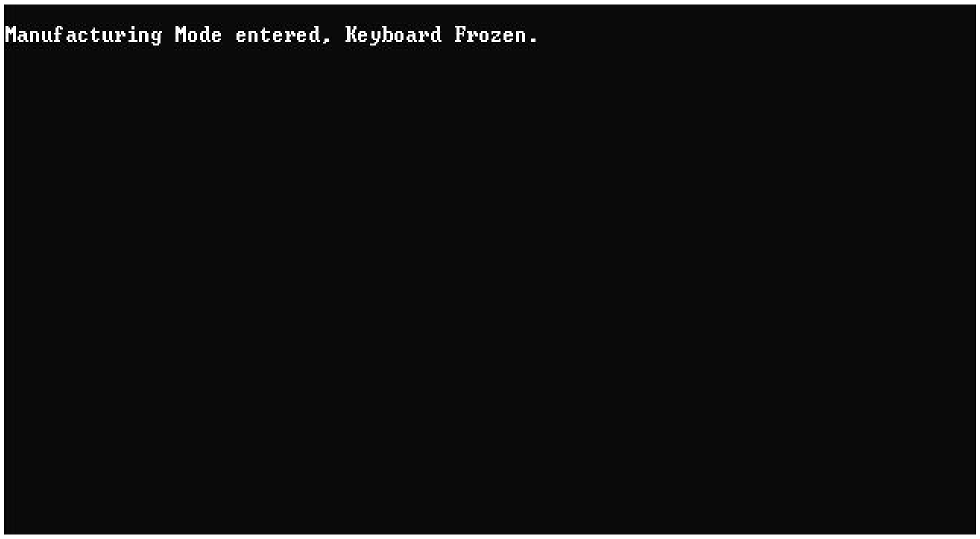
```
Type HELP for help, or G <Enter> to resume SETUP.

Embedded BIOS Debugger Breakpoint Trap
EAX = 00091426  CS:EIP = 1426:00000287  EFL = 0000E000  pl nz .. na .. po ?? nc
EBX = 68741F0A  SS:ESP = 0000:00003FB6  EBP = 7EE80D20  .. NT IOPL2 nv up di ..
ECX = 00002000  DS:ESI = E000:000003DF  FS  = 0010      .. .. id vp vi al vm rf
EDX = 0000184F  ES:EDI = 9F00:000012B8  GS  = 0000
1426:00000287   db      FFh             / "." (255).

EBDEBUG:
```

Figure 3.6. Embedded BIOS 2000 Integrated Debugger provides a powerful, integrated development tool.

You can find more information about the Integrated Debugger in Chapter 7.

3.7 Start RS232 Manufacturing Link Setup Screen

The Embedded BIOS 2000 Manufacturing Mode may be invoked from the Setup Screen main menu, as well as a boot activity. Once invoked, Manufacturing Mode takes over the system and freezes the system console (Figure 3.7). The host can resume operation of the system and give control back to the DBSC1200 development board Setup Screen system with special control software.



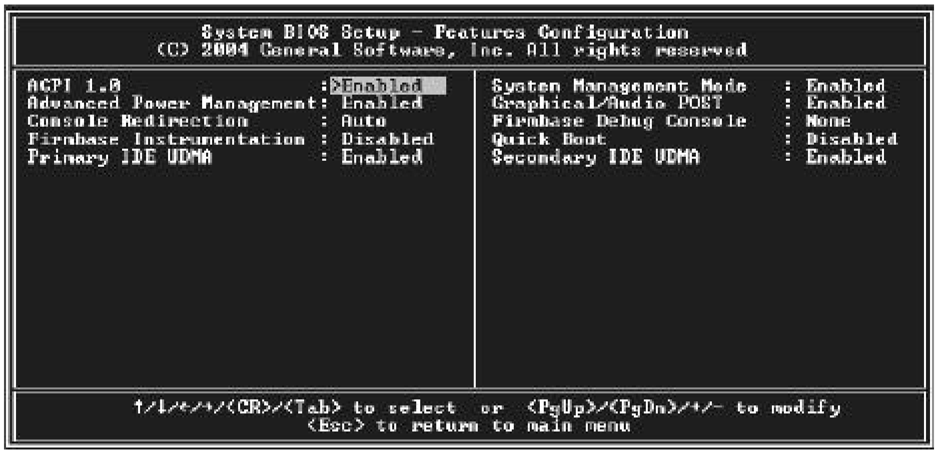
```
Manufacturing Mode entered, Keyboard Frozen.
```

Figure 3.7. The Start RS232 Manufacturing Link Setup Screen Provides Access to Manufacturing Mode.

You can find more information about Manufacturing Mode in Chapter 4; a complete discussion of Manufacturing Mode can be found in Chapter 8 of the *Embedded BIOS 2000 Technical Reference Manual*.

3.8 BIOS Features Screen

The Embedded BIOS 2000 Features Setup screen can be used to enable or disable certain features of this platform. Figure 3.8 shows an example Features Setup screen. It should be noted that this screen may vary from target to target, based on what features were enabled at the time the BIOS was built for this platform. Features that can be controlled with this screen include ACPI, Console Redirection, Splash Screen, Multiprocessing, PMM, Quick Boot, SMBIOS, SMM, and Firmbase.

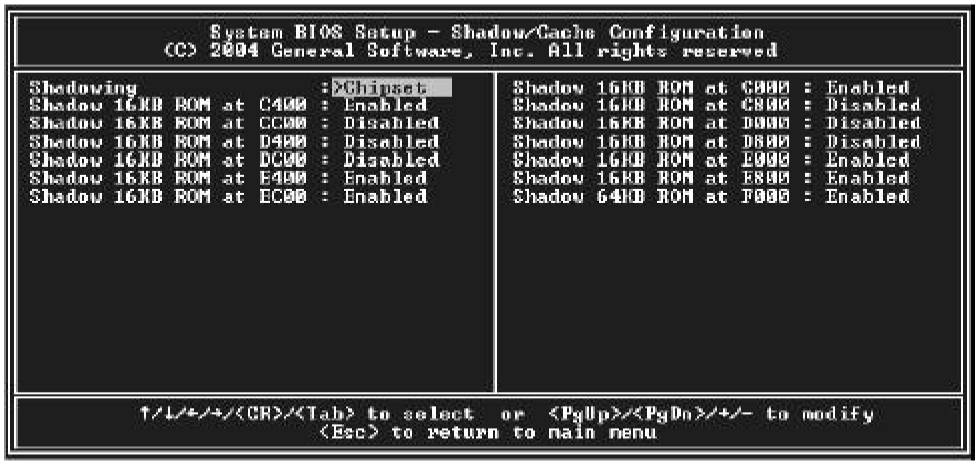


```
                 System BIOS Setup - Features Configuration
           (C) 2004 General Software, Inc. All rights reserved

ACPI 1.0                   :>Enabled     System Management Mode    : Enabled
Advanced Power Management: Enabled       Graphical/Audio POST      : Enabled
Console Redirection        : Auto        Firmbase Debug Console    : None
Firmbase Instrumentation : Disabled      Quick Boot                : Disabled
Primary IDE UDMA           : Enabled     Secondary IDE UDMA        : Enabled

        ↑/↓/←/→/<CR>/<Tab> to select  or  <PgUp>/<PgDn>/+/- to modify
                         <Esc> to return to main menu
```

Figure 3.8. The Embedded BIOS 2000 Features Setup Screen allows you to re-configure BIOS-level features.

3.9 BIOS Plug-n-Play Setup Screen

The DBSC1200 development board Plug-n-Play Setup Screen (Figure 3.9) allows the selective enabling and disabling of both Plug-n-Play and Plug-n-Play OS support within the BIOS. It also gives the user the ability to specify which IRQs and DMAs are to be used by the Plug-n-Play system.



```
              System BIOS Setup - Shadow/Cache Configuration
           (C) 2004 General Software, Inc. All rights reserved

Shadowing                  :>Chipset     Shadow 16KB ROM at C000 : Enabled
Shadow 16KB ROM at C400 : Enabled        Shadow 16KB ROM at C800 : Disabled
Shadow 16KB ROM at CC00 : Disabled       Shadow 16KB ROM at D000 : Disabled
Shadow 16KB ROM at D400 : Disabled       Shadow 16KB ROM at D800 : Disabled
Shadow 16KB ROM at DC00 : Disabled       Shadow 16KB ROM at E000 : Enabled
Shadow 16KB ROM at E400 : Enabled        Shadow 16KB ROM at E800 : Enabled
Shadow 16KB ROM at EC00 : Enabled        Shadow 64KB ROM at F000 : Enabled

        ↑/↓/←/→/<CR>/<Tab> to select  or  <PgUp>/<PgDn>/+/- to modify
                         <Esc> to return to main menu
```

Figure 3.9. The Embedded BIOS 2000 Plug-n-Play Setup Screen is used to configure PnP parameters.

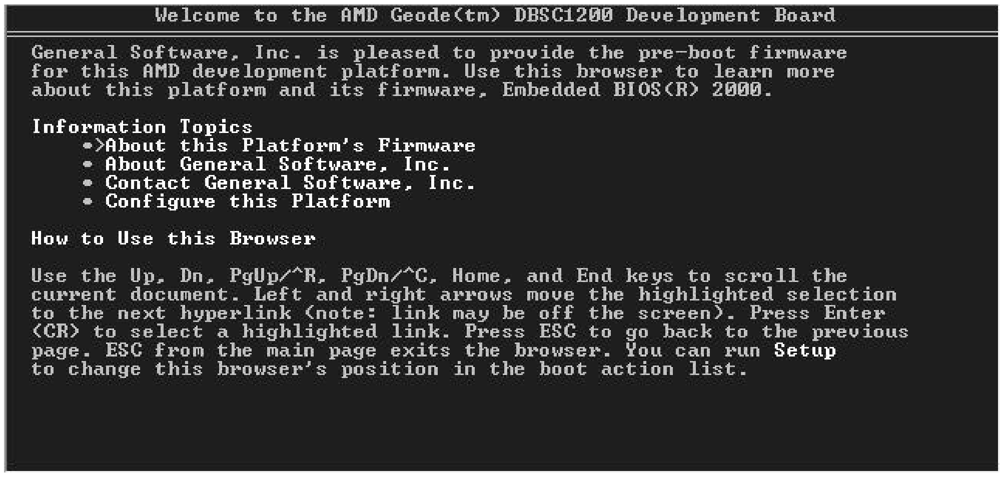
More information about PnP is available in Chapter 10 of the *Embedded BIOS 2000 Technical Reference Manual*.

3.10 BIOS HTML Browser Setup Screen

The DBSC1200 development board BIOS contains a built-in HTML browser that allows the user to browse (in text mode using the keyboard instead of the mouse) HTML documents stored in the BIOS ROM.

The HTML browser can be a boot action item, and by default is the primary boot action. Thus, when a platform boots for the first time, it should cause the browser to display its first page. The HTML pages for a platform are created using the Embedded BIOS 2000 Adaptation Kit. The ones for this platform were designed to provide the user with information about the platform, General Software, the firmware, and how to obtain technical support for the platform.

Customers can create their own HTML user interface for their embedded device, simply by writing HTML. The HTML browser works over the main console and over console redirection. More information about the BIOS Browser can be found in Chapter 8.

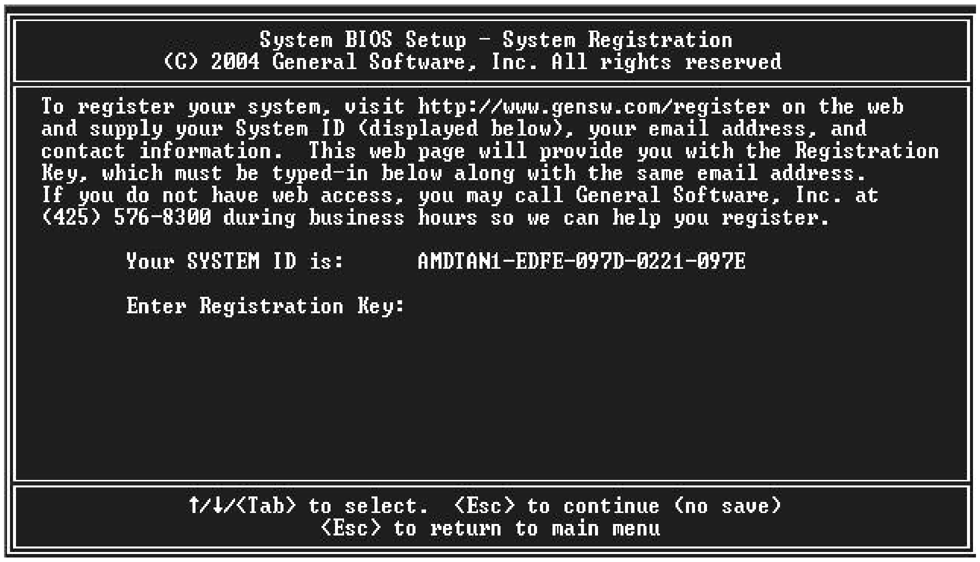


```
             Welcome to the AMD Geode(tm) DBSC1200 Development Board

  General Software, Inc. is pleased to provide the pre-boot firmware
  for this AMD development platform. Use this browser to learn more
  about this platform and its firmware, Embedded BIOS(R) 2000.

  Information Topics
      •>About this Platform's Firmware
      • About General Software, Inc.
      • Contact General Software, Inc.
      • Configure this Platform

  How to Use this Browser

  Use the Up, Dn, PgUp/^R, PgDn/^C, Home, and End keys to scroll the
  current document. Left and right arrows move the highlighted selection
  to the next hyperlink (note: link may be off the screen). Press Enter
  (CR) to select a highlighted link. Press ESC to go back to the previous
  page. ESC from the main page exits the browser. You can run Setup
  to change this browser's position in the boot action list.
```

Figure 3.10. The Integrated HTML Browser setup screen. Use it to create your own HTML user interface for your embedded device.

3.11 BIOS Registration Setup Screen

The DBSC1200 development board BIOS can incorporate a system registration facility that provides a method for users to obtain technical support from General Software directly. As the supplier of the BIOS for this platform, and as a supplier of Embedded BIOS 2000 Adaptation Kit products that ODMs and OEMs can use to build perfect-fit BIOSes for their own designs, General Software is ideally positioned to provide the technical support that embedded ODM/OEM user's need. Find more information about System Registration in Chapter 9.



```
                 System BIOS Setup - System Registration
         (C) 2004 General Software, Inc. All rights reserved

 To register your system, visit http://www.gensw.com/register on the web
 and supply your System ID (displayed below), your email address, and
 contact information.  This web page will provide you with the Registration
 Key, which must be typed-in below along with the same email address.
 If you do not have web access, you may call General Software, Inc. at
 (425) 576-8300 during business hours so we can help you register.

        Your SYSTEM ID is:      AMDTAN1-EDFE-097D-0221-097E

        Enter Registration Key:

              ↑/↓/<Tab> to select.  <Esc> to continue (no save)
                         <Esc> to return to main menu
```

Figure 3.11. The Registration Screen provides a registration key for access to technical support.

This page intentionally left blank.

CHAPTER 4

MANUFACTURING MODE

4.1 Using Manufacturing Mode

The DBSC1200 development board BIOS may provide a special mode, called Manufacturing Mode that allows the DBSC1200 development board to be controlled by a host computer such as a laptop or desktop PC. Running special software supplied by General Software, the host can access the DBSC1200 development board drives and manage the file systems on the DBSC1200 development board, reprogram Flash memories, and test the DBSC1200 development board hardware.

There are several methods by which the DBSC1200 development board can enter Manufacturing Mode. These methods are detailed elsewhere in this manual and others.

Once the DBSC1200 development board has entered Manufacturing Mode, the host PC may cause the target to perform functions by issuing commands in protocol over either an RS232 connection or a parallel port-based connection using ECP or EPP. There are two ways to access the target from the host PC — from the Manufacturing Mode HOST Program and from the Manufacturing Mode device driver.

4.1.1 Sample Manufacturing Mode HOST Program

The first way is to run a program that accesses the host-side Manufacturing Mode functions. An example of such a program is `HOST.EXE`, which can be obtained from General Software. This program runs under DOS and, using a full-screen windowing interface, illustrates the basic functionality of the Manufacturing Mode protocol. It should be noted that this program is a working sample program, and is not intended to be a production-quality control tool.

Run the HOST program on a “host” computer, so that its main menu is displayed (Figure 4.1). By default, HOST connects via COM1 (3F8h). The default baud rate is set to **[Auto]**, meaning it will use whatever baud rate the DBSC1200 development board is set to. You can change both of these by using command line switches. Type "HOST /?" for the available switches.

On the host, select **[Get Target Attention]**, within a couple seconds of selecting the manufacturing mode on the DBSC1200 development board. You should see the host program immediately display a yellow status box that shows that the connection has been established.

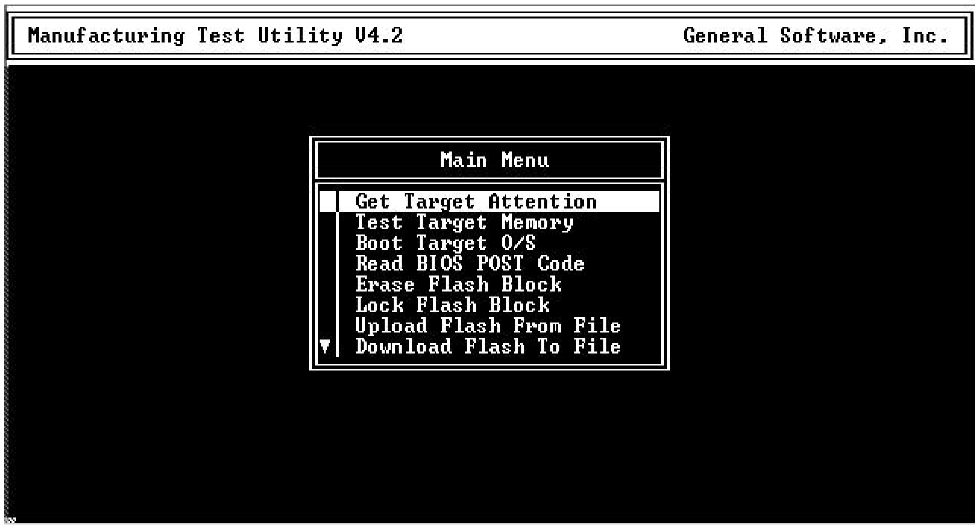


Figure 4.1. The Embedded BIOS 2000 Manufacturing Mode HOST Program.

If the connection hasn't been established, then try the connection again on the host side, or reboot the DBSC1200 development board and try again, this time having the host program get the DBSC1200 development board's attention within two seconds or so of the DBSC1200 development board entering of manufacturing mode. If this still fails, check that your null modem serial cable is connected securely to the proper ports. You may also want to lower the baud rate for the manufacturing mode on the DBSC1200 development board, since a higher baud rate may be more error prone.

Once you have established a connection, you can use HOST to test the link by continuously exercising it, or scanning the DBSC1200 development board's drives, or uploading files in Flash, and more.

4.1.2 Manufacturing Mode Drive Redirection

The second way to access the DBSC1200 development board through Manufacturing Mode is to install the `MFGDRV.SYS` device driver on the host system. This device driver loads under MS-DOS and Windows 98 DOS mode, and maps a new drive letter on the host to a drive on the DBSC1200 development board. Unfortunately, it is a DOS-only driver, and will not operate under Windows, not even in a Windows DOS box. Nor do we have a Linux version available at this time.

The INT 13h redirection support in the Manufacturing Mode protocol can be exposed by loading the `MFGDRV.SYS` device driver on the host by using the following `CONFIG.SYS` line:

```
DEVICE=MFGDRV.SYS /BAUD=rate /PORT=COMn /UNIT=u /AUTO
```

This device driver runs under any DOS-compatible operating system, and creates a drive letter on your host PC (usually D: if your last hard drive is C:) that can be used to interact with the specified INT 13h unit.

The *u* parameter specifies the BIOS unit number of the floppy disk, RAM disk, RFD drive, or ROM disk to be redirected, where 0 corresponds to drive A: and 1 for drive B. By default, this value is 80 (a hex number without a "0x" in front or 'h' appended to it), which corresponds with the unit for the first hard drive or emulator.

The `/BAUD=rate` parameter can be used to match the baud rate used by the target's BIOS. Legal values are 19K, 28K, 38K, 56K, and 115K. If this parameter is not specified, then the baud rate is autodetected.

The `/AUTO` parameter, if specified, tells `MFGDRV.SYS` to automatically format the remote drive if it determines that it is unformatted. By default, `MFGDRV.SYS` will not automatically format the remote drive, and will instead examine the media for a pre-existing format. If not found, then `MFGDRV.SYS` asks the host PC operator if the remote drive should be formatted.

Once the connection is established, you can read and write the DBSC1200 development board's drive as if it were simply another drive on your host system. The only difference is that it will be a bit slower over the serial connection.

IMPORTANT: MFGDRV.SYS assumes that other software does not reprogram the COM port being used on the host for its purposes, and that it has exclusive access to it. If you run other software, such as terminal emulation programs, they may disable the COM port UART, causing MFGDRV.SYS to appear to stop working. It is best to avoid running such software on the host when MFGDRV.SYS is loaded. Note: HOST.EXE is an example of such a program, since it takes over the UART for its own purposes. If you run HOST.EXE when MFGDRV.SYS is loaded, you must reboot the host PC for the MFGDRV.SYS driver to reestablish its control over the UART.

For more information about Manufacturing Mode, please contact your General Software Account Manager to obtain a copy of the *Embedded BIOS 2000 Technical Reference Manual*.

This page intentionally left blank.

CHAPTER 5

CONSOLE REDIRECTION

5.1 Using Console Redirection

The DBSC1200 development board can operate either with a standard PC/AT or PS/2 keyboard and VGA video monitor, or with a special emulation of a console over an RS232 cable connected to a host computer running a terminal emulation program. See Figure 7.1 for an example of a HyperTerminal session.

To use the Console Redirection feature, simply connect the target to a HOST computer with an RS232 cable. The HOST computer needs to be running a terminal emulation program that supports ANSI terminal mode, using 115200 baud, no parity, one stop bit, and set to not use flow control (Note: This can be modified by the OEM during BIOS adaptation.) Then, boot the DBSC1200 development board and press the DEL key to reach the System BIOS Setup menu. Next, enter the Features Setup menu and set Console Redirection to **[Auto]**. Exit the Features Setup screen to the System BIOS Setup menu and select **[Write to CMOS and Exit]**.

The target makes three checks during POST to determine if console redirection should be enabled, or if the default video system (if any) should be used. First, if a cable is not connected to a booted host, then it does not use console redirection. Second, if a cable is detected, then if the communication program is not running, it does not enable console redirection. Finally, if the communication program is running with the cable connected, POST requires the user to type any of three characters (ESC, ENTER/CR, or Ctrl-C). It is recommended that the remote user using console redirection press one of these characters several times during POST (well before video appears) so that the character is received after the power supply has come up but before the POST tests are performed.

If the user presses ESC during this test on the remote console, then POST begins console redirection, and uses the ESC as it would an ESC type-in on the main console, to indicate that it should accelerate the memory test and boot the system more quickly.

If the user presses ENTER (or CR) during this test on the remote console, then POST begins console redirection and does not otherwise use the key press.

If the user presses Ctrl-C during this test on the remote console, then POST begins console redirection, and uses the Ctrl-C as it would a Ctrl-C type-in on the main console, to indicate that it should enter the Setup screen before booting the target.

CAUTION: HyperTerminal’s default setting is to use flow control, which will render the console inoperative. To change this, create a new session, change the flow control setting to **[none]**, save the session, and exit HyperTerminal. Then reinvoke HyperTerminal with the session, and it will operate with the new flow control setting.

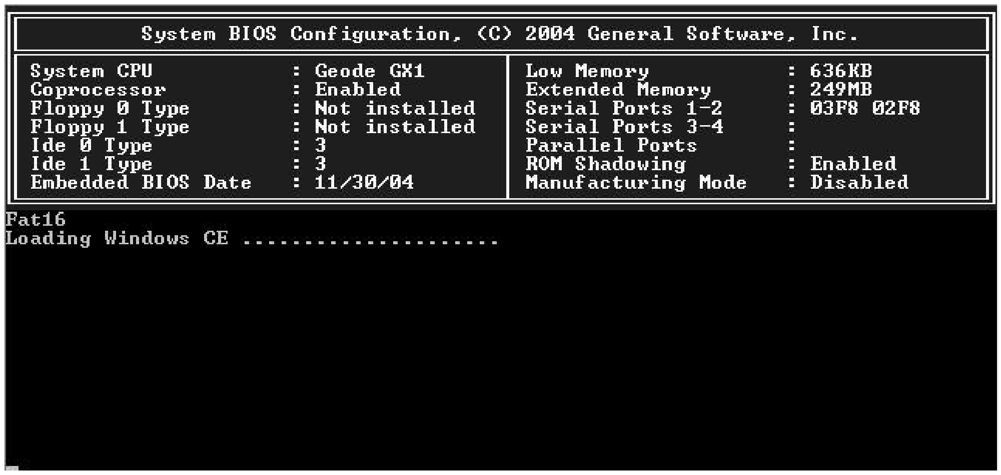
More information about Console Redirection can be found in the *Embedded BIOS 2000 Technical Reference Manual*. Please contact your General Software Account Manager to obtain a copy.

CHAPTER 6

CE-READY—WINDOWS CE LOADER

6.1 Embedded BIOS® 2000 and Windows CE

Windows CE is an embedded operating system provided by Microsoft that is used in embedded applications where a fixed limited version of Windows is desirable. Windows CE is best loaded with the Embedded BIOS 2000 Windows CE loader, so that extra loaders at the DOS prompt are not necessary. Windows CE is configured and adapted to the target with a Windows CE SDK from Microsoft, and the output from that adaptation is a file called NK.BIN. This file can be placed on any FAT 12, FAT 16, or FAT 32 drive controlled by Embedded BIOS 2000, and then the CE Ready feature can load this at boot time.



```
              System BIOS Configuration, (C) 2004 General Software, Inc.

 System CPU            : Geode GX1        Low Memory            : 636KB
 Coprocessor           : Enabled          Extended Memory       : 249MB
 Floppy 0 Type         : Not installed    Serial Ports 1-2      : 03F8 02F8
 Floppy 1 Type         : Not installed    Serial Ports 3-4      :
 Ide 0 Type            : 3                Parallel Ports        :
 Ide 1 Type            : 3                ROM Shadowing         : Enabled
 Embedded BIOS Date    : 11/30/04         Manufacturing Mode    : Disabled
Fat16
Loading Windows CE ......................
```

Figure 6.1. The CE-Ready feature can boot Windows CE natively without requiring a bootstrap O/S loader.

Windows CE differs from other operating systems because all the Windows CE files are packed into a single binary that is loaded into RAM to form a Windows CE-specific, RAM-based file system. Windows CE itself is very compact when compared to Windows XP, because it does not contain the same quantity of drivers nor does it have full Windows 32 support built in. Windows XP contains many drivers and other sub-systems required for full Windows 32 support on multiple platforms, and it is intended to execute using a hard drive-based file system.

To configure your system to boot Windows CE natively from a disk drive, you'll need to have a drive with a FAT file system that contains the `NK.BIN` image in the root directory. Once it is connected to your DBSC1200 development board, enter Setup and go to the Basic CMOS Configuration Setup Screen. Set the Boot Method field to **[Windows CE]**. Next, assign the drive to a drive letter, such as drive C:, configure the drive's geometry, and set the boot order. Then save the configuration and exit CMOS, rebooting the system. After POST, the configuration box should be displayed (Figure 6.1) followed by the message **[Loading Windows CE...]**. This indicates that the loading process is continuing. Once fully loaded, Windows CE takes over the system and runs using the standard PC keyboard, screen, and PS/2 mouse.

Not every `NK.BIN` image will work with the loader as it is configured. Some versions of `NK.BIN` have been customized for a particular platform, and some PC platforms require specific drivers to be built into the Windows CE image in order to operate properly. So, if the hardware does not match what the `NK.BIN` image was intended for (i.e., not enough memory), the image may not boot or function properly. The generic PC project for Windows CE will work for most 486+ systems, provided that you have a standard PS/2 mouse and a video card that supports standard VGA at a resolution of 320 x 200 x 256 colors. (These settings can be modified in the Embedded BIOS 2000 Adaptation Kit.)

Windows CE 3.0, 4.0, .NET, and 5.0 are supported. Future versions may or may not be supported; contact General Software for more information.

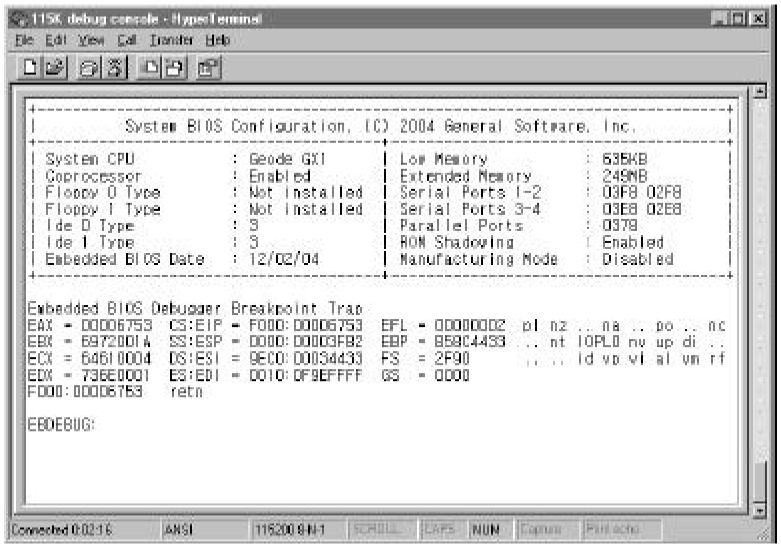
Note that this support is generic to some extent; the Windows CE boot loader can be configured to load any raw binary from a FAT file system, without the need for an intermediate operating system such as DOS.

CHAPTER 7

INTEGRATED BIOS DEBUGGER

7.1 Using the Integrated BIOS Debugger

The DBSC1200 development board BIOS contains a built-in debugger that can be a valuable tool to aid the board bring-up process on new designs similar to the reference board. It supports a DOS SYMDEB-style command line interface, and can be used on the main console's keyboard and screen, or over a redirected connection to a terminal program. (See Chapter 5, Console Redirection.)


115K debug console - HyperTerminal
File Edit View Call Transfer Help
Connected 0:02:16
ANSI
115200 8-N-1
SCROLL
CAPS
NUM
Capture
Print echo

```
+------------------------------------------------------------------------------+
|          System BIOS Configuration, (C) 2004 General Software, Inc.          |
+---------------------------------------+--------------------------------------+
| System CPU          : Geode GX1       | Low Memory           : 635KB         |
| Coprocessor         : Enabled         | Extended Memory      : 249MB         |
| Floppy 0 Type       : Not installed   | Serial Ports 1-2     : 03F8 02F8     |
| Floppy 1 Type       : Not installed   | Serial Ports 3-4     : 03E8 02E8     |
| Ide 0 Type          : 3               | Parallel Ports       : 0378          |
| Ide 1 Type          : 3               | ROM Shadowing        : Enabled       |
| Embedded BIOS Date  : 12/02/04        | Manufacturing Mode   : Disabled      |
+---------------------------------------+--------------------------------------+

Embedded BIOS Debugger Breakpoint Trap
EAX = 00006753  CS:EIP = F000:00006753  EFL = 00000002  pl nz .. na .. po .. nc
EBX = 6972001A  SS:ESP = 0000:00003FB2  EBP = 858C4433  .. nt IOPL0 nv up di ..
ECX = 64610004  DS:ESI = 9EC0:00034433  FS  = 2F90      .. .. id vp vi al vm rf
EDX = 736E0001  ES:EDI = 0010:0F9EFFFF  GS  = 0000
F000:00006753   retn

EBDEBUG:
```

Figure 7.1. The Integrated BIOS Debugger can run over a remote terminal connection or on the main console.

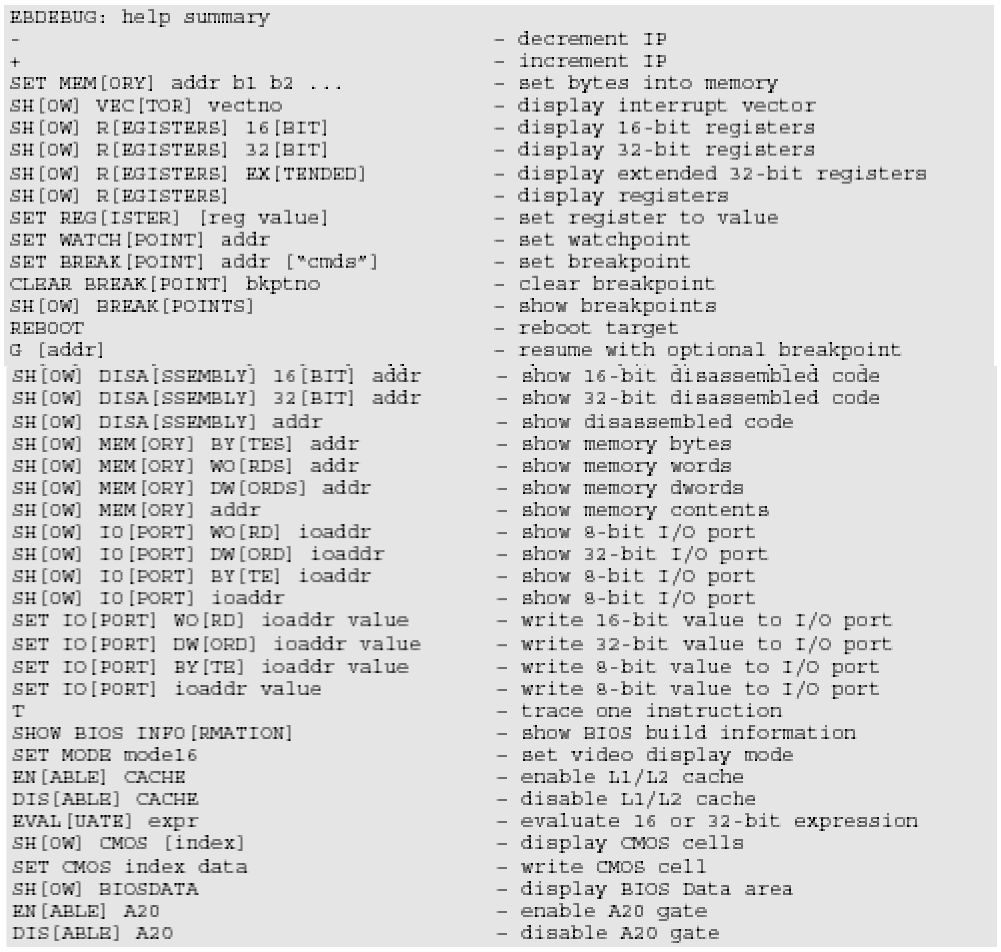
To activate the debugger at any time from the main console, press left-Ctrl-left-Shift together. A display similar to the one in the HyperTerminal session shown here (Figure 7.1) will appear, containing the title, **[Embedded BIOS 2000 Debugger Breakpoint Trap]** and a snapshot of the CPU general registers.

To leave the debugger and resume the interrupted activity (whether POST, BIOS, DOS, Windows, or an application program), enter the **[G]** command (short for "go") and press ENTER. If you were at a DOS prompt when you entered the debugger, then DOS will still be waiting for its command, and will not prompt again until you press ENTER again.

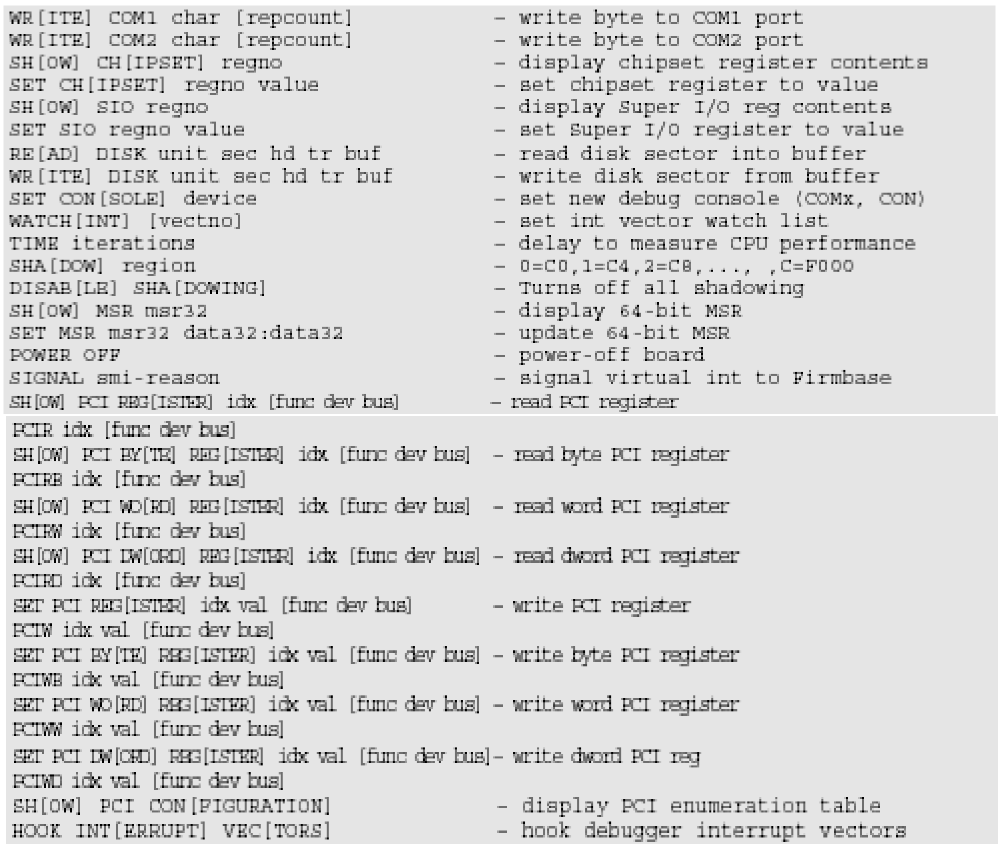
The debugger can also be entered from the Basic Setup Screen (Figure 3.2). In addition, if no bootable device is available, you may have Embedded BIOS 2000 boot directly to the debugger.

If your version of DOS, an application, or any OEM-supplied BIOS extensions have debugging code (i.e., INT 3 instructions) remaining, then these will invoke the debugger automatically, although this is not an error. To continue, use the **[G]** command. When Embedded BIOS 2000 is adapted by the OEM, the debugger can be removed from the final production BIOS, and superfluous debugging code in the application will not cause the debugger to be invoked.

The BIOS debugger is highly configurable from within each adaptation, so type "help" at the debugger command prompt to see a list of currently supported commands along with the list of parameters needed (Figure 7.2).



```
EBDEBUG: help summary
-                                       - decrement IP
+                                       - increment IP
SET MEM[ORY] addr b1 b2 ...             - set bytes into memory
SH[OW] VEC[TOR] vectno                  - display interrupt vector
SH[OW] R[EGISTERS] 16[BIT]              - display 16-bit registers
SH[OW] R[EGISTERS] 32[BIT]              - display 32-bit registers
SH[OW] R[EGISTERS] EX[TENDED]           - display extended 32-bit registers
SH[OW] R[EGISTERS]                      - display registers
SET REG[ISTER] [reg value]              - set register to value
SET WATCH[POINT] addr                   - set watchpoint
SET BREAK[POINT] addr ["cmds"]          - set breakpoint
CLEAR BREAK[POINT] bkptno               - clear breakpoint
SH[OW] BREAK[POINTS]                    - show breakpoints
REBOOT                                  - reboot target
G [addr]                                - resume with optional breakpoint
SH[OW] DISA[SSEMBLY] 16[BIT] addr       - show 16-bit disassembled code
SH[OW] DISA[SSEMBLY] 32[BIT] addr       - show 32-bit disassembled code
SH[OW] DISA[SSEMBLY] addr               - show disassembled code
SH[OW] MEM[ORY] BY[TES] addr            - show memory bytes
SH[OW] MEM[ORY] WO[RDS] addr            - show memory words
SH[OW] MEM[ORY] DW[ORDS] addr           - show memory dwords
SH[OW] MEM[ORY] addr                    - show memory contents
SH[OW] IO[PORT] WO[RD] ioaddr           - show 8-bit I/O port
SH[OW] IO[PORT] DW[ORD] ioaddr          - show 32-bit I/O port
SH[OW] IO[PORT] BY[TE] ioaddr           - show 8-bit I/O port
SH[OW] IO[PORT] ioaddr                  - show 8-bit I/O port
SET IO[PORT] WO[RD] ioaddr value        - write 16-bit value to I/O port
SET IO[PORT] DW[ORD] ioaddr value       - write 32-bit value to I/O port
SET IO[PORT] BY[TE] ioaddr value        - write 8-bit value to I/O port
SET IO[PORT] ioaddr value               - write 8-bit value to I/O port
T                                       - trace one instruction
SHOW BIOS INFO[RMATION]                 - show BIOS build information
SET MODE mode16                         - set video display mode
EN[ABLE] CACHE                          - enable L1/L2 cache
DIS[ABLE] CACHE                         - disable L1/L2 cache
EVAL[UATE] expr                         - evaluate 16 or 32-bit expression
SH[OW] CMOS [index]                     - display CMOS cells
SET CMOS index data                     - write CMOS cell
SH[OW] BIOSDATA                         - display BIOS Data area
EN[ABLE] A20                            - enable A20 gate
DIS[ABLE] A20                           - disable A20 gate
```



```
WR[ITE] COM1 char [repcount]           - write byte to COM1 port
WR[ITE] COM2 char [repcount]           - write byte to COM2 port
SH[OW] CH[IPSET] regno                 - display chipset register contents
SET CH[IPSET] regno value              - set chipset register to value
SH[OW] SIO regno                       - display Super I/O reg contents
SET SIO regno value                    - set Super I/O register to value
RE[AD] DISK unit sec hd tr buf         - read disk sector into buffer
WR[ITE] DISK unit sec hd tr buf        - write disk sector from buffer
SET CON[SOLE] device                   - set new debug console (COMx, CON)
WATCH[INT] [vectno]                    - set int vector watch list
TIME iterations                        - delay to measure CPU performance
SHA[DOW] region                        - 0=C0,1=C4,2=C8,..., ,C=F000
DISAB[LE] SHA[DOWING]                  - Turns off all shadowing
SH[OW] MSR msr32                       - display 64-bit MSR
SET MSR msr32 data32:data32            - update 64-bit MSR
POWER OFF                              - power-off board
SIGNAL smi-reason                      - signal virtual int to Firmbase
SH[OW] PCI REG[ISTER] idx [func dev bus]        - read PCI register
PCIR idx [func dev bus]
SH[OW] PCI BY[TE] REG[ISTER] idx [func dev bus]   - read byte PCI register
PCIRB idx [func dev bus]
SH[OW] PCI WO[RD] REG[ISTER] idx [func dev bus]   - read word PCI register
PCIRW idx [func dev bus]
SH[OW] PCI DW[ORD] REG[ISTER] idx [func dev bus] - read dword PCI register
PCIRD idx [func dev bus]
SET PCI REG[ISTER] idx val [func dev bus]       - write PCI register
PCIW idx val [func dev bus]
SET PCI BY[TE] REG[ISTER] idx val [func dev bus] - write byte PCI register
PCIWB idx val [func dev bus]
SET PCI WO[RD] REG[ISTER] idx val [func dev bus] - write word PCI register
PCIWW idx val [func dev bus]
SET PCI DW[ORD] REG[ISTER] idx val [func dev bus]- write dword PCI reg
PCIWD idx val [func dev bus]
SH[OW] PCI CON[FIGURATION]              - display PCI enumeration table
HOOK INT[ERRUPT] VEC[TORS]              - hook debugger interrupt vectors
```

Figure 7.2. The Integrated BIOS Debugger commands with input parameters.

More information about the BIOS Debugger can be found in the *Embedded BIOS 2000 Technical Reference Manual*. Please contact your General Software Account Manager to obtain a copy of this valuable reference tool.

This page intentionally left blank.

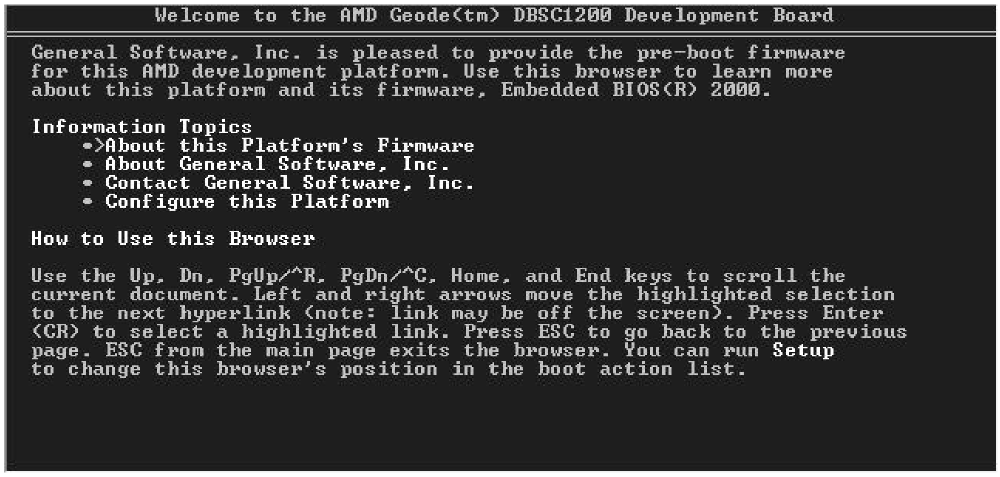
CHAPTER 8

INTEGRATED HTML BROWSER

8.1 Using the Browser

The DBSC1200 development board BIOS contains a built-in HTML browser that allows the user to browse (in text mode using the keyboard instead of the mouse) HTML documents stored in the BIOS ROM.

The HTML browser can be a boot action item, and by default is the primary boot action. Thus, when a platform boots for the first time, it should cause the browser to display its first page. The HTML pages for a platform are created using the Embedded BIOS 2000 Adaptation Kit. The ones for this platform were designed to provide the user with information about the platform, General Software, the firmware, and how to obtain technical support for the platform.



```
           Welcome to the AMD Geode(tm) DBSC1200 Development Board

 General Software, Inc. is pleased to provide the pre-boot firmware
 for this AMD development platform. Use this browser to learn more
 about this platform and its firmware, Embedded BIOS(R) 2000.

 Information Topics
      •>About this Platform's Firmware
      • About General Software, Inc.
      • Contact General Software, Inc.
      • Configure this Platform

 How to Use this Browser

 Use the Up, Dn, PgUp/^R, PgDn/^C, Home, and End keys to scroll the
 current document. Left and right arrows move the highlighted selection
 to the next hyperlink (note: link may be off the screen). Press Enter
 (CR) to select a highlighted link. Press ESC to go back to the previous
 page. ESC from the main page exits the browser. You can run Setup
 to change this browser's position in the boot action list.
```

Figure 8.1. The Integrated HTML Browser can be run over a remote terminal connection or from the main console.

General Software ODM/OEM customers can use the Embedded BIOS 2000 Adaptation Kit to create their own HTML user interface for their embedded device, simply by writing HTML. The HTML browser works over the main console, and over console redirection as well.

The main page (Figure 8.1) is automatically displayed when the browser is invoked. To exit a page and go to a previous page (or from the main page, to exit the browser), simply press the ESC key.

To scroll up and down, use the Up and Down arrow keys on your keypad (making sure NumLock is turned off in order for these keys to work). To advance the cursor to the previous or next hyperlink (highlighted in a different color), use the Left and Right arrow keys on your keypad. You may also use the PgUp, PgDn, Home, and End keys to navigate the current page.

To select the hyperlink highlighted by the cursor, press ENTER (CR). This will cause the new page to be displayed. To get back to the previous page, press ESC.

The browser can display HTML text and can also invoke BIOS functionality (and even cause OEM code to be executed) with special embedded targets. This is demonstrated in the demonstration HTML for this platform, allowing the user to enter the Setup screen and perform other maintenance functions from within the browser's display.

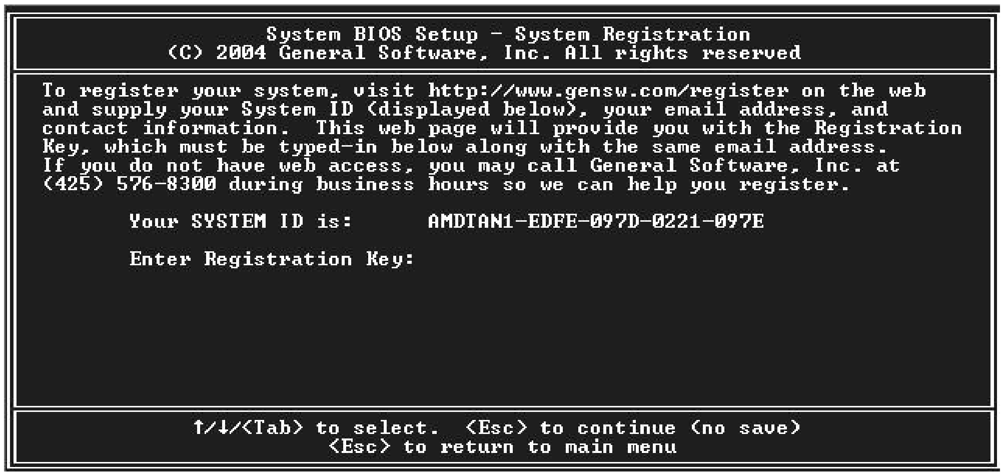
A complete discussion of the browser, including the limited set of supported HTML tags, is beyond the scope of this chapter; however, additional documentation is available from General Software.

CHAPTER 9

SYSTEM REGISTRATION

9.1 Registering Your System

The DBSC1200 development board BIOS can incorporate a system registration facility that provides a method for users to obtain technical support from General Software directly. As the supplier of the BIOS for this platform, General Software is ideally positioned to provide the technical support that embedded ODM/OEM user's need.



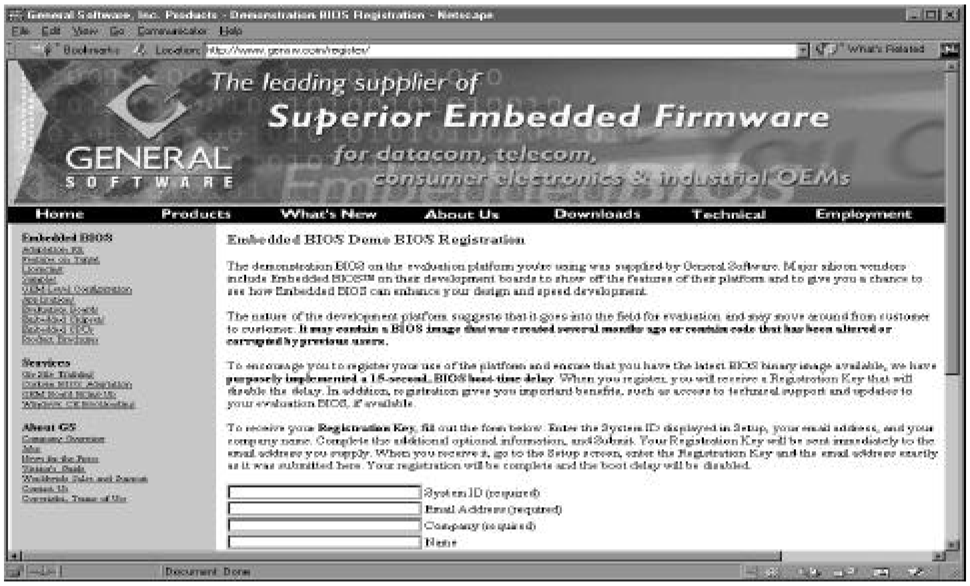
```
                  System BIOS Setup - System Registration
           (C) 2004 General Software, Inc. All rights reserved

 To register your system, visit http://www.gensw.com/register on the web
 and supply your System ID (displayed below), your email address, and
 contact information.  This web page will provide you with the Registration
 Key, which must be typed-in below along with the same email address.
 If you do not have web access, you may call General Software, Inc. at
 (425) 576-8300 during business hours so we can help you register.

        Your SYSTEM ID is:      AMDIAN1-EDFE-097D-0221-097E

        Enter Registration Key:

              ↑/↓/<Tab> to select.  <Esc> to continue (no save)
                        <Esc> to return to main menu
```

Figure 9.1. The Registration Screen, accessed through the board's Setup screen, provides the system's identification string and registration key for support services access.

Registration of this system provides two important benefits. The first is to provide access to General Software's technical support services. The second is to reduce the intentionally set boot-time delay, resulting in a faster system during startup.

To register your system, enter the Setup main menu and select **[Register System]**. Then, follow the directions on that screen (Figure 9.1). This screen will display your system's ID, which should be copied down and then entered into General Software's on-line registration web page (Figure 9.2).

To obtain the registration key associated with your system, visit General Software's web site at **www.gensw.com/register**. You should see a page similar to that shown in Figure 9.2.


General Software, Inc. Products - Demonstration BIOS Registration - Netscape
File Edit View Go Communicator Help
Bookmarks Location: http://www.gensw.com/register/ What's Related
The leading supplier of
Superior Embedded Firmware
for datacom, telecom,
consumer electronics & industrial OEMs
GENERAL SOFTWARE
Home Products What's New About Us Downloads Technical Employment
Embedded BIOS
Services
About GS
Document: Done

Embedded BIOS Demo BIOS Registration

The demonstration BIOS on the evaluation platform you're using was supplied by General Software. Major silicon vendors include Embedded BIOS™ on their development boards to show off the features of their platform and to give you a chance to see how Embedded BIOS can enhance your design and speed development.

The nature of the development platform suggests that it goes into the field for evaluation and may move around from customer to customer. **It may contain a BIOS image that was created several months ago or contain code that has been altered or corrupted by previous users.**

To encourage you to register your use of the platform and ensure that you have the latest BIOS binary image available, we have **purposely implemented a 15-second, BIOS boot-time delay**. When you register, you will receive a Registration Key that will disable the delay. In addition, registration gives you important benefits, such as access to technical support and updates to your evaluation BIOS, if available.

To receive your **Registration Key**, fill out the form below. Enter the System ID displayed in Setup, your email address, and your company name. Complete the additional optional information, and Submit. Your Registration Key will be sent immediately to the email address you supply. When you receive it, go to the Setup screen, enter the Registration Key and the email address exactly as it was submitted here. Your registration will be complete and the boot delay will be disabled.

System ID (required)
Email Address (required)
Company (required)
Name

Figure 9.2. General Software's registration web site provides users with registration keys.

To receive your registration key and be eligible to receive technical support from General Software, you must supply basic contact information on the web site, including your email address. General Software does not provide this information to third parties, but uses it internally to qualify you for support.

Once you submit the information on this form, a General Software registration server will email your registration key back to you, at the email address you specified when you registered.

If for any reason you are unable to use the on-line registration system, please email General Software at **platform-registration@gensw.com** or contact us by telephone at (425) 576-8300. We will be happy to help you register your system.

Some users of this platform (such as Field Application Engineers or Sales Engineers) may loan the system to other people. General Software encourages the person loaning the system to deregister it by going into the Registration screen and canceling registration. This action will remove the email address from the system's sign-on banner, and encourages the person receiving the board to register the system so they may receive technical support from General Software. Whenever the system moves between users, General Software encourages re-registration of the system.

CHAPTER 10

UNIVERSAL SERIAL BUS

10.1 USB Services

The DBSC1200 development board BIOS replaces the 16-bit AMD-proprietary VSA2 with industry-standard 32-bit Firmbase 1.3 technology. Firmbase's USB services provide USB functionality pre-OS and to operating systems that lack USB drivers, for example MS-DOS and Windows NT. The USB services support complex bus topologies incorporating USB hubs, up to eight USB keyboards and mice simultaneously, and a large variety of USB mass storage devices including floppy disk drives, hard disk drives, CD/DVD ROM drives, CompactFlash readers, Zip disk drives, DiskOnKey devices and much more.

Firmbase runs in System Management Mode (SMM) so SMM must be enabled within the BIOS "Features" setup screen shown in Figure 3.8. Legacy USB keyboard and mouse support is licensed with Embedded BIOS 2000 at no additional charge while USB Mass Storage support is available as an option. Please contact General Software for details.

10.2 Legacy USB Keyboard and Mouse

USB keyboards with or without integrated hubs are supported by default on the DBSC1200 development board with SMM enabled. The devices must be connected either directly to the platform's USB ports or through a USB hub prior to power-up in order to make sure they are detected and configured properly. The USB keyboard is also available during POST like a traditional PS/2 device so it can be utilized for BIOS configuration and with the integrated debugger.

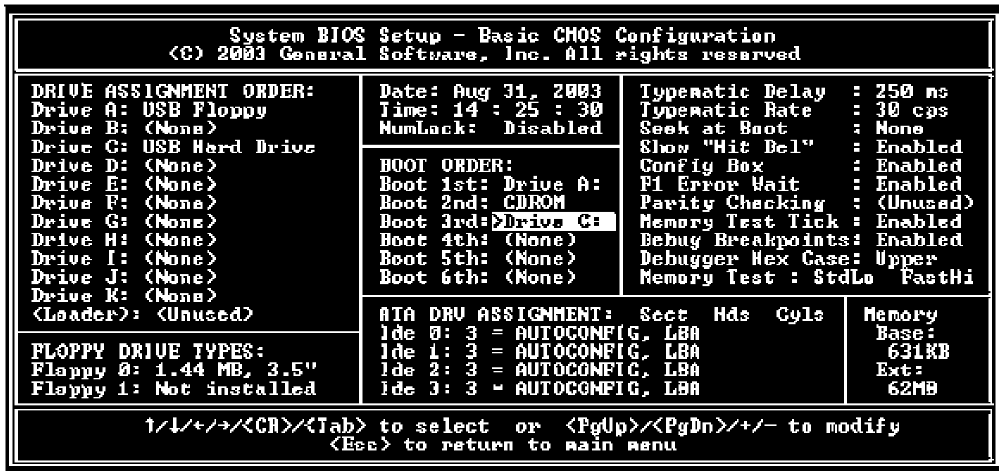
USB mice must also be connected to the DBSC1200 development board prior to power-up so they too can be properly detected and configured during POST. The operating system level driver, for example MOUSE.COM in MS-DOS, will need to be installed to enable complete functionality.

10.3 USB Mass Storage

Firmbase USB Mass Storage enables the user to boot the DBSC1200 development board from a variety of USB storage devices that use no emulation, floppy disk emulation, and hard disk emulation. Some of the USB devices supported include:

- Hard disk drives
- Floppy disk drives
- CD/DVD ROM drives
- CompactFlash reader
- DiskOnKey or memory stick devices
- Zip disk drives

Many of these devices are not configured from the factory to be used as boot devices, therefore it may be necessary to run FDISK /MBR on the device to rewrite the MBR and make it bootable. Any device with a capacity greater then 1.44MB, other than CD/DVD ROM drives, will be treated as a hard disk drive and must be configured as one within the Basic setup screen, Figure 10.1.



```
              System BIOS Setup - Basic CMOS Configuration
         (C) 2003 General Software, Inc. All rights reserved

DRIVE ASSIGNMENT ORDER:   Date: Aug 31, 2003   Typematic Delay   : 250 ms
Drive A: USB Floppy       Time: 14 : 25 : 30   Typematic Rate    : 30 cps
Drive B: (None)           NumLock:  Disabled   Seek at Boot      : None
Drive C: USB Hard Drive                        Show "Hit Del"    : Enabled
Drive D: (None)           BOOT ORDER:          Config Box        : Enabled
Drive E: (None)           Boot 1st: Drive A:   F1 Error Wait     : Enabled
Drive F: (None)           Boot 2nd: CDROM      Parity Checking   : (Unused)
Drive G: (None)           Boot 3rd:>Drive C:   Memory Test Tick  : Enabled
Drive H: (None)           Boot 4th: (None)     Debug Breakpoints: Enabled
Drive I: (None)           Boot 5th: (None)     Debugger Hex Case: Upper
Drive J: (None)           Boot 6th: (None)     Memory Test : StdLo  FastHi
Drive K: (None)
(Loader): (Unused)        ATA DRV ASSIGNMENT:  Sect  Hds  Cyls  Memory
                          Ide 0: 3 = AUTOCONFIG, LBA             Base:
FLOPPY DRIVE TYPES:       Ide 1: 3 = AUTOCONFIG, LBA              631KB
Floppy 0: 1.44 MB, 3.5"   Ide 2: 3 = AUTOCONFIG, LBA             Ext:
Floppy 1: Not installed   Ide 3: 3 = AUTOCONFIG, LBA              62MB

       ↑/↓/←/→/<CR>/<Tab> to select  or  <PgUp>/<PgDn>/+/- to modify
                       <Esc> to return to main menu
```

Figure 10.1. General Software's Basic setup screen configured to use USB Mass Storage registration.

To utilize USB Boot functionality simply modify the drive type within the Drive Assignment Order table and include the device within the Boot Order table. For example, to boot from a USB floppy drive, modify the Drive A: assignment to "USB Floppy" and add Drive A to the Boot Order table. To boot from a USB CD/DVD ROM drive, simply add CDROM to the Boot Order table. In order to boot from all other USB devices, they must be formatted and treated as hard disk drives, then simply modify Drive C to be a "USB Hard Drive" and add Drive C to the Boot Order table. In each case, if is more than one of each device type connected to the system, the first of each type detected will become the boot device.

USB Boot functionality is enabled within the demonstration binary provided for the DBSC1200 development board for evaluation purposes and is an optional Embedded BIOS 2000 core feature. Please contact General Software for additional details.

CHAPTER 11

FIRMBASE®

11.1 Firmbase Support

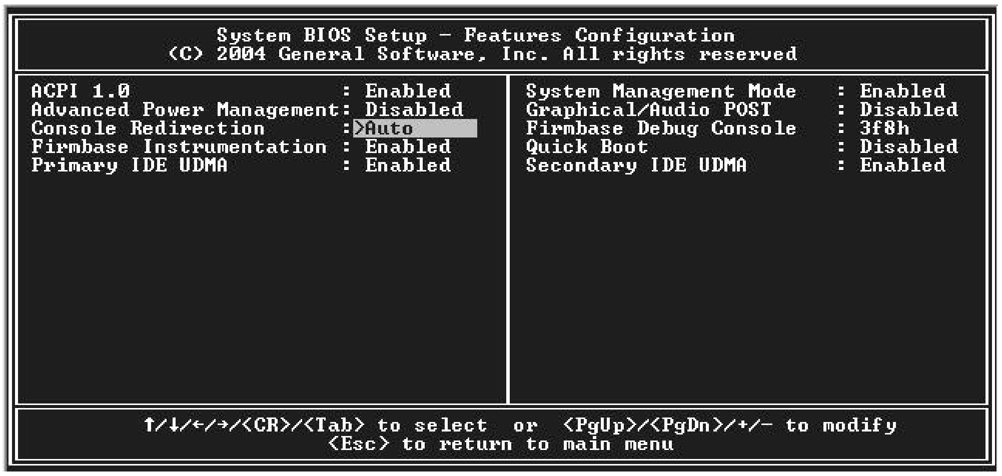
The DBSC1200 development board BIOS is using industry-standard 32-bit Firmbase technology to replace the National Semiconductor's proprietary 16-bit VSA2. Firmbase provides much more than just a replacement for VSA2. Its highly-expandable framework includes:

- Industry-standard, replaces 16-bit VSA2
- Runs 32-bit PE executables from SMM, stored in ROM or disk
- Keeps running even if OS crashes or is missing
- Full USB 2.0 stack, TCP/IP stack, High Availability (HA) API, Remote access

A complete discussion of Firmbase's capabilities is beyond the scope of this Users Guide. For more information, please contact General Software to obtain a copy of the *Firmbase Technical Reference Manual*.

11.2 Firmbase Console Configuration

The Firmbase support for the DBSC1200 development board contains a powerful integrated shell, known as "Demo", offering remote access and debugging capabilities even without the foreground OS. The Demo shell can be accessed either via an RS-232 serial port or via a Telnet connection. To utilize the serial port connection, you must enable "Firmbase Insturmentation" and configure the COM port address for the "Firmbase Debug Console" within the Features setup screen, Figure 11.1.



```
              System BIOS Setup - Features Configuration
          (C) 2004 General Software, Inc. All rights reserved

ACPI 1.0                   : Enabled    System Management Mode    : Enabled
Advanced Power Management: Disabled     Graphical/Audio POST      : Disabled
Console Redirection        :>Auto       Firmbase Debug Console    : 3f8h
Firmbase Instrumentation : Enabled      Quick Boot                : Disabled
Primary IDE UDMA           : Enabled    Secondary IDE UDMA        : Enabled

        ↑/↓/←/→/<CR>/<Tab> to select  or  <PgUp>/<PgDn>/+/- to modify
                        <Esc> to return to main menu
```

Figure 11.1. General Software's Firmbase console enabled and configured to use COM1.

After the proper values have been set, save your new values to CMOS as you exit the BIOS setup system and the system will reboot.

In addition to the proper setup within the BIOS, you will also need to configure a Host system to communicate with the Firmbase console. The HOST computer needs to be running a terminal emulation program that supports ANSI terminal mode, Telnet or Tera Term for example, using 115200 baud, no parity, one stop bit, and set to not use flow control (Note: This can be modified by the OEM during BIOS adaptation.).

Telnet support is also available, but is beyond the scope of this Users Guide. Please contact General Software directly for additional configuration information.

11.2 Firmbase Console Commands

The Firmbase Console provides a powerful set of Firmbase and SMM debug tools that can be expanded to meet the OEM's need using the Firmbase SDK, which is sold separately. Here is an example of just some of the Firmbase Console Commands:

```
Console /DEV/CONSOLE/STDOUT ready
Type INFO or HELP for more information.

/DEV/ROM> help
Command reference:

  help                       - Display this help text
  info                       - Display information about this
demo program
  get                        - Process HTTP request
  exit                       - Terminate this console
  reboot                     - Reboot the target
  off                        - Soft-off the target
  signal     intno           - Signal virtual interrupt number
  spawn      [fn] [args]     - Run program without wait
  ls         [-l] [filename] - Display directory
  dir        [filename] [/w] - Display directory
  cp         fn1 fn2         - Copy fn1 to fn2
  copy       fn1 fn2         - Copy fn1 to fn2
  cat        filename        - Display file contents
  type       filename        - Display file contents
  bufsize    nbytes          - Set buffer size for TYPE/CAT in bytes
  rlba       devname lba     - Read lba from device
  pwd                        - Print working directory
  cd                         - Change working directory
  mkdir      dirname         - Create directory
  md         dirname         - Create directory
  rmdir      dirname         - Remove directory
  rd         dirname         - Remove directory
  del        filename        - Delete file
  open       filename        - Open file
  close      handle          - Close file handle
  set        keyword value   - Set process string
  bypass     ON|OFF          - Set bypass mode
  ps         [prhndl]        - Display process information
  thread     [prhndl]        - Display thread information
  dev        [devname] [prhn - Display device information
  kill       prhndl          - Kill process
  ht         [handle]        - Display system handle table
  pool       [QOS]           - Display memory pool
  showpool                   - Display memory pool objects
  sysinfo                    - Display system information
  setevent   handle          - Set event object
  clearevent handle          - Clear event object
  disprate   rate_in_ms      - Set system dispatch rate
  dispdur    duration_percen - Set system dispatch duration
  restdispdu handle          - Restore dispatch duration from handle
  chargen    recordsize      - Character generator
  sendmsg    status param    - Send message to loader
  i          port            - Read 8-bit I/O port
  iw         port            - Read 16-bit I/O port
  id         port            - Read 32-bit I/O port
  o          port value      - Write value to 8-bit I/O port
  ow         port value      - Write value to 16-bit I/O port
  od         port value      - Write value to 32-bit I/O port
  e          physaddr byte [ - Enter byte(s) in memory
  d          [physaddr]      - Dump memory contents
  db         [physaddr]      - Dump memory contents in bytes
  dw         [physaddr]      - Dump memory contents in words
  dd         [physaddr]      - Dump memory contents in dwords
  pcirb      idx fn dev bus  - Read PCI config byte
  pcirw      idx fn dev bus  - Read PCI config word
  pcird      idx fn dev bus  - Read PCI config dword
  pciwb      idx val fn dev  - Write PCI config byte
  pciww      idx val fn dev  - Write PCI config word
  pciwd      idx val fn dev  - Write PCI config dword
  uhci       [hc]            - Display UHCI USB HC Status
  frame      [hc] framenum   - Display Frame List Entry
  smi        [hc] millisecs  - Generate one SMI via HC
  smitrain   [hc] millisecs  - Generate repeating SMI via HC
  tdp        [1|0]           - Enable/Disable TD polling
  qh         addr            - Display UHCI Queue Head Object
  td         addr            - Display UHCI Transfer Descriptor
Object
  ep         addr            - Display HCD Endpoint Object
  oe         devname         - Open HCD endpoint by device name
  ce         handle          - Close HCD endpoint by handle
  ba         handle UsbAdr   - Bind endpoint to USB address
  be         handle UsbEpt   - Bind endpoint to USB endpoint
  bp         handle prot qos - Bind endpoint to USB protocol
  dt         handle toggleva - Set data toggle on endpoint
  ss         handle speedval - Set speed on endpoint
  sp         handle port sta - Set root hub port state
  dp         handle port     - Display root hub port information
  read       handle nbytes   - Read data from device
  gs         handle          - Get status from device
  ru16       [hc] reg        - Read 16-bit register from UHCI HC
  ru32       [hc] reg        - Read 32-bit register from UHCI HC
  wu16       [hc] reg val16  - Write 16-bit UHCI HC register
  wu32       [hc] reg val32  - Write 32-bit UHCI HC register
  aeh                        - Allocate Emulation Handler
  deh                        - Deallocate Emulation Handler
  sptm       mask            - Set port trap mask
  gptm                       - Get port trap mask
  sdte       data [count]    - Send data to emulator
  rf         devname lba32   - Read 512-byte sector from floppy.
  rc         devname lba32   - Read 2048-byte sector from cdrom.
  tur                        - Issue TEST_UNIT_READY to latest USB device.
  modesense                  - Issue MODE_SENSE to latest USB device.
  reqsense                   - Issue REQUEST_SENSE to latest USB device.
  readcap                    - Issue READ_CAPACITY to latest USB device.
  readtoc                    - Issue READ_TOC to latest USB device.
  read10                     - Issue READ_10 to latest USB device.
  startunit                  - Issue START_UNIT to latest USB device.
  getevent                   - Issue GET_EVENT_STATUS to latest USB device.
  botreset                   - Issue BOT Mass Storage Reset to latest USB
device.
  rkbc                       - Reset keyboard controller
  esio                       - Enable SIO 8042
  dsio                       - Disable SIO 8042
  gkbcm                      - Get KBC mode byte
  skbcm      KBC mode byte   - Set KBC mode byte
  gstat                      - Get KBC status byte
  sstat      KBC mode byte   - Set KBC status byte
  o60        data            - out 60h
  o64        data            - out 64h
  sch        data            - Send character
  ech        data            - Enqueue character
  debugdemo  [any args]      - Does nothing

Type INFO for more information.

/DEV/ROM>
```

Figure 11.1. General Software's Firmbase Demo console commands.

For additional information about Firmbase, please contact General Software.

APPENDIX A

TROUBLESHOOTING POST

Embedded BIOS 2000 POST Codes

This appendix provides a reference for debug aids useful in diagnosing booting. Embedded BIOS 2000 writes progress codes, also known as POST codes, to I/O port 80h during POST, in order to provide information to OEM developers about system faults. These POST codes may be monitored by a port 80h card in either an ISA slot or PCI slot; they are not displayed on the screen. For more information about POST codes, contact General Software. Please note that the Embedded BIOS 2000 adaptation may be configured to reroute these codes over another I/O port or device.

Mnemonic Code	Code	System Progress Report
POST_STATUS_START	00h	Start POST (BIOS is executing).
POST_STATUS_CPUTEST	01h	Start CPU register test.
POST_STATUS_DELAY	02h	Start power-on delay.
POST_STATUS_DELAYDONE	03h	Power-on delay finished.
POST_STATUS_KBDBATRDY	04h	Keyboard BAT finished.
POST_STATUS_DISABSHADOW	05h	Disable shadowing & cache.
POST_STATUS_CALCCKSUM	06h	Compute ROM CRC, wait for KBC.
POST_STATUS_CKSUMGOOD	07h	CRC okay, KBC ready.
POST_STATUS_BATVRFY	08h	Verifying BAT command to KB.
POST_STATUS_KBDCMD	09h	Start KBC command.
POST_STATUS_KBDDATA	0ah	Start KBC data.
POST_STATUS_BLKUNBLK	0bh	Start pin 23,24 blocking & unblocking.
POST_STATUS_KBDNOP	0ch	Start KBC NOP command.
POST_STATUS_SHUTTEST	0dh	Test CMOS RAM shutdown register.
POST_STATUS_CMOSDIAG	0eh	Check CMOS checksum.
POST_STATUS_CMOSINIT	0fh	Initialize CMOS contents.
POST_STATUS_CMOSSTATUS	10h	Initialize CMOS status for date/time.

Mnemonic Code	Code	System Progress Report (continued)
POST_STATUS_DISABDMAINT	11h	Disable DMA, PICs.
POST_STATUS_DISABPORTB	12h	Disable Port B, video display.
POST_STATUS_BOARD	13h	Initialize board, start memory detection.
POST_STATUS_TESTTIMER	14h	Start timer tests.
POST_STATUS_TESTTIMER2	15h	Test 8254 T2, for speaker, port B.
POST_STATUS_TESTTIMER1	16h	Test 8254 T1, for refresh.
POST_STATUS_TESTTIMER0	17h	Test 8254 T0, for 18.2Hz.
POST_STATUS_MEMREFRESH	18h	Start memory refresh.
POST_STATUS_TESTREFRESH	19h	Test memory refresh.
POST_STATUS_TEST15US	1ah	Test 15usec refresh ON/OFF time.
POST_STATUS_TEST64KB	1bh	Test base 64KB memory.
POST_STATUS_TESTDATA	1ch	Test data lines.
POST_STATUS_TESTADDR	20h	Test address lines.
POST_STATUS_TESTPARITY	21h	Test parity (toggling).
POST_STATUS_TESTMEMRDWR	22h	Test Base 64KB memory.
POST_STATUS_SYSINIT	23h	Prepare system for IVT initialization.
POST_STATUS_INITVECTORS	24h	Initialize vector table.
POST_STATUS_8042TURBO	25h	Read 8042 for turbo switch setting.
POST_STATUS_POSTTURBO	26h	Initialize turbo data.
POST_STATUS_POSTVECTORS	27h	Modification of IVT.
POST_STATUS_MONOMODE	28h	Video in monochrome mode verified.
POST_STATUS_COLORMODE	29h	Video in color mode verified.
POST_STATUS_TOGGLEPARITY	2ah	Toggle parity before video ROM test.
POST_STATUS_INITBEFOREVIDEO	2bh	Initialize before video ROM check.
POST_STATUS_VIDEOROM	2ch	Passing control to video ROM.
POST_STATUS_POSTVIDEO	2dh	Control returned from video ROM.
POST_STATUS_CHECKEGAVGA	2eh	Check for EGA/VGA adapter.
POST_STATUS_TESTVIDEOMEMORY	2fh	No EGA/VGA found, test video memory.
POST_STATUS_RETRACE	30h	Scan for video retrace signal.
POST_STATUS_ALTDISPLAY	31h	Primary retrace failed.
POST_STATUS_ALTRETRACE	32h	Alternate found.
POST_STATUS_VRFYSWADAPTER	33h	Verify video switches.
POST_STATUS_SETDISPMODE	34h	Establish display mode.
POST_STATUS_CHECKSEG40A	35h	Initialize ROM BIOS data area.
POST_STATUS_SETCURSOR	36h	Set cursor for power-on msg.
POST_STATUS_PWRONDISPLAY	37h	Display power-on message.
POST_STATUS_SAVECURSOR	38h	Save cursor position.
POST_STATUS_BIOSIDENT	39h	Display BIOS identification string.
POST_STATUS_HITDEL	3ah	Display “Hit <DEL> to ...” message.
POST_STATUS_VIRTUAL	40h	Prepare protected mode test.
POST_STATUS_DESCR	41h	Prepare descriptor tables.
POST_STATUS_ENTERVM	42h	Enter virtual mode for memory test.
POST_STATUS_ENABINT	43h	Enable interrupts for diagnostics mode.
POST_STATUS_CHECKWRAP1	44h	Initialize data for memory wrap test.
POST_STATUS_CHECKWRAP2	45h	Test for wrap, find total memory size.
POST_STATUS_HIGHPATTERNS	46h	Write extended memory test patterns.
POST_STATUS_LOWPATTERNS	47h	Write conventional memory test patterns.
POST_STATUS_FINDLOWMEM	48h	Find low memory size from patterns.
POST_STATUS_FINDHIMEM	49h	Find high memory size from patterns.
POST_STATUS_CHECKSEG40B	4ah	Verify ROM BIOS data area again.
POST_STATUS_CHECKDEL	4bh	Check for <DEL> pressed.

Mnemonic Code	Code	System Progress Report (continued)
POST_STATUS_CLREXTMEM	4ch	Clear extended memory for soft reset.
POST_STATUS_SAVEMEMSIZE	4dh	Save memory size.
POST_STATUS_COLD64TEST	4eh	Cold boot: Display 1st 64KB memtest.
POST_STATUS_COLDLOWTEST	4fh	Cold boot: Test all of low memory.
POST_STATUS_ADJUSTLOW	50h	Adjust memory size for EBDA usage.
POST_STATUS_COLDHITEST	51h	Cold boot: Test high memory.
POST_STATUS_REALMODETEST	52h	Prepare for shutdown to real mode.
POST_STATUS_ENTERREAL	53h	Return to real mode.
POST_STATUS_SHUTDOWN	54h	Shutdown successful.
POST_STATUS_DISABA20	55h	Disable A20 line.
POST_STATUS_CHECKSEG40C	56h	Check ROM BIOS data area again.
POST_STATUS_CHECKSEG40D	57h	Check ROM BIOS data area again.
POST_STATUS_CLRHITDEL	58h	Clear "Hit <DEL> " message.
POST_STATUS_TESTDMAPAGE	59h	Test DMA page register file.
POST_STATUS_VRFYDISPMEM	60h	Verify from display memory.
POST_STATUS_TESTDMA0BASE	61h	Test DMA0 base register.
POST_STATUS_TESTDMA1BASE	62h	Test DMA1 base register.
POST_STATUS_CHECKSEG40E	63h	Checking ROM BIOS data area again.
POST_STATUS_CHECKSEG40F	64h	Checking ROM BIOS data area again.
POST_STATUS_PROGDMA	65h	Program DMA controllers.
POST_STATUS_INITINTCTRL	66h	Initialize PICs.
POST_STATUS_STARTKBDTEST	67h	Start keyboard test.
POST_STATUS_KBDRESET	80h	Issue KB reset command.
POST_STATUS_CHECKSTUCKKEYS	81h	Check for stuck keys.
POST_STATUS_INITCIRCBUFFER	82h	Initialize circular buffer.
POST_STATUS_CHECKLOCKEDKEYS	83h	Check for locked keys.
POST_STATUS_MEMSIZEMISMATCH	84h	Check for memory size mismatch.
POST_STATUS_PASSWORD	85h	Check for password or bypass setup.
POST_STATUS_BEFORESETUP	86h	Pwd check. Do programming before setup
POST_STATUS_CALLSETUP	87h	Entering setup system.
POST_STATUS_POSTSETUP	88h	Setup system exited.
POST_STATUS_DISPPWRON	89h	Display power-on screen message.
POST_STATUS_DISPWAIT	8ah	Display "Wait…" message.
POST_STATUS_ENABSHADOW	8bh	Shadow system & video BIOS.
POST_STATUS_STDCMOSSETUP	8ch	Load standard setup values from CMOS.
POST_STATUS_MOUSE	8dh	Test and initialize mouse.
POST_STATUS_FLOPPY	8eh	Test floppy disks.
POST_STATUS_CONFIGFLOPPY	8fh	Configure floppy drives.
POST_STATUS_IDE	90h	Test hard disks.
POST_STATUS_CONFIGIDE	91h	Configure IDE drives.
POST_STATUS_CHECKSEG40G	92h	Checking ROM BIOS data area.
POST_STATUS_CHECKSEG40H	93h	Checking ROM BIOS data area.
POST_STATUS_SETMEMSIZE	94h	Set base & extended memory sizes.
POST_STATUS_SIZEADJUST	95h	Adjust low memory size for EBDA.
POST_STATUS_INITBEFOREC8000	96h	Initialize before calling C800h ROM.
POST_STATUS_CALLC8000	97h	Call ROM BIOS extension at C800h.
POST_STATUS_POSTC8000	98h	ROM C800h extension returned.
POST_STATUS_TIMERPRNBASE	99h	Configure timer/printer data.
POST_STATUS_SERIALBASE	9ah	Configure serial port base addresses.
POST_STATUS_INITBEFORENPX	9bh	Prepare to initialize coprocessor.
POST_STATUS_INITNPX	9ch	Initialize numeric coprocessor.

Mnemonic Code	Code	System Progress Report (continued)
POST_STATUS_POSTNPX	9dh	Numeric coprocessor initialized.
POST_STATUS_CHECKLOCKS	9eh	Check KB settings.
POST_STATUS_ISSUEKBDID	9fh	Issue keyboard ID command.
POST_STATUS_RESETID	0a0h	KB ID flag reset.
POST_STATUS_TESTCACHE	0a1h	Test cache memory.
POST_STATUS_DISPSOFTERR	0a2h	Display soft errors.
POST_STATUS_TYPEMATIC	0a3h	Set keyboard typematic rate.
POST_STATUS_MEMWAIT	0a4h	Program memory wait states.
POST_STATUS_CLRSCR	0a5h	Clear screen.
POST_STATUS_ENABPTYNMI	0a6h	Enable parity and NMIs.
POST_STATUS_INITBEFOREE000	0a7h	Initialize before calling ROM at E000h.
POST_STATUS_CALLE000	0a8h	Call ROM BIOS extension at E000h.
POST_STATUS_POSTE000	0a9h	ROM extension returned.
POST_STATUS_DISPCONFIG	0b0h	Display system configuration box.
POST_STATUS_INT19BOOT	00h	Call INT 19h bootstrap loader.
POST_STATUS_LOWMEMEXH	0b1h	Test low memory exhaustively.
POST_STATUS_EXTMEMEXH	0b2h	Test extended memory exhaustively.
POST_STATUS_PCIENUM	0b3h	Enumerate PCI busses.
POST_STATUS_ADMGRINIT	0b4h	Initialize address manager.
POST_STATUS_ADMGRBOOT	0b5h	Preboot address manager callout.
POST_STATUS_HUGEMEMEXH	0b6h	Test huge memory exhaustively.
POST_STATUS_SMBIOSINIT	0b7h	Initialize SMBIOS structure table.
POST_STATUS_FBSIGNAL	0b8h	About to signal Firmbase.
POST_STATUS_MEMMGRINIT	0b9h	About to initialize low small memory mgr.
POST_STATUS_INITDRIVER	0bah	About to initialize driver manager.
POST_STATUS_SMPINIT	0bbh	About to start multiprocessor init.

APPENDIX B

CRITICAL ERROR BEEP CODES

Using Beep Codes for Hardware Troubleshooting

Embedded BIOS 2000 tests much of the hardware early in POST before messages can be displayed on the screen. When system failures are encountered at these early stages, POST uses beep codes (a sequence of tones on the speaker) to identify the source of the error.

The following is a comprehensive list of POST beep codes for the system BIOS. BIOS extensions, such as VGA ROMs and SCSI adapter ROMs, may use their own beep codes, including short/long sequences, or possibly beep codes that sound like the ones below. When diagnosing a system failure, remove these adapters if possible before making a final determination of the actual POST test that failed.

Mnemonic Code	**Beep Count**	**Description of Problem**
POST_BEEP_REFRESH	1	Memory refresh is not working.
POST_BEEP_PARITY	2	Parity error found in 1st 64KB of memory.
POST_BEEP_BASE64KB	3	Memory test of 1st 64KB failed.
POST_BEEP_TIMER	4	T1 timer test failed.
POST_BEEP_CPU	5	CPU test failed.
POST_BEEP_GATEA20	6	Gate A20 test failed.
POST_BEEP_DMA	7	DMA page/base register test failed.
POST_BEEP_VIDEO	8	Video controller test failed.
POST_BEEP_KEYBOARD	9	Keyboard test failed.
POST_BEEP_SHUTDOWN	10	CMOS shutdown register test failed.
POST_BEEP_CACHE	11	External cache test failed.
POST_BEEP_BOARD	12	General board initialization failed.
POST_BEEP_LOWMEM	13	Exhaustive low memory test failed.
POST_BEEP_EXTMEM	14	Exhaustive extended memory test failed.
POST_BEEP_CMOS	15	CMOS restart byte test failed.

Mnemonic Code	Beep Count	Description of Problem (continued)
POST_BEEP_ADDRESS_LINE	16	Address line test failed.
POST_BEEP_DATA_LINE	17	Data line test failed.
POST_BEEP_INTERRUPT	18	Interrupt controller test failed.
POST_BEEP_PASSWORD	1	Incorrect password used to access SETUP.
POST_BEEP_HUGEMEM	19	Exhaustive huge memory test.
POST_BEEP_EBDA_LOC	20	Address manager failed to reloc EBDA.
POST_BEEP_ADDR_MGR	21	Address manager failed to initialize.
POST_BEEP_ADSYNCH	22	Address mgr failed to synch legacy mem parameters.
POST_BEEP_LOMEMMGR	23	Low memory manager failed to initialize.
POST_BEEP_POST_FAIL	24	POST driver failed.
POST_BEEP_PMM	25	PMM failed to initialize.

Embedded BIOS 2000 User's Guide Revision History

Revision 1, March, 2002	Formatting changes and content updates.
Revision 2, September, 2002	Version 5.1 release.
Revision 3, September, 2003	Version 5.2 release.
Revision 4, December, 2004	Version 5.3 release for the AMD Geode™ DBSC1200 development board.