

SBC-486DX

SBC-586

Rev M through P

Copyright 1993
Computer Dynamics, Inc.

7640 Pelham Road
Greenville, SC 29615
864-627-8800

TRADEMARKS

Western Digital® and Paradise® are registered trademarks of Western Digital Corporation.
VESA® is a registered trademark and VBE™ is a trademark of the Video Electronics Standards Association.

COPYRIGHTS

All Western Digital Software, Drivers, and Utilities © 1989-1992 Western Digital Corporation.
All Rights Reserved.
© 1985-1992 Congruent Corporation.
All Rights Reserved.
© 1985-1992 Microsoft Corporation.
All Rights Reserved.

Revised:

October 1997
June 22, 2000

ECO #
ECO # 00904

FCC Testing

This subassembly is marketed to be sold to equipment manufacturers for incorporation into systems. This equipment is not FCC tested. FCC testing is the responsibility of the final equipment manufacturer.

TABLE OF CONTENTS

1. INTRODUCTION	1
1.1 Specifications	2
2. HARDWARE CONFIGURATION	5
2.1 CPU Type (SF1, SF8, SF9, SF14, SF20, SF21, SF23, SF24, SF25, SF26).....	8
2.2 Mono/Color Select (SF2)	9
2.3 Keyboard Enable/Disable (SF3).....	10
2.4 WatchDog Time Delay (SF4)	10
2.5 Video Strapping Options (SF5, SF10).....	10
2.6 RAM/ROM Size (SF6, SF17, SF18)	11
2.7 High Speed Enable (SF7).....	13
2.8 Parallel Port Mode (SF11)	14
2.9 RAM/ROM Power Select (SF12, SF32)	14
2.10 BIOS ROM Size -- Video BIOS/ CDI Ext. Disable (SF13).....	15
2.11 IRQ3 & IRQ4 on SBX (SF15, SF19)	15
2.12 WatchDog Voltage Tolerance Select Option (SF16).....	16
2.13 Video Hardware Disable (SF22)	16
2.14 Flash BIOS Program Enable (SF37).....	16
2.15 PC Interface (J1 and J2).....	17
2.16 SBC Power (J3)	20
2.17 Keyboard/Speaker Interface (J4).....	21
2.17.1 Keyboard	22
2.17.2 Speaker/Annunciator Interface	22
2.18 SBX Connector (J5)	24
2.19 Reset (J6)	24
2.20 IDE Interface (J7).....	25
2.20.1 Select IDE hard disk Type	25
2.20.2 Setting up a Hard Disk for use with MS-DOS.....	25
2.20.3 Setting up a Hard Disk for use with Windows 95.....	26

2.21	Printer Interface (J8)	27
2.22	Floppy Interface (J10).....	28
2.23	Serial Port Interface (J11 and J12).....	28
2.24	Flat Panel Interface (J13).....	29
2.25	External Power (J14).....	30
2.26	VGA Video Interface (J9).....	30
2.27	Fanned Heat Sink Power Connector (J16).....	31
2.28	External Battery (J17).....	31
2.29	WatchDog LED (LE1)	32
2.30	Reset LED (LE2).....	32
2.31	System DRAM Installation (U17, U18).....	32
2.32	BIOS and Semiconductor Devices (U15, U37, U19, U41)	33
3.	PRODUCT OVERVIEW	34
3.1	SBC Memory Map	35
3.2	SBC I/O Map.....	39
3.3	SBC Interrupt Assignments.....	40
3.4	SBC DMA Assignments.....	41
3.5	WatchDog Timer	41
3.6	RAM/ROM Disk Bank Select	42
3.7	Automatic Wait States.....	43
3.8	Real-Time Clock RAM Usage.....	44
3.9	Flash Programming Voltage Port.....	45
3.10	SBX Interrupt Port.....	45
4.	SOFTWARE OPERATION	46
4.1	SBX I/O	46
4.2	SBX Interrupt Service	47
4.3	WatchDog timer	47

4.4	RAM/ROM Disk Bank Select	48
4.5	Enable/Disable VGA Memory	50
4.6	Flash Programming Voltage Enable/Disable	52
4.7	ACC 2178 Registers	53
4.8	Super I/O Options.....	53
4.9	SBC Cache Control	54
4.10	Memory Managers (DOS-Based SBC Systems Only).....	54
4.10.1	CDIMEM.SYS	54
4.10.2	Memory Mappings and Considerations.....	54
4.11	Dither Support for Color TFT Flat Panels Displays.....	55
4.11.1	The Dither TSR.....	55
4.12	FVBIOS	56
5.	BIOS	58
5.1	AT-Compatible Core Characteristics	58
5.2	Fixed Disk Tables.....	59
5.3	Recoverable POST Errors	60
5.4	BIOS Testpoints and Beep Codes.....	61
5.5	BIOS Messages.....	64
5.6	Boot-Up Display Screen	66
5.7	The Main Menu	66
5.7.1	The Menu Bar	67
5.7.2	The Legend Bar.....	67
5.7.3	The Field Help Window	68
5.7.4	The General Help Window.....	68
5.7.5	Main Menu Selections.....	69
5.7.6	IDE Adapters	69
5.7.7	Memory Shadow	72
5.7.8	Boot Sequence	73
5.7.9	Numlock.....	74
5.8	The Advanced Menu	75
5.8.1	Integrated Peripherals.....	75
5.8.2	Menu Selections	76
5.8.3	Advanced Chipset Control.....	76
5.8.4	Menu Selections	77
5.8.5	Large Disk Access Mode	78
5.9	The Exit Menu.....	79

5.9.1	Save Changes & Exit.....	79
5.9.2	Exit Without Saving Changes.....	79
5.9.3	Get Default Values.....	80
5.9.4	Load Previous Values.....	80
5.9.5	Save Current Values.....	80
5.10	BIOS Programmer's Interface	80
5.11	Interrupt 10H Functions - Video Services.....	81
5.11.1	Interrupt 10H Functions - Western Digital Paradise Extensions.....	83
5.11.2	Interrupt 10H Functions - VESA VBE Compliant SVGA Extensions	90
5.11.3	Interrupt 11H Functions - Return System Information.....	97
5.11.4	Interrupt 12H Functions - Return System Memory Size	97
5.11.5	Interrupt 13H Functions - Diskette Services.....	97
5.11.6	Interrupt 13H Functions - Fixed Disk Services.....	101
5.11.7	Interrupt 14H Functions - Serial Services.....	104
5.11.8	Interrupt 15H Functions - Miscellaneous Hardware Services.....	105
5.11.9	Interrupt 16H Functions - Keyboard Services.....	107
5.11.10	Interrupt 17H Functions - Parallel Printer Services.....	108
5.11.11	Interrupt 1AH Functions - Time of Day Services.....	108
5.12	BIOS Data Area.....	109
5.13	Interrupt Vectors	112
6.	SEMICONDUCTOR DISK REFERENCE	114
6.1	Overview.....	114

APPENDICES

ADDITIONAL READING	116
SBC I/O OPTIONS	117
SBX EXPANSION BOARDS	122
MATING CONNECTORS	124
MECHANICAL OUTLINE	125
SOFTWARE EXAMPLE - SBX IMPLEMENTATION.....	126
SOFTWARE EXAMPLE - SRAM IMPLEMENTATION.....	135
WARRANTY.....	142

1. INTRODUCTION

The SBC-486DX/SBC-586 (referred to as the SBC for the remainder of this manual) is a complete PC/AT computer on a single circuit board. The SBC is 100% PC/AT compatible, and provides all the functionality of a PC motherboard plus disk drive, local-bus video, real-time clock, and communications boards. Smaller than a 5.25" disk drive, the SBC is easily embedded within your application.

All functions of the SBC are exactly the same as on a standard PC/AT. We do not risk the board's compatibility by using non-standard COM ports, DMA, interrupts, or counters and then try to make it look like a PC in software. The SBC uses an off-the-shelf BIOS and offers a 80486SX2/DX2/DX4/5x86 CPU running at a 25 or 33 Mhz external bus speed. (The 80486 DX2, DX4, SX2, and Cyrix 5x86 run at faster internal clock speeds.) The SBC will run thousands of programs that run on a similarly equipped PC/AT, even with hardware security keys.

The compact board (5.75" x 7.75" x 1.50") draws less than 10 watts from a single 5V supply.

Sockets for on-board RAM/ROM disk are standard. This lets you build a ROM-based controller for your target system while you develop your program under DOS. Each of the three device sockets supports a 512k battery-backed RAM disk or 1M ROM disk. When the rotating media is replaced with the on-board ROM/RAM disk, the SBC will withstand harsh, high-vibration environments and industrial temperatures.

The SBC includes many functions that you might otherwise have to purchase and install separately. The integrated video/LCD/EL controller is fully PC/AT VGA compatible and uses PC-compatible VGA color monitors. The SBC can also directly drive liquid crystal (LCD) and electroluminescent (EL) displays.

The SBC includes a battery-backed real-time clock so time stamping is always correct. Additionally, a WatchDog timer brings the CPU back on track should it ever get lost.

The SBC offers an SBX connector that lets you snap on any single wide Intel SBX-compatible daughterboard. Hundreds of different modules are available from many vendors, including Computer Dynamics. Each cost-effective module has one or two functions and is designed and tested for rugged industrial environments. Typical modules provide parallel I/O, analog I/O, serial I/O, and network functions.

For PC specific I/O, a simple ribbon cable lets you add any PC/AT-compatible expansion board. Or, you can add up to six PC/AT boards using a passive motherboard. This lets you connect the SBC to any board in the PC/AT world, including networks.

Features:

- * One SBX expansion socket
- * Licensed BIOS
- * 80486SX2, DX2, DX4 or 5x86 CPU
- * LCD/EL display controller with super VGA local bus video
- * PC/AT keyboard port
- * Battery-backed real-time clock
- * Two 3.5"/5.25" floppy drive control
- * Two PC-compatible serial ports
- * +5V only power input (external +12V and -12V required for SBX and PC Bus)
- * IDE/AT hard drive control
- * Parallel printer port with read back
- * RAM/ROM disk to 3M
- * PC/AT Bus expansion
- * Up to 64Mbytes DRAM
- * Mounts on a 5.25" disk drive

1.1 Specifications

Compatibility	100% IBM PC/AT hardware, BIOS/software compatible
CPU	100% compatible with: Intel 486SX2-50, 486DX2-66, 486DX4-100 Texas Instrument 486DX2-66, 486DX4-100 Cyrix 5x86-100 IBM 5x86-100
BIOS	100% compatible clean room BIOS, Flash compatible
DRAM Memory	Incremental steps ranging between 1M to 64Mbytes without parity and 0/1 wait states
RAM/ROM Disk	Three 32k-1Mbyte sockets; bank selected
DMA, CTC, INT	8237 DMA, 8254 CTC, 8259 interrupt, 8284 clock and 8288 bus controller standard cells embedded into VLSI silicon, guaranteeing hardware compatibility
Operator Interface	PC/AT keyboard interface; speaker interface
Video/Display	VGA compatible with analog Super VGA color or monochrome video. Liquid crystal (LCD) and electroluminescent (EL) displays can be directly connected. Software initialization drivers included.
Communications	Two RS-232 ports, XT pinout (8250 standard cell) including all PC handshaking & true RS-232 levels; one parallel printer port (with read back); speaker connection
Disk Storage	Floppy disk interface (two 3.5/5.25" 250/500 kbps); IDE/AT hard drive interface (emulates a Western Digital Winchester interface board); RAM/ROM disk (see memory specification) ROM-Disk generation software optionally available to run both DOS and your programs without rotating disk drives
Additional	Battery-backed real-time clock with drivers; WatchDog timer with software enable
Parallel I/O	Printer port useable as 8 outputs, 4 inputs, and 5 inputs/outputs; IDE hard drive interface useable as 16 inputs
SBX Expansion	One SBX socket with interrupts (Intel SBX standard)
PC Expansion	64-pin header and 32-pin header connector to connect up to 6 PC/AT boards via short ribbon cable and/or passive backplane
Physical	5.75" x 7.75" x 1.50" w/ four mounting holes; size and mounting slots match 5.25" disk drive; operating ranges: 0-70° C, 5-95% RH (non-condensing), 0-10,000 feet
Power	5V-only, typical

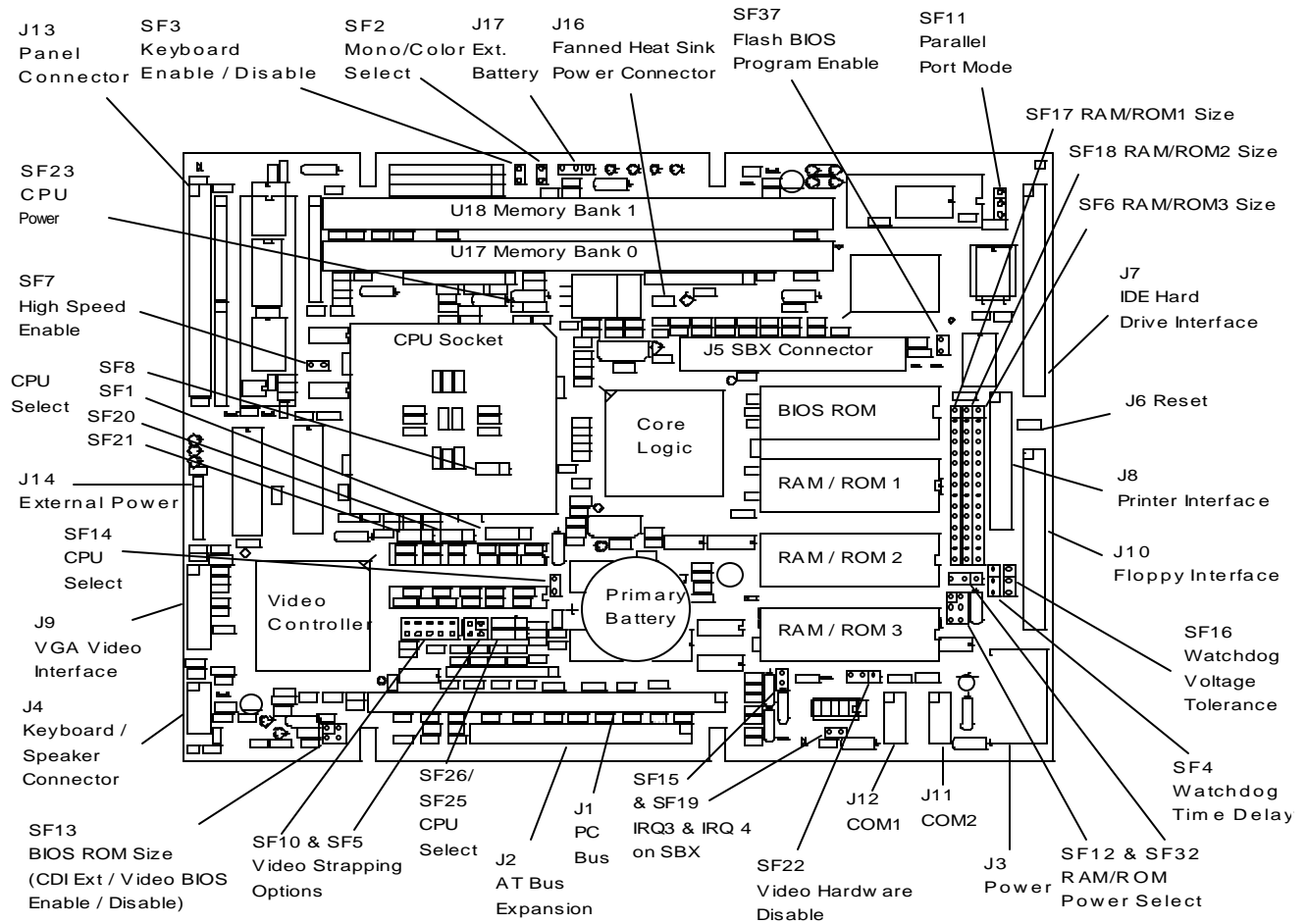
Power consumption for selected CPU speeds and RAM capacities.				
	4 Mb	8 Mb	16 Mb	32 Mb
SX/DX-25	1.40A	1.40A	1.40A	1.40A
SX2-50	1.40A	1.40A	1.40A	1.40A
DX2-50	1.65A	1.65A	1.65A	1.65A
SX/DX-33	1.60A	1.60A	1.60A	1.60A
DX2-66	1.92A	1.92A	1.92A	1.92A
DX4-100	1.92A	1.92A	1.92A	1.92A

	Maximum Ambient Operating Temperature			
	No Heat Sink	Still Air with Passive Heat Sink	Moving Air with Passive Heat Sink**	Fanned Heat Sink
SX/DX-25	31° C	58° C	64° C	70° C
DX/2-50	15° C	51° C	58° C	70° C
SX/DX-33	8° C	46° C	50° C	70° C
DX/2-66	-5° C	33° C	41° C	70° C

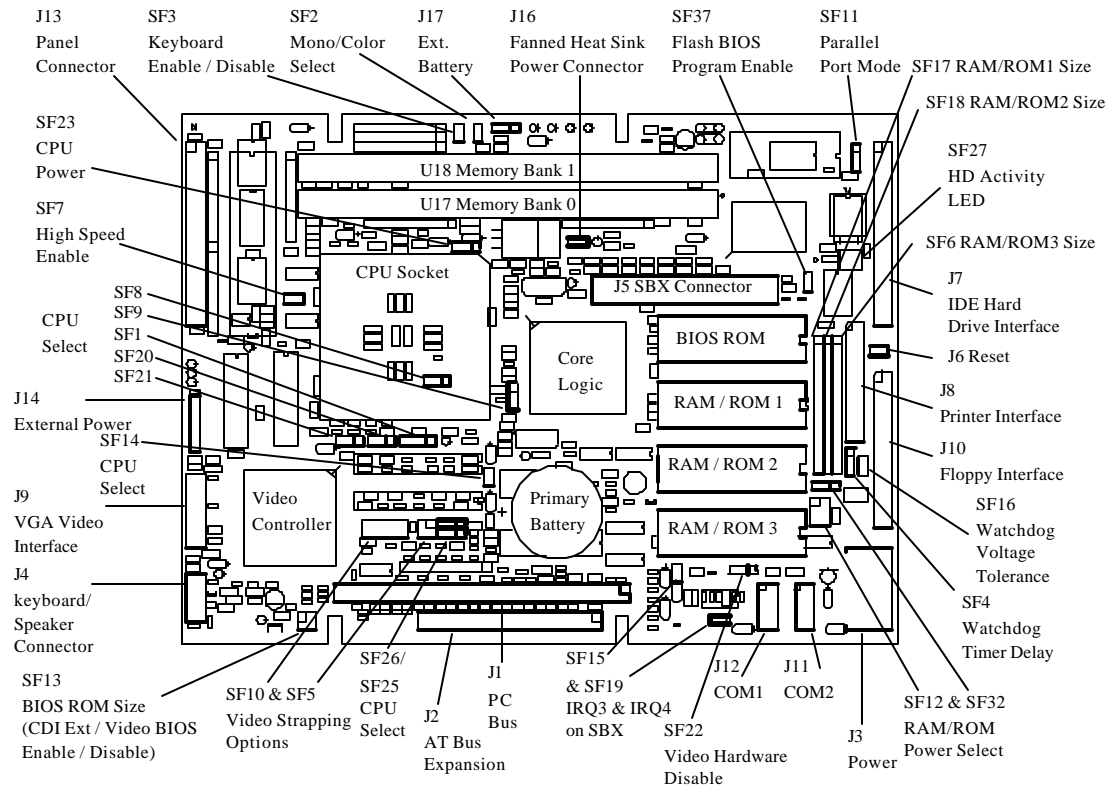
** Maximum ambient temperature is based on airflow at 200 ft/min.

2. HARDWARE CONFIGURATION

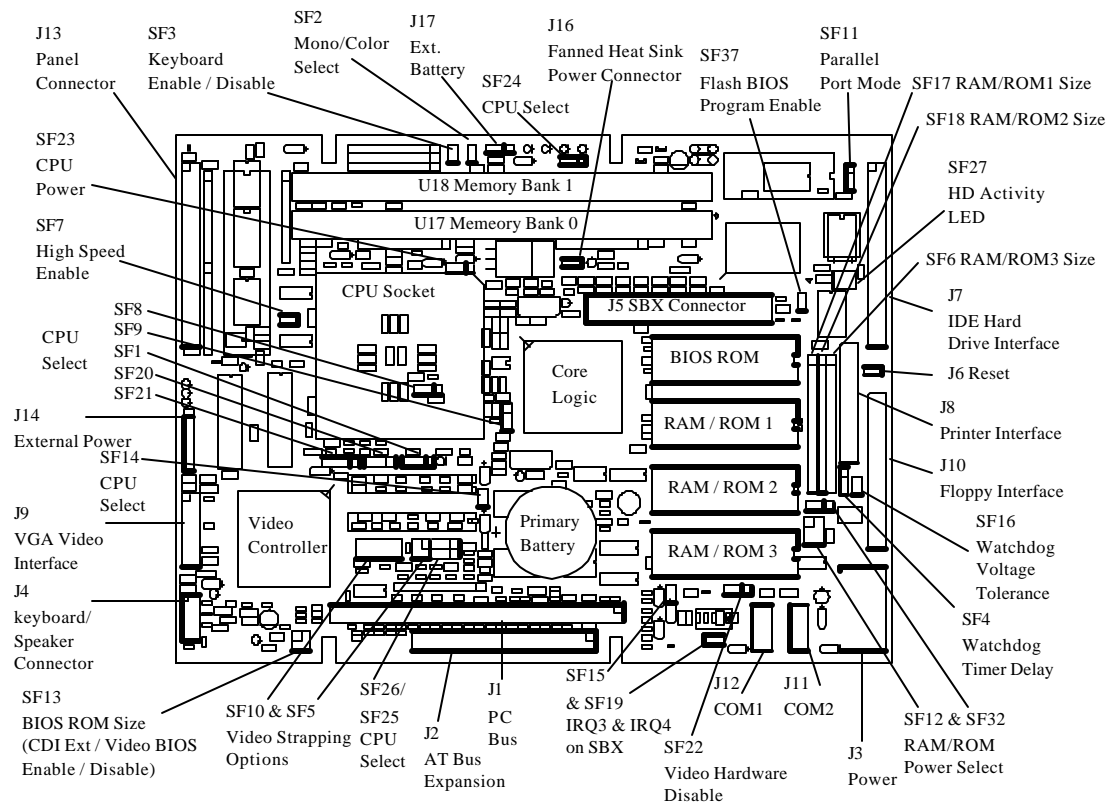
Before using your SBC you will want to attach your monitor, keyboard, speaker, floppy, IDE drive, printer, and RS-232 devices. The following drawing shows the location of the RS-232 ports (COM1 and COM2), printer port (LPT1), video connector, parallel connector, keyboard/speaker connector, floppy connector, and Hard Drive connector.



User Configuration for Rev M

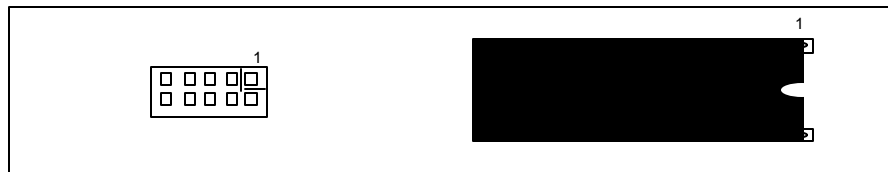


User Configuration for Rev N



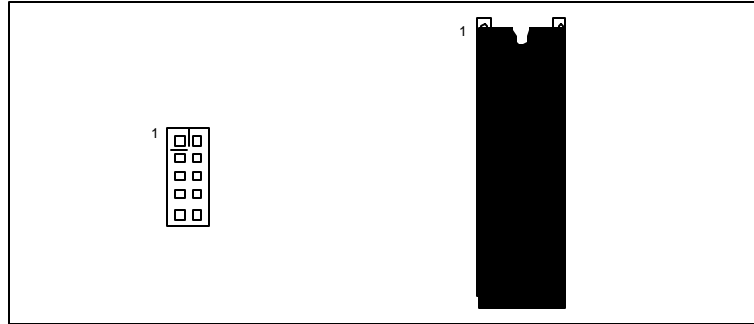
User Configuration for Rev P

You will also need to set or confirm the setting of a few strapping fields. If the connector or I.C. is horizontal, pin 1 is in the top right corner.



Horizontal Pin 1 Alignment

If the connector or I.C. is vertical, pin 1 is in the top left corner.



Vertical Pin 1 Alignment

2.1 CPU Type (SF1, SF8, SF9, SF14, SF20, SF21, SF23, SF24, SF25, SF26)

The SBC is flexible enough to support many different CPU manufacturers' such as Intel, Texas Instrument, Cyrix, IBM and AMD. The SBC contains several stepping fields which must be configured for proper operation. Your SBC has been properly preset at Computer Dynamics for the CPU which you've purchased.

Note:

If you desire to upgrade your present CPU, please ensure to follow the CPU Type Selection Guide very carefully or damage may occur to either your SBC, the CPU or both.

CPU Type Selection Guide for Rev M

CPU Type	SF1	SF8	SF14	SF20	SF21	SF23	SF25	SF26
Intel SX-25	2-3	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel SX2-50	1-2/3-4	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel DX-33	1-2/3-4	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel DX2-66	1-2/3-4	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel DX4-100	1-2/3-4	1-2	Open	1-2	Open	2-3	2-3	2-3
TI DX2-66 ₃	1-2/3-4	1-2	Open	2-3	2-3	2-3	2-3 ₁	2-3 ₁
TI DX4-100 ₃	1-2/3-4	1-2	Open	2-3	2-3	2-3	2-3 ₁	2-3 ₁
Cyrix 5x86-100 ₃	1-2/3-4	1-2	1-2	1-2	1-2	2-3	1-2	1-2
SGS DX4-100 ₃	1-2/3-4	1-2	1-2	1-2	1-2	2-3 ₂	2-3 ₁	2-3 ₁

Note 1 For write-back mode, strap 1-2. Also requires the BIOS to include write-back option.

Note 2 For 5V CPU (**ST486DX-10HS**), strap 1-2. For 3.45V CPU (**ST486DXV10HS**), strap 2-3.

Note 3 Requires SMT resistor change on PCB. Contact your Applications Engineer for details.

CPU Type Selection Guide for Rev N

CPU Type	SF1	SF8	SF9	SF14	SF20	SF21	SF23	SF25	SF26
Intel SX-25	2-3	1-2	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel SX2-50	1-2/3-4	1-2	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel DX-33	1-2/3-4	1-2	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel DX2-66	1-2/3-4	1-2	1-2	Open	1-2	Open	1-2	2-3	2-3
Intel DX4-100	1-2/3-4	1-2	1-2	Open	1-2	Open	2-3	2-3	2-3
TI DX2-66	1-2/3-4	1-2	2-3	Open	2-3	2-3	2-3	2-3 ₁	2-3 ₁
TI DX4-100	1-2/3-4	1-2	2-3	Open	2-3	2-3	2-3	2-3 ₁	2-3 ₁
Cyrix 5x86-100	1-2/3-4	1-2	1-2	1-2	1-2	1-2	2-3	1-2	1-2
SGS DX4-100	1-2/3-4	1-2	1-2	1-2	1-2	1-2	2-3 ₂	2-3 ₁	2-3 ₁

Note 1 For write-back mode, strap 1-2. Also requires the BIOS to include write-back option.

Note 2 For 5V CPU (**ST486DX-10HS**), strap 1-2. For 3.45V CPU (**ST486DXV10HS**), strap 2-3.

CPU Type Selection Guide for Rev P

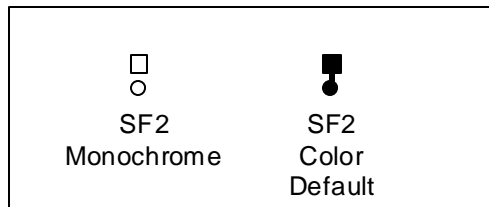
CPU Type	SF1	SF8	SF9	SF14	SF20	SF21	SF23	SF24	SF25	SF26
Intel SX-25	2-3	1-2	1-2	Open	1-2	Open	1-2	2-3	2-3	2-3
Intel SX2-50	1-2/3-4	1-2	1-2	Open	1-2	Open	1-2	2-3	2-3	2-3
Intel DX-33	1-2/3-4	1-2	1-2	Open	1-2	Open	1-2	2-3	2-3	2-3
Intel DX2-66	1-2/3-4	1-2	1-2	Open	1-2	Open	1-2	1-2	2-3	2-3
Intel DX4-100	1-2/3-4	1-2	1-2	Open	1-2	Open	2-3	1-2	2-3	2-3
TI DX2-66	1-2/3-4	1-2	2-3	Open	2-3	2-3	2-3	1-2	2-3 ₁	2-3 ₁
TI DX4-100	1-2/3-4	1-2	2-3	Open	2-3	2-3	2-3	1-2	2-3 ₁	2-3 ₁
Cyrix 5x86-100	1-2/3-4	1-2	1-2	1-2	1-2	1-2	2-3	1-2	1-2	1-2
AMD DX2-66	1-2/3-4	1-2	1-2	1-2	1-2/3-4	1-2/3-4	2-3	1-2	1-2	1-2
AMD DX4-100	1-2/3-4	1-2	1-2	1-2	1-2/3-4	1-2	2-3	1-2	1-2	1-2
SGS DX4-100	1-2/3-4	1-2	1-2	1-2	1-2	1-2	2-3 ₂	1-2	2-3 ₁	2-3 ₁

Note 1 For write-back mode, strap 1-2. Also requires the BIOS to include write-back option.

Note 2 For 5V CPU (**ST486DX-10HS**), strap 1-2. For 3.45V CPU (**ST486DXV10HS**), strap 2-3.

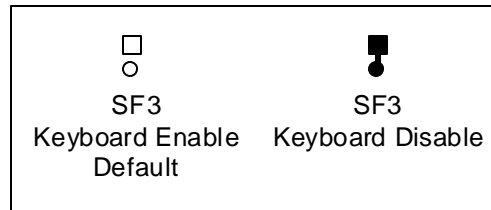
2.2 Mono/Color Select (SF2)

Select monochrome or color for the primary adapter. Color adapter operation is selected by installing a strap in SF2 as shown. This selection is duplicated in the battery-backed CMOS configuration RAM.



2.3 Keyboard Enable/Disable (SF3)

The keyboard function is enabled or disabled by installing a strap as shown. When the keyboard is disabled, the SBC will boot and execute your program. Keyboard input will not be allowed.

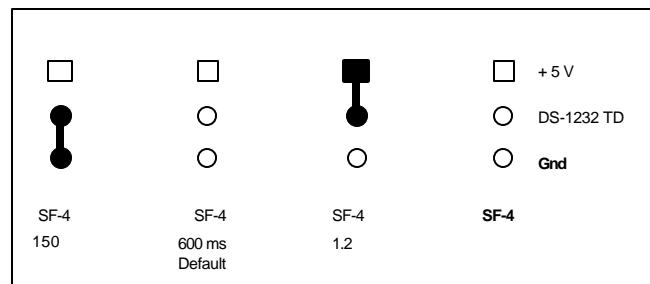


SF3: Keyboard Enable/Disable

2.4 WatchDog Time Delay (SF4)

The WatchDog timer verifies that the SBC is continuing to execute your program. Unless you will be using a WatchDog refresh period other than the factory default, there is no need to modify SF4. Note that the WatchDog timer is not activated until strobed as discussed in section 4.3.

The WatchDog timer period can be set for 150 ms (250 ms maximum) by strapping SF-4 from 2-3. The period can be set for 600 ms (1s maximum) by not installing any strap on SF4. A 1.2s (2s maximum) can be selected by strapping SF4 from 1-2. The factory default is no strap for a 600 ms period.



Watch-dog Time Delay Select

Note:

This strapping field is only applicable when the WatchDog Timer option is requested from CDI.

2.5 Video Strapping Options (SF5, SF10)

Strapping fields SF5 and SF10 provide power-on default options for video. These straps will be set to the default value at the factory based on the flat panel configuration ordered.

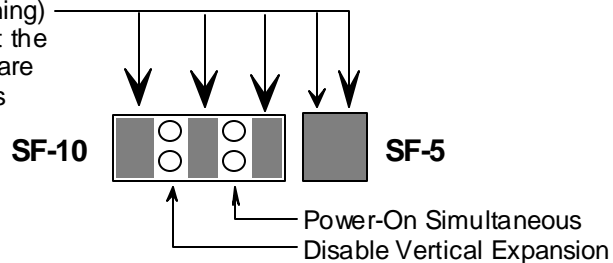
Note:

The default strapping for the SBC single board computer that will drive a monitor is with SF5 and SF10 not-strapped. If the SBC is part of a Display-Pac or if it is pre-configured for a specific panel, the default strapping from the factory may be different.

Note:

Maintaining an up to date chart for video strapping in this manual is not possible due to the constant addition of new flat panel displays supported by the SBC product line. The straps in SF5 and SF10 should have been pre-configured for the flat panel in your Display-Pac. If you are experiencing problems with your flat panel, or if you are implementing a panel and require additional information contact your Computer Dynamics Sales Engineer with the flat panel specifics. They will provide you with the appropriate information.

Panel Specific straps (shown here with hatching) will be configured per the customers order at the factory. If additional assistance or changes are needed, call your Computer Dynamics Sales Engineer for Assistance.



NOTE: The two straps show above shows functionality only. The state of the strap (i.e. whether you insert or remove the strap to achieve the Power-On Simultaneous mode) is panel specific. By default, boards are shipped from the factory configured for NON-Simultaneous operation.

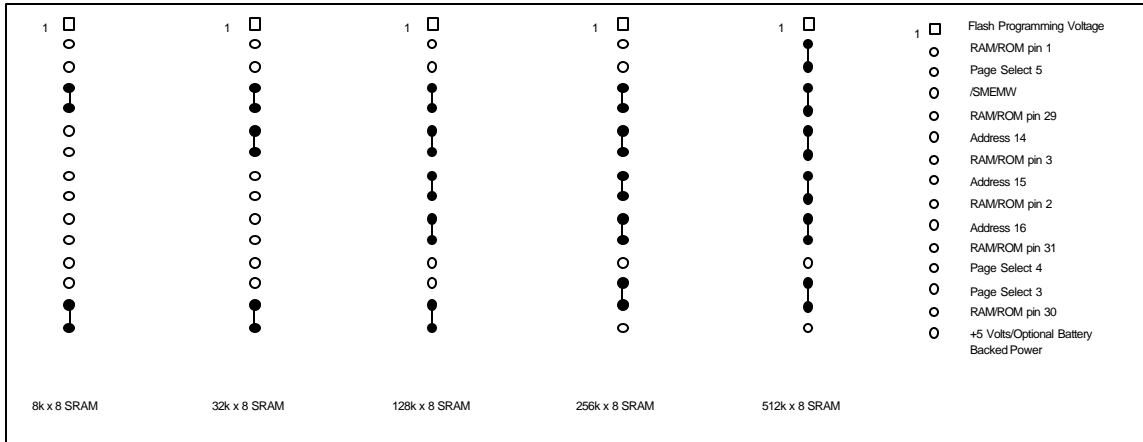
NOTE: The default Vertical Expansion depends mainly on the size of the panel. Normally:
 Panel Height < 480 Lines - Disable Expansion
 Panel Height => 480 Lines - Enable Expansion

SF5, SF10: Video Strapping Options

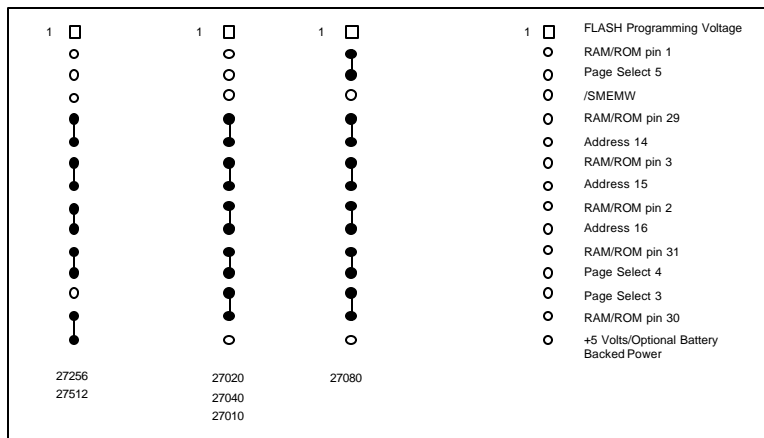
2.6 RAM/ROM Size (SF6, SF17, SF18)

Set strapping fields SF17, SF18, and SF6 for the RAM or ROM type as shown in the following figures. See section 4.4 for more information regarding software access of the RAM/ROM devices. The factory default is no strap. Be sure to set SF12 and SF32 as required for battery-backup on the RAM/ROM devices.

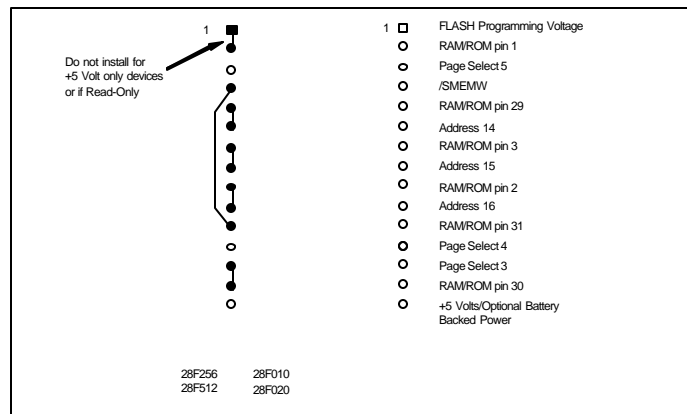
Strap	RAM/ROM Device
SF17	RAM/ROM1
SF18	RAM/ROM2
SF6	RAM/ROM3



SRAM Selections

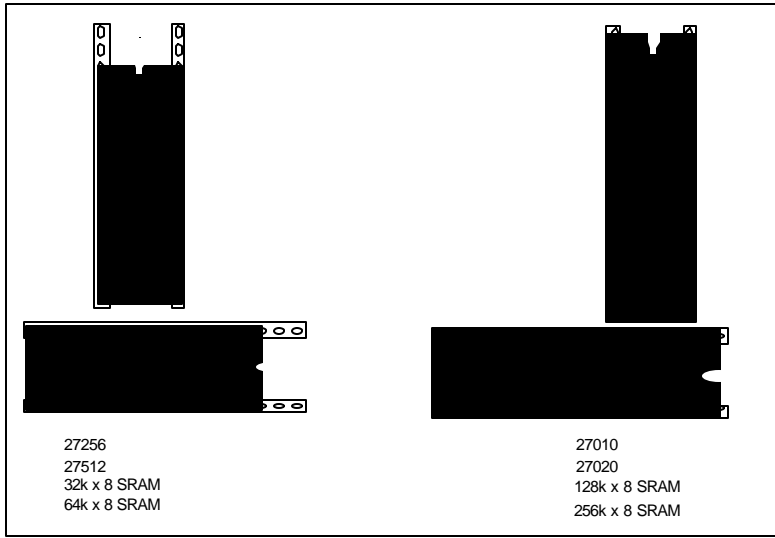


EPROM Selections



FLASH Selections

Install the devices in the sockets as shown in the following figure.



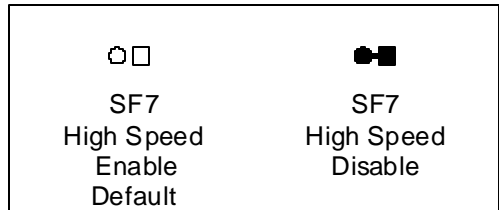
SRAM/EPROM Alignment

2.7 High Speed Enable (SF7)

Before power-up, SF7 determines whether the SBC operates at <8 MHz or the speed you purchased (25 MHz, 33 MHz, or 50 MHz external bus speed). The SBC will operate at low speed with SF7 strapped, and at high speed with SF7 not strapped.

Note:

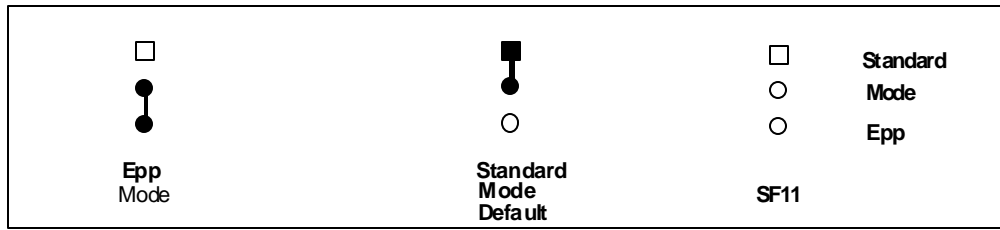
Clock multiplied processors (DX2 and DX4) do not allow speed switching during operation. You can only select your speed option before power-up.



SF7: High Speed Enable

2.8 Parallel Port Mode (SF11)

Strap 2-3 on SF11 to select Enhanced Parallel Port (EPP) mode. Strap 1-2 to select standard parallel port mode.



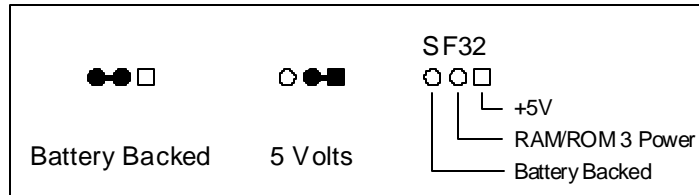
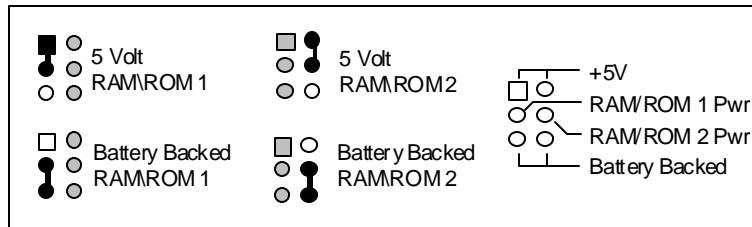
The Parallel Port is default to DMA Channel 6 when shipped from CDI. Alternate DMA Channels are DMA 3, 5 and 7. Call your CDI Sales Engineer on how to change DMA Channels if needed.

2.9 RAM/ROM Power Select (SF12, SF32)

Each RAM/ROM power source may be individually selected from battery backup or from +5V using SF31, SF32 or SF33.

Strapping Field	Controls power to
SF12 (Pins 1,3,5)	RAM/ROM1
SF12 (Pins 2,4,6)	RAM/ROM2
SF32	RAM/ROM3

RAM/ROM Power Select

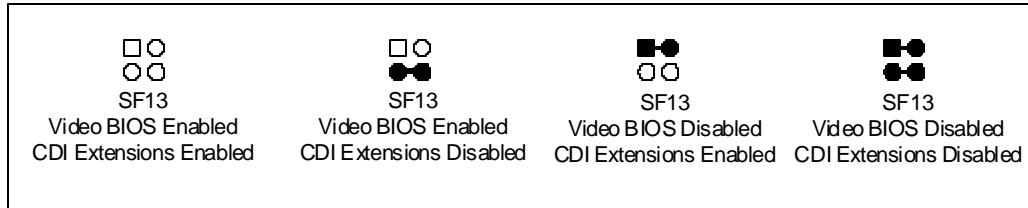


SF12: RAM/ROM 1 and 2 Power Select

SF32: RAM/ROM 3 Power Select

2.10 BIOS ROM Size -- Video BIOS/ CDI Ext. Disable (SF13)

The BIOS EPROM Map is selected using strapping field SF13 as shown in the following figure. This strapping field adjusts the memory range for the BIOS EPROM chip select as discussed in section 3.1. By adjusting the BIOS ROM Map the Video BIOS and CDI Extensions can be selectively disabled if needed. The default strapping is for CDI Extensions to be disabled and for the Video BIOS to be enabled.



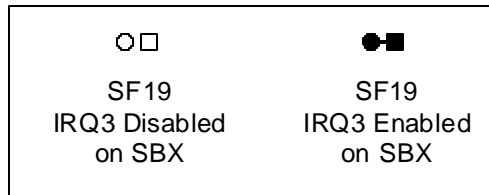
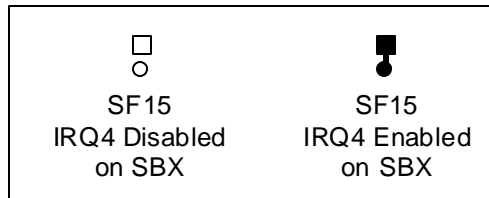
SF13: BIOS EPROM Map Select

Note:

On-board video must be disabled to use a standard PC BUS expansion Video card. (MDA Adapters may reside in the BUS without disabling on-board video) In addition to the Video BIOS Disable strap in SF13, the video hardware must also be disable via SF22. Refer to the SF22 strapping diagram for more details.

2.11 IRQ3 & IRQ4 on SBX (SF15, SF19)

Interrupts for COM1 and COM2 are provided to the SBX by strapping SF-15 (COM1) and/or SF-19 (COM2). These strapping fields are not installed at the factory. The SBC is shipped with these straps not installed. Thus, the default for SF-15 and SF-19 is not strapped.



SF15, SF19: COM3 and COM4 on SBX

Note:

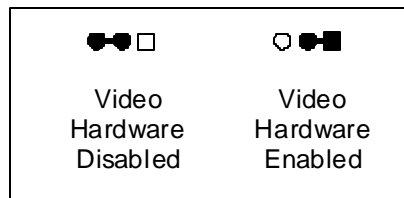
COM3 /COM4 Compatibility strapping requires a special PAL on the SBC board. This PAL is socketed. Request one from your CDI Sales Engineer.

2.12 WatchDog Voltage Tolerance Select Option (SF16)

The WatchDog Select option is hard-wired on the SBC for monitoring out-of-tolerance voltage only. Should your application require additional microprocessor supervision please consult your Computer Dynamics Sales engineer.

2.13 Video Hardware Disable (SF22)

The on-board video hardware can be disabled by the use of SF22.



SF22: Video Hardware Disable

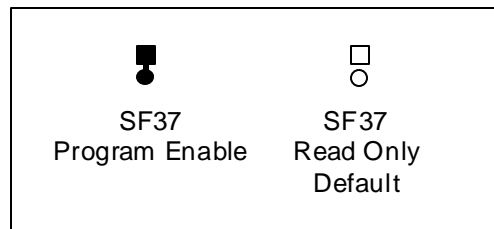
Note:

On-board video must be disabled to use any standard PC BUS expansion Video card (MDA Adapters may reside in the BUS without disabling on-board video). In addition to the Video Hardware Disable strap SF22 above, the video BIOS must also be disabled via SF13. Refer to the SF13 strapping diagram for more details.

2.14 Flash BIOS Program Enable (SF37)

Flash BIOS on the SBC can be erased and reprogrammed by setting SF37 as shown in the following figures. While in the Program Enable mode, +12 Volts can be applied by outputting to port 550H (see section 3.9). The Flash BIOS may be both programmed and read (program execution) with +12 Volts applied. In the Program Enable mode, Flash memories consume more power than in the read-only mode.

Note that most flash memories require erasure of the entire device before re-programming. If the BIOS is erased and not completely reprogrammed, the SBC will not operate.



SF37: FLASH BIOS Program Enable

2.15 PC Interface (J1 and J2)

PC Bus-compatible boards can be attached to the SBC using the PC Interface on J1 and J2. J1 is PC/XT compatible. J2 adds the additional lines for PC/AT compatibility. A single PC/XT Bus board can be attached using a flat cable as shown in the tables on the following pages. Up to two PC/XT or PC/AT boards may be connected to the SBC using the MB-AT-2.8.

Note:

-5 Volts is not supplied by the SBC. If your expansion board requires -5 Volts you must supply it or use the MB-AT-2.8, MB-AT-4.8 or MB-AT-6.8. The MB-AT series of motherboards generates -5 Volts from the -12 Volt supply.

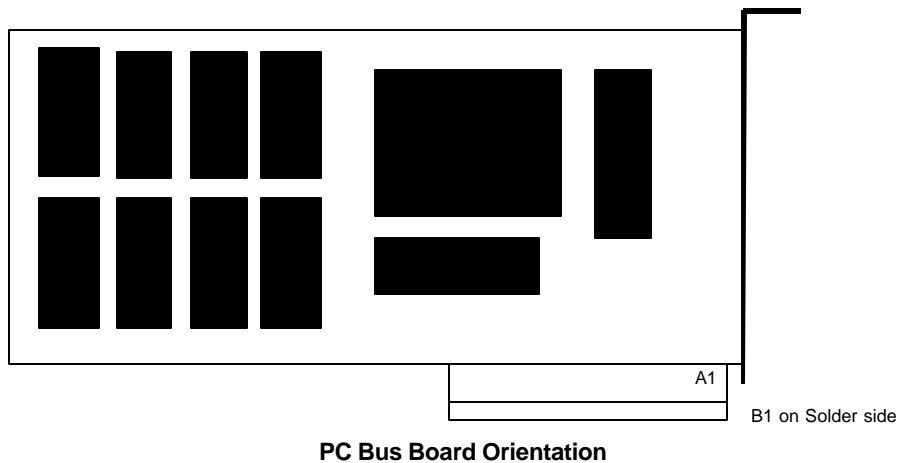
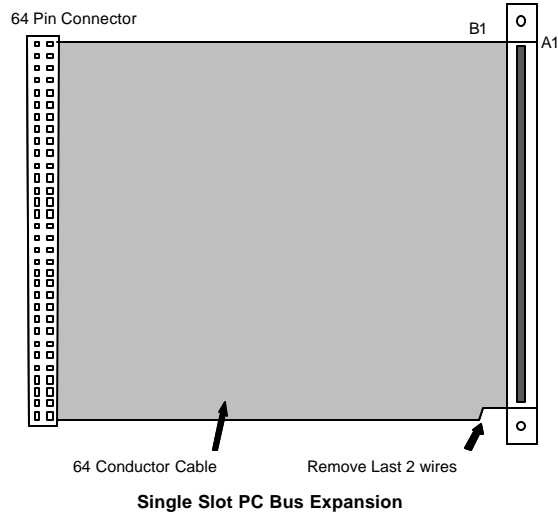
Signal	XT Bus Pin	J1 Pin	J1 Pin	XT Bus Pin	Signal
/IO Channel Check	A1	1	2	B1	Ground
SDATA 07	A2	3	4	B2	Reset Driver
SDATA 06	A3	5	6	B3	+5 Volts
SDATA 05	A4	7	8	B4	IRQ 2
SDATA 04	A5	9	10	B5	
SDATA 03	A6	11	12	B6	DRQ 2
SDATA 02	A7	13	14	B7	-12 Volts
SDATA 01	A8	15	16	B8	/OWS
SDATA 00	A9	17	18	B9	+12V
IO Channel Ready	A10	19	20	B10	Ground
AEN	A11	21	22	B11	/MEMW
SADDRESS 19	A12	23	24	B12	/MEMR
SADDRESS 18	A13	25	26	B13	/IOW
SADDRESS 17	A14	27	28	B14	/IOR
SADDRESS 16	A15	29	30	B15	/DACK 3
SADDRESS 15	A16	31	32	B16	DRQ 3
SADDRESS 14	A17	33	34	B17	/DACK 1
SADDRESS 13	A18	35	36	B18	DRQ 1
SADDRESS 12	A19	37	38	B19	/DACK 0
SADDRESS 11	A20	39	40	B20	CLOCK
SADDRESS 10	A21	41	42	B21	IRQ 7
SADDRESS 09	A22	43	44	B22	IRQ 6
SADDRESS 08	A23	45	46	B23	IRQ 5
SADDRESS 07	A24	47	48	B24	IRQ 4
SADDRESS 06	A25	49	50	B25	IRQ 3
SADDRESS 05	A26	51	52	B26	/DACK 2
SADDRESS 04	A27	53	54	B27	T/C
SADDRESS 03	A28	55	56	B28	ALE
SADDRESS 02	A29	57	58	B29	+5 Volts
SADDRESS 01	A30	59	60	B30	14.3818 MHz
SADDRESS 00	A31	61	62	B31	Ground
+5 Volts	No Pin	63	64	No Pin	Ground

PC/XT Bus Connections (J1)

Signal	AT Bus Pin	J2 Pin	J2 Pin	AT Bus Pin	Signal
SBHE	C1	1	2	D1	/MEM CS 16
LA 23	C2	3	4	D2	/IO CS 16
LA 22	C3	5	6	D3	IRQ 10
LA 21	C4	7	8	D4	IRQ 11
LA 20	C5	9	10	D5	IRQ 12
LA 19	C6	11	12	D6	IRQ 15
LA 18	C7	13	14	D7	IRQ 14
LA 17	C8	15	16	D8	/DACK 0
/MEMR	C9	17	18	D9	DRQ 0
/MEMW	C10	19	20	D10	/DACK 5
SDATA 08	C11	21	22	D11	DRQ 5
SDATA 09	C12	23	24	D12	/DACK 6
SDATA 10	C13	25	26	D13	DRQ 6
SDATA 11	C14	27	28	D14	/DACK 7
SDATA 12	C15	29	30	D15	DRQ 7
SDATA 13	C16	31	32	D16	+5 Volts
SDATA 14	C17	33	34	D17	/MASTER
SDATA 15	C18	35	36	D18	Ground
No Connection	No Pin	37	38	No Pin	+5 Volts
No Connection	No Pin	39	40	No Pin	Ground

PC/AT Bus Connections (J2)

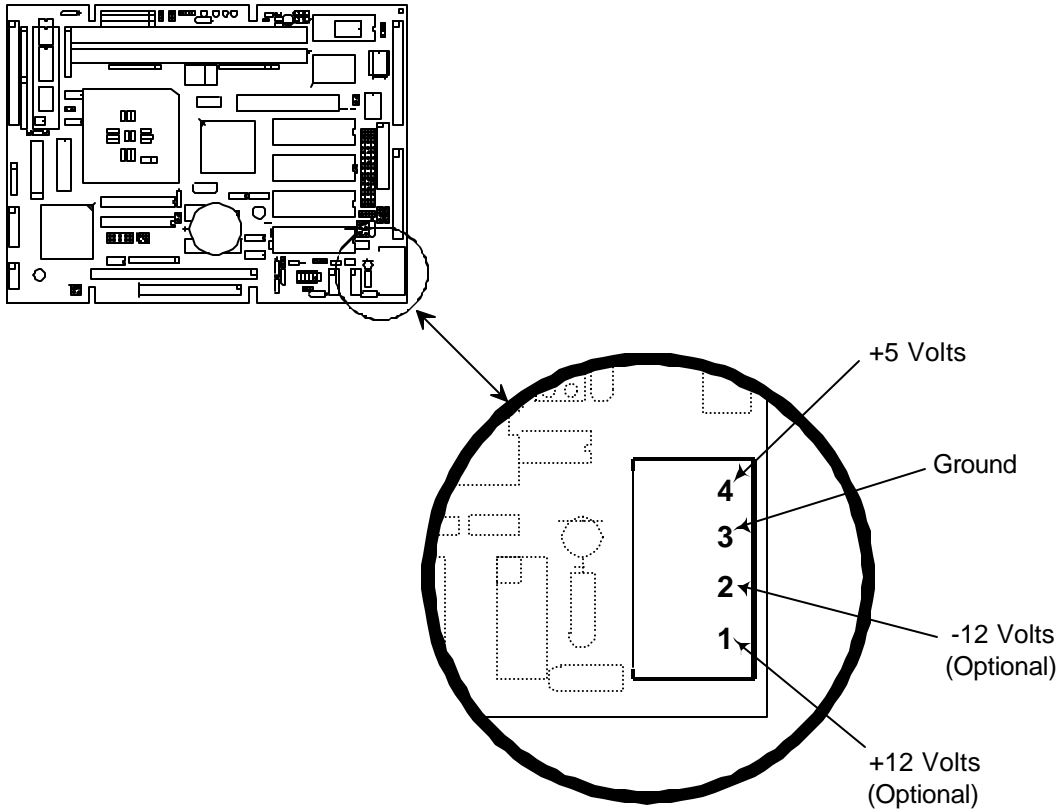
When using the single XT board interface, it may be necessary to add a noise filter to the RESET signal. Noise is induced into RESET from the PC data bus bits 6 and 7. Solder a 0.01 μ F capacitor from RESET (flat cable wire 4) to ground (flat cable wire 2) on the expansion board end of the cable. Or, install a similar capacitor on your expansion board.



Some PC Bus boards are sensitive to ringing and cross talk in the flat cable. While external video boards perform well on a foot or more of cable, network boards have problems with cables longer than six inches. Keep the flat cable as short as possible. Longer cables can be made using a ground plane cable such as 3M 3476/64.

2.16 SBC Power (J3)

Power is applied to the SBC through the connector in the lower right corner of the board as shown in the following figure. Only +5V and ground are required for the SBC. If the SBX connector or the PC Bus is used, +12V and -12V should be supplied through the power connector as shown.

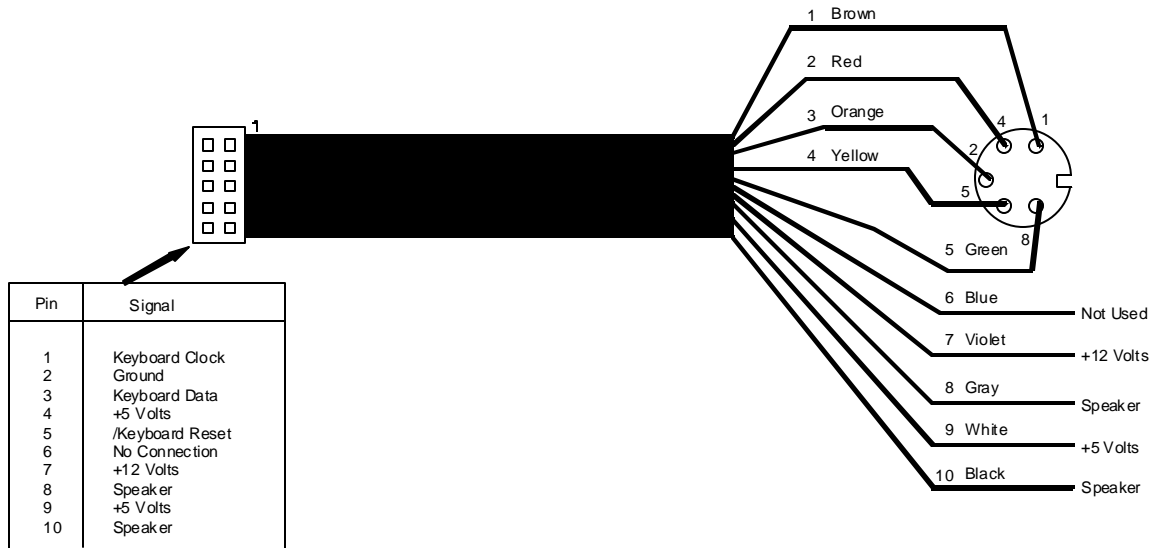


J3: SBC Power Connections

This connector is very similar to the power connector on 5.25" floppy drives. Floppy drives do not use -12V. You will find an additional ground on the SBC's -12V pin. This -12V is used to drive the PC Bus and the SBX connector and is not used to drive any circuitry in the SBC. In order to use a floppy drive power connector, you will need to cut the ground that corresponds to -12V and apply -12V through the floppy drive connector pin or through the PC Bus expansion connector.

2.17 Keyboard/Speaker Interface (J4)

The keyboard/speaker connector provides an interface to PC/AT- and PC/XT-compatible keyboards and a speaker or annunciator.



Keyboard Cable

WARNING

Inserting the keyboard/speaker cable in the serial port connector or a serial port cable in the keyboard/speaker connector may damage either your keyboard or your SBC.

2.17.1 Keyboard

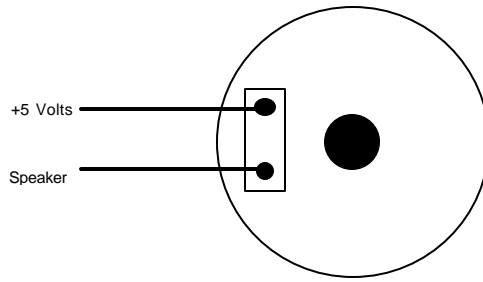
The DIN connector on the end of the cable is compatible with PC/AT and PC/XT type keyboards.

Pin	Signal
1	Keyboard Clock
2	Keyboard Data
3	/Keyboard Reset
4	Ground
5	+5 Volts

PC Keyboard Pin Assignments

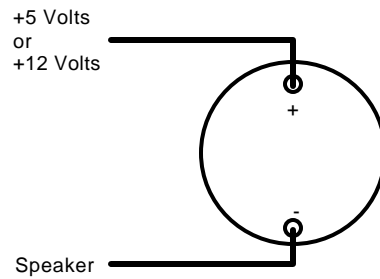
2.17.2 Speaker/Annunciator Interface

A driver transistor is provided to activate a speaker or annunciator. Attach an 8Ω speaker (2" to 2.5" are typically used) to the keyboard cable as shown.



Speaker Interface

Annunciators may be connected to either the 5V supply or the 12V supply depending on the requirements of the particular device.



Typical Annunciator Connector

2.18 SBX Connector (J5)

The SBC provides a connector for boards compatible with the 8-bit SBX specification. The I/O capacity of the SBC can be increased by using this expansion. SBX modules are available from several vendors, including Computer Dynamics. The signals provided in this interface are shown in the following table.

Description	Mnemonic	Pin	Pin	Mnemonic	Description
+12 Volts	+12V	1	2	-12V	-12 Volts
Ground	GND	3	4	+5V	+5 Volts
Reset	RESET	5	6	MCLK	System Clock
SADDRESS 2	MA2	7	8	/MPST	Not Used
SADDRESS 1	MA1	9	10		
SADDRESS 0	MA0	11	12	MINTR1	Interrupt 0
I/O Write	/IOWR	13	14	MINTR0	Interrupt 1
I/O Read	/IORD	15	16	/MWAIT	Module Wait
Ground	GND	17	18	+5V	+5 Volts
SDATA 7	MD7	19	20	/MCS1	Module Select 1
SDATA 6	MD6	21	22	/MCS0	Module Select 0
SDATA 5	MD5	23	24		
SDATA 4	MD4	25	26	TDMA	Not Used
SDATA 3	MD3	27	28	OPT1	Not Used
SDATA 2	MD2	29	30	OPT0	Not Used
SDATA 1	MD1	31	32	/MDACK	Not Used
SDATA 0	MD0	33	34	/MDRQT	Not Used
Ground	GND	35	36	+5V	+5 Volts

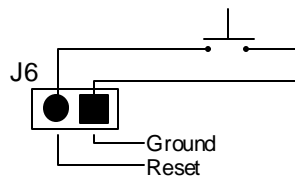
SBX Connections

Note:

If the SBX board you are using requires + 12 volts or -12 volts, you must supply +12 volts and -12 volts through the power connector as shown in section 2.18.

2.19 Reset (J6)

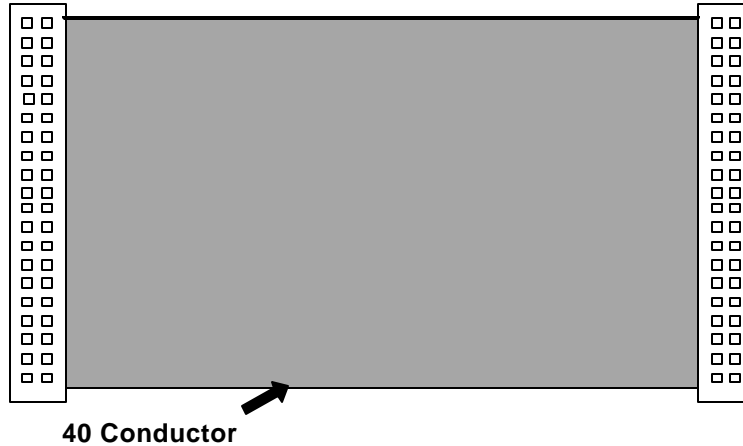
A push button reset may be added to the SBC at J6. The signals at J6 are shown in the following figure.



J6: Push button Reset

2.20 IDE Interface (J7)

The SBC supports up to two AT Interface IDE hard drives, such as Conner's model CFS210A. A single cable is used to connect the drives to the SBC as shown in the following figure.



Winchester Cable

To install two IDE hard disk drives, follow the instructions in the drive manufacturer's manual. Configure the first drive as the Master and the second drive as the Slave.

2.20.1 Select IDE hard disk Type

Before the SBC can use the IDE hard disk drive, the BIOS must know the drive type. The BIOS on the SBC contains a setup utility. **See section 5.7.**

2.20.2 Setting up a Hard Disk for use with MS-DOS

Before the IDE hard disk drive can be used by DOS, several setup steps must be completed. The first is low-level formatting. For all IDE hard disk drives purchased directly from CDI, low-level formatting will have been done prior to shipment.

Before the IDE hard disk drive can be used, it must be partitioned and formatted. The first of these steps is accomplished by booting the system from floppy and running the partition program FDISK.EXE (FDISK.COM on older versions on MS-DOS). FDISK is discussed in detail in Appendix D of the Microsoft MS-DOS User's Guide and User's Reference. We will not attempt to duplicate that discussion here, but will guide you through the basics required to initialize your drive.

WARNING

Partitioning a hard-disk will destroy any files that were on it.

With a diskette containing FDISK in your floppy drive, type

FDISK

and press the "Enter" key. FDISK will provide an option menu. Select the "Create DOS Partition or Logical DOS drive" option. On the next menu, select the "Create Primary DOS Partition" option. When the system asks if you want to use the maximum size DOS partition, answer "Y". The system will reboot afterwards, be sure to have a bootable floppy diskette in the floppy disk drive.

Once the drive has been partitioned, each partition must be formatted. For the primary partition, use the command:

FORMAT C: /S

to both format the partition and install a bootable copy of MS-DOS on the partition.

For any remaining partitions or drives, enter the command without the "/S" option (since they don't need MS-DOS). For example, the D: partition would be formatted with:

FORMAT D:

WARNING

Always specify a drive when formatting to avoid accidentally losing valuable files by formatting the wrong drive.

Refer to the Microsoft MS-DOS User's Guide and User's Reference for details on the format process.

Once formatted with the "/S" option, you can complete the MS-DOS installation procedure provided with your copy of MS-DOS.

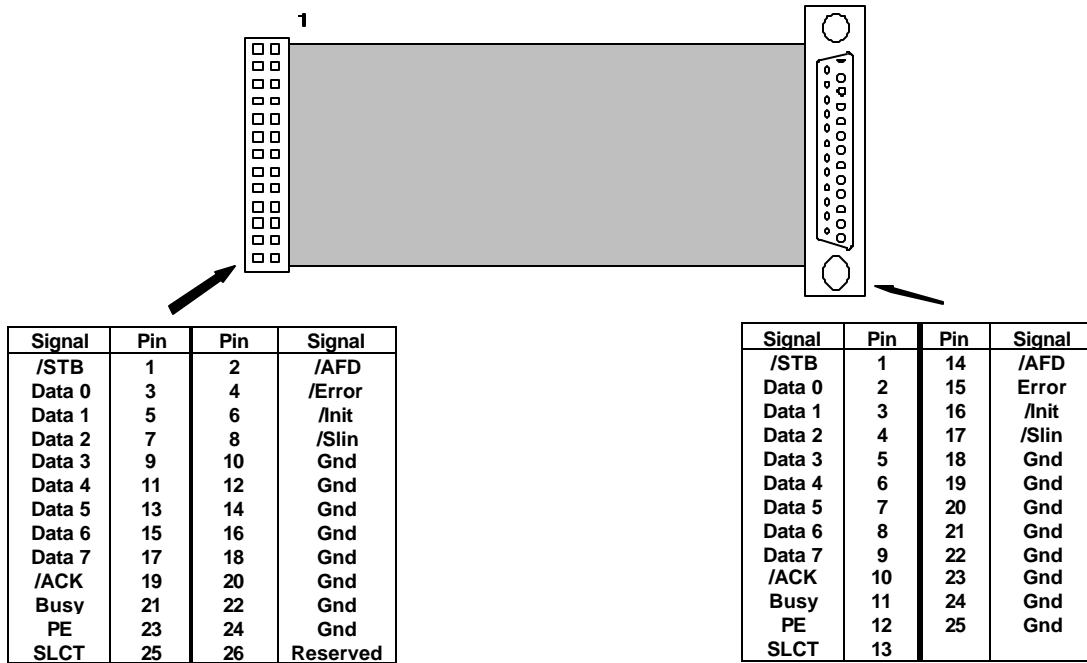
2.20.3 Setting up a Hard Disk for use with Windows 95

Insert the Windows 95 boot disk into the floppy disk drive and reboot your system.

When the first screen appears press enter to continue the Windows 95 installation. Select the "configure unallocated disk space" option, and press enter. The system will reboot automatically, and then format the hard disk for use with Windows 95. When formatting is complete the Windows 95 installation program will continue to install Windows 95.

2.21 Printer Interface (J8)

The printer port on the SBC is flat-cable-compatible with the standard PC 25-pin parallel connector as shown in the following figure. The 25-pin connector for the printer port is typically a female connector.



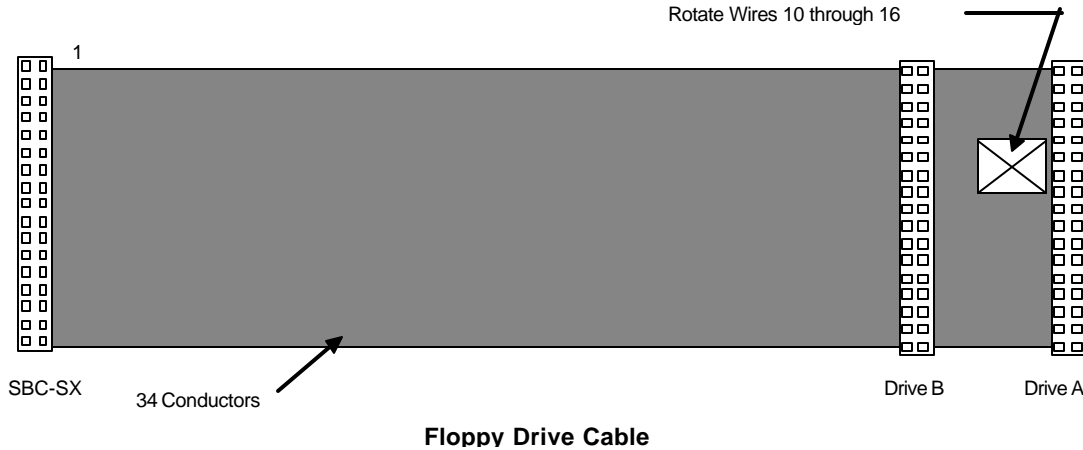
SBC to PC Parallel Cable

The 25-pin connector can be converted to a Centronics-compatible connector using an adapter cable available from most computer stores.

2.22 Floppy Interface (J10)

The SBC supports up to two floppy drives which, under MS-DOS, can be any combination of 5.25" and 3.5" drives.

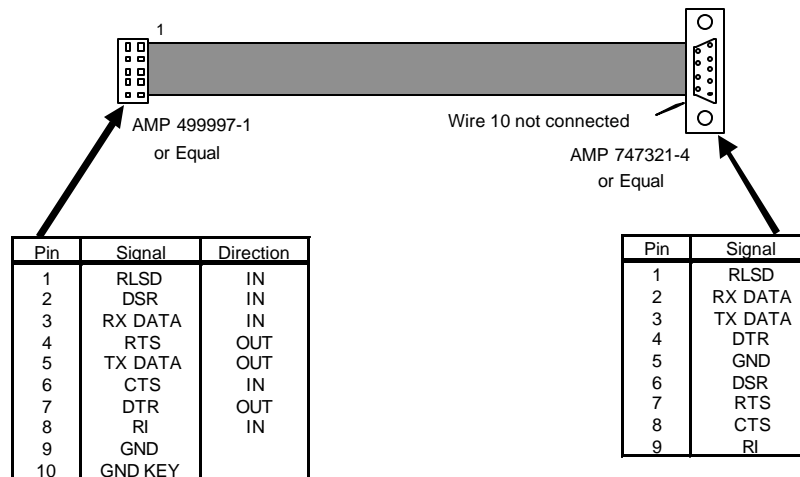
The drive cable for the SBC is identical to that used in an IBM-PC. Wires 10 through 16 are swapped. This swap, or flip, provides the drive select to the drives. Mount drive A: on the connector at the end of the cable. Drive B: uses the middle connector on the cable. Set the straps or configuration switch on the drives so that both drives are configured as the **second** drive in the system. Note that 3.5" floppy drives require header-type connectors while 5.25" drives require card-edge connectors.



The BIOS for the SBC must be informed of the types of floppy drives attached to the system. This is accomplished using the BIOS Setup. See Section 5.7.

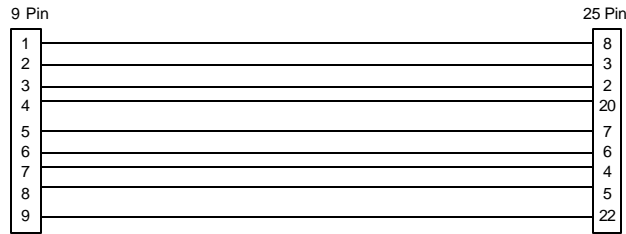
2.23 Serial Port Interface (J11 and J12)

The two serial ports on the PC/AT are flat-cable-compatible with the standard PC 9-pin serial connector, as shown in the following figure. The 9-pin connector is typically a male connector.



SBC TO PC 9 PIN STANDARD CABLE

Many devices now support the standard PC 9-pin connector. The original 25-pin connector may be connected as shown. This adapter is available from many suppliers. The 25-pin connector is typically a male. Note that the signal directions are those for data terminal equipment (DTE). The 25-pin connector is flat-cable-compatible with modems.



9 Pin to 25 Pin Adapter

2.24 Flat Panel Interface (J13)

Flat-panel displays (liquid crystal displays, electroluminescent, plasma, etc.) may be connected to the SBC through J13, a 2 x 20 flat-cable-compatible header connector. This connector can be used with panels with dual-row and some-single row connectors. CDI offers cable interfaces for common LCD or EL displays. Contact your applications engineer for availability on specific panel interfaces.

Signal	Pin	Pin	Signal
Con	1	2	B3/RGB3/VD15
Blank/WF/FR	3	4	UD3/R3/RGB15/VD7
LP	5	6	LD3/G3/RGB9/VD3
PCLK/XSCLL	7	8	RGB11/VD11
Reset	9	10	RGB11/VD11
Ground ⁽¹⁾	11	12	RGB10/VD10
Ground ⁽¹⁾	13	14	RGB17/VD9
Ground ⁽¹⁾	15	16	RGB16/VD8
Ground ⁽¹⁾	17	18	RGB4 for 18 Bit TFT/VD16
LD1/G1/RGB7/VD1	19	20	RGB5 for 18 Bit TFT/VD17
LD0/G0/RGB6/VD0	21	22	Ground
B2/RGB2/VD14	23	24	Ground
B1/RGB1/VD13	25	26	+5 Volts
+5 Volts	27	28	N/C
VRTC/YD/FP	29	30	N/C
Inverted PCLK/XSCLK	31	32	B0/RGB0/VD12
UD2/R2/RGB14/VD6	33	34	Key
UD1/R1/RGB13/VD5	35	36	+12 Volts
UD0/R0/RGB12/VD4	37	38	Switched +12 Volts
LD2/G2/RGB8/VD2	39	40	-12 Volts

Panel Interface

(1) Rev C and later only. Previous versions were N/C.
 Blank/WF is the blanking signal for CRTs, Plasma and EL panels. It is the backplane bias signal for LCDs.
 LP is the latch pulse for LCDs.
 PCLK/XSCLL is the pixel clock for CRTs. It is the shift clock for LCDs.
 R0-R3,G0-G3 and B0-B3 are the data bit's for 3-bit, 9-bit, and 12-bit TFTs.
 RGB0-RGB17 are the data bits for 18-bit TFTs
 UD0-UD7 are the data for the upper half of dual screen LCDs.
 LD0-LD7 are the data for the lower half of dual screen LCDs.
 Switched +12 Volts can be used in power sequencing for flat panels requiring a LCD bias voltage of +12 Volts.

2.25 External Power (J14)

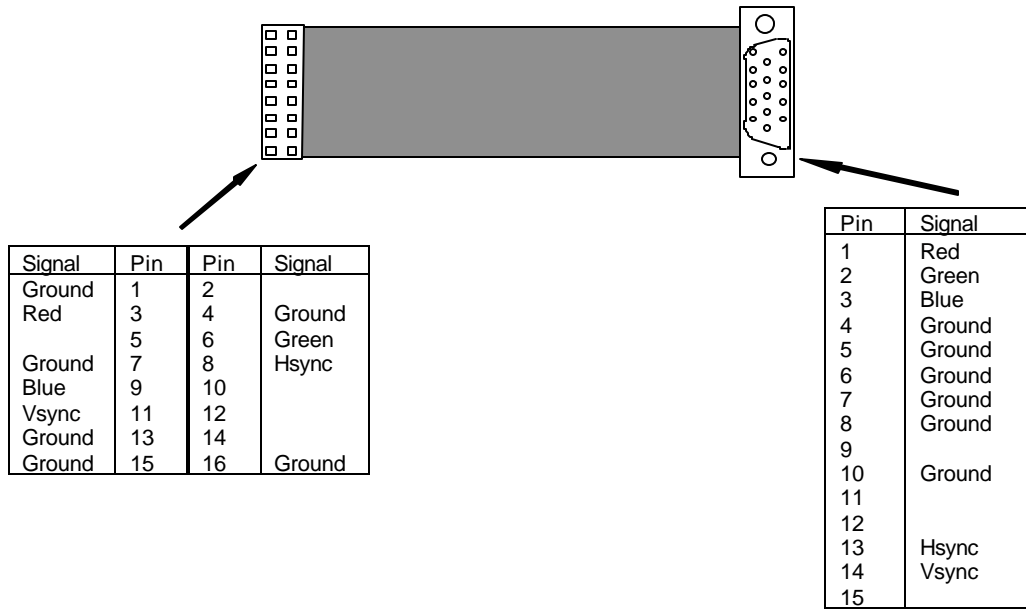
Power for peripheral devices may be supplied through J14. The anode of an LED may be connected to pin 4 and the cathode to pin 3 to provide a power on indicator.

Pin	Function
1	+5 Volts
2	Key
3	Ground
4	LED Anode
5	+12 Volts
6	-12 Volts

External Power Connections

2.26 VGA Video Interface (J9)

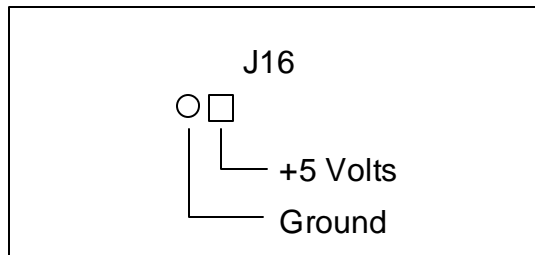
A VGA monitor is connected to the SBC using the VGA adapter module. This module re-routes the signals from the header connector, J9, to the industry standard 15-pin D-type connector as shown in the following figure.



SBC VGA cable

2.27 Fanned Heat Sink Power Connector (J16)

Under most circumstances the 486DX/DX2/DX4, and 5x86 processors need a heat sink with an attached fan. J16 provides power for the fan. Its pinouts are shown below.



Fanned Heat Sink Power

2.28 External Battery (J17)

Battery back-up is provided for a single static RAM in the RAM/ROM sockets. If a second static RAM is used, additional battery power must be provided using J17. The connections for J17 are shown below. The Battery Voltage must be between 3.6V and 5V.

Pin	Function
1	Ground
2	+ Battery
3	Ground

External Battery Connection

2.29 WatchDog LED (LE1)

The WatchDog LED turns on when the WatchDog Timer is refreshed by software (see section 3.5) .

2.30 Reset LED (LE2)

The Reset LED turns on whenever the SBC is reset. The SBC can be reset by the push button input, low voltage (see section 2.12) and loss of watchdog refresh (see section 4.3).

2.31 System DRAM Installation (U17, U18)

The SBC can support up to two 72-pin DRAM modules for a maximum of 64 Mbytes of memory. The module speed specification is a function of processor speed and module size as shown below. Module speed was selected as a function of both access time and fast page mode access time.

Refer to Section 2.0 for Bank 0 and Bank 1 locations.

Board Memory	U17 Bank 0	U18 Bank 1
1Mbytes	256 x 32	---
2Mbytes	256 x 32	256 x 32
4Mbytes	1M x 32	---
8Mbytes or:	1M x 32 2M x 32	1M x 32 ---
12Mbytes	2M x 32	1M x 32
16Mbytes or:	2M x 32 4M x 32	2M x 32 ---
20Mbytes	4M x 32	1M x 32
24Mbytes	4M x 32	2M x 32
32Mbytes or:	4M x 32 ---	4M x 32 8M x 32

Note:

For SX2-50: Use 80ns Fast Page SIMM Modules.

For DX2-66 / DX4-100 / 5x86-100: Use 70ns Fast Page SIMM Modules.

2.32 BIOS and Semiconductor Devices (U15, U37, U19, U41)

Refer to Section 2 for device locations.

U15 is the BIOS EPROM.

U37 is RAM/ROM 1.

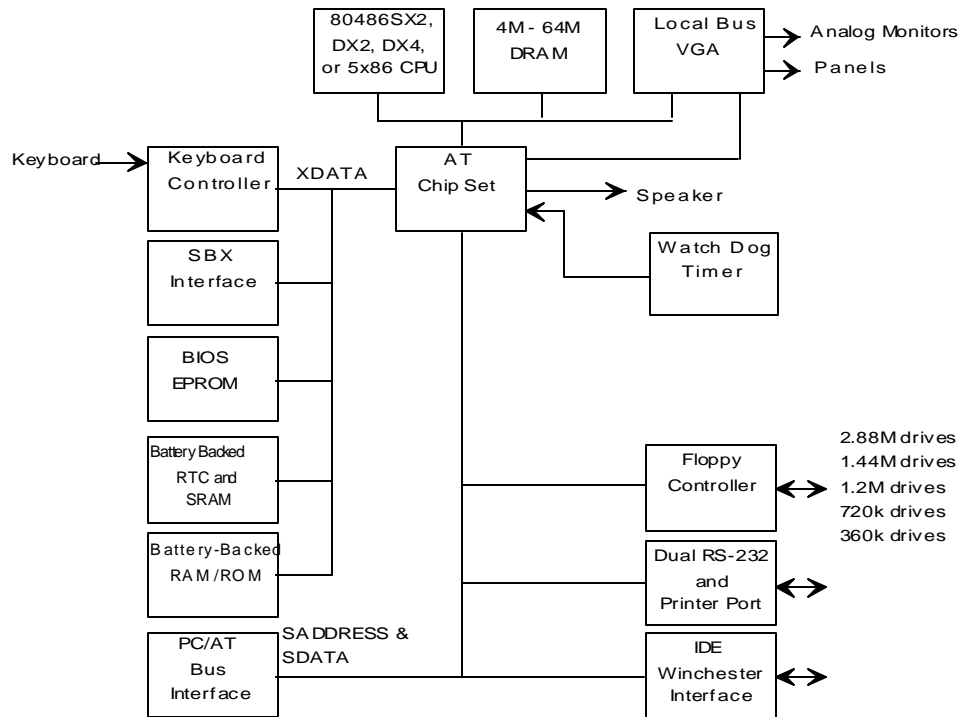
U19 is RAM/ROM 2.

U41 is RAM/ROM 3.

Use 200 ns or faster parts for the BIOS and Semiconductor Disk. The SBC slows down to 8 MHz for accesses to these devices.

3. PRODUCT OVERVIEW

The SBC is a module that provides a PC/AT with a PC Bus interface and a SBX connector. The following diagram is a block representation of the SBC module.



The module consists of the following components:

1. The CPU (80486SX2, DX2, DX4 or 5x86)
2. Socket for a 28010 BIOS EPROM
3. Three sockets for RAM/ROM disks
4. Up to 64 Mbytes of dynamic RAM
5. Keyboard port
6. Local bus video
7. Super VGA monitor and panel interface
8. Two RS-232 ports
9. Printer port
10. Floppy disk interface
11. AT IDE hard disk interface

The system also contains an optional WatchDog Timer with software selectable automatic WatchDog refresh, or refresh generated by your application program. The SBX connector of the module provides access to 16 I/O ports and a maskable Interrupt.

3.1 SBC Memory Map

The SBC memory map is similar to the memory map in any PC/AT system as shown in the following table. The battery-backed RAM/EPROM disk is mapped at A0000H to BFFFFFFH. VGA uses this region of memory in several modes. Software-controllable switches are provided to enable VGA memory and disable RAM/EPROM memory or disable VGA memory and enable RAM/EPROM memory. See section 4.5 for a description of the software for this operation.

A single BIOS ROM socket is provided on the SBC. This socket is compatible with 27010 EPROMs and 28010 Flash EPROMs. This EPROM contains the BIOS for the SBC as well as adapter modules (drivers) for the VGA and the semiconductor disk. The EPROM responds to memory reads from the EPROM as shown in the following table.

SF-13 Setting	Address
Video BIOS Enabled CDI Extensions Enabled	C0000H to CFFFFFFH F0000H to FFFFFFFH
Video BIOS Disabled CDI Extensions Enabled	CC000H - CFFFFFFH F0000H to FFFFFFFH
Video BIOS Enabled CDI Extensions Disabled	C0000H to CBFFFFH F0000H to FFFFFFFH
Video BIOS Disabled CDI Extensions Disabled	F0000H to FFFFFFFH

EPROM Map

Only video adapter modules may be installed from C0000H to CBFFFFH or the video adapter software will not function properly. ROM Tools (available separately) provides a utility to assist with adapter modules. For more information refer to CDI's ROM Tools User Manual.

Address	Function
00000	Interrupt Vector Table
00400	BIOS Data Area, Device Drivers, DOS, and Programs
A0000	VGA or RAM/EPROM
C0000	VGA BIOS
CC000	Adapter ROMs, PC Bus, and ROM Disk
F0000	BIOS and Setup

SBC Memory Map

The location of the EPROM moves in the memory map based on the setting of SF-13 as shown in the following figures. Network interfaces and some other peripheral boards map RAM in the PC ROM area.

Address	Function
C0000	VGA BIOS
CC000	CDI Extensions
F8000	BIOS

CDI Extensions Enabled - Video BIOS Enabled Map

Address	Function
C0000	PC Bus
CC000	CDI Extensions
F0000	BIOS

CDI Extensions Enabled - Video BIOS Disabled Map

Address	Function
C0000	VGA BIOS
CC000	PC Bus
F8000	BIOS

CDI Extensions Disabled - Video BIOS Enabled Map

Address	Function
C0000	PC Bus
F0000	BIOS

CDI Extensions Disabled - Video BIOS Disabled Map

The VGA occupies memory from A0000H to BFFFFH depending on the mode, as shown in the following figure. These addresses are the same as defined in the PC/AT. Address ranges listed as "not used" respond to memory reads and writes, but are not used by the video mode. Address ranges not identified do not respond to memory reads or writes.

A0000			Mode 13	Mode 14	Modes 15 & 16	Modes 17 & 18	Mode 19
A1F40			not used	not used			
A3E80							
A6D60					not used		
A9600						not used	
AFA00							not used
B0000							
B8000	Modes 0-3	Modes 4-6					
B8FA0	not used	even scans					
B9F40		not used					
BA000		Modes 4-6 odd scans					
BBF40		not used					
BFFFF							

VGA Memory Map

3.2 SBC I/O Map

The following figure shows the SBC I/O map. The addresses used are PC/AT compatible.

The PC/AT uses 10-bit I/O address decoding. Any I/O reference that uses greater than 10 bits is actually decoded based on the least significant 10 bits. For example: 480H would be decoded as 080H. The SBC uses 11-bit address decoding except in the first 100H I/O ports (PC/AT System Board functions) and the SBC Interrupt Service Port. The PC peripheral functions (IDE hard disk, floppy, CGA, parallel and serial) all appear at their normal addresses and at reflections on 800H boundaries (e.g., 320H and B20H). The PC System Board functions (DMA controller, interrupt controller, timer) use 10-bit decoding and will reflect on 400H boundaries (i.e., 020H and 420H).

The SBC has assigned the I/O ports after the first System Board reflection to the WatchDog strobe (580H-5BFH), the VGA-RAM/EPROM Memory Enable/Disable (540H-54FH), SBX connectors (740H-74FH) and the SBX Interrupt Source Port (4740H). These I/O ports will reflect on 800H boundaries. The VGA subsystem uses port 46E8H as an enable/disable.

Address	Function
000-01F	DMA Controller 1
020-03F	Interrupt Controller 1
040-05F	Counter Timer
060	Keyboard Controller
061	Port B register 8255
062-06F	Keyboard Controller
070-07F	Real-Time Clock / NMI Mask
080-08F	DMA Page Register
090-091	DMA Map Register
092	Alternate Gate A20 / Hot Reset
093-09F	DMA Map Register
0A0-0BF	Interrupt Controller 2
0C0-0DF	DMA Controller 2
0F0	Clear Math Co-processor Busy
0F1	Reset Math Co-processor
0F2-0F3	ACC Core Logic Control Registers
0F8-0FF	Math Co-processor
100-101	PC Bus
102	VGA Enable Register
103-1EF	PC Bus
1F0-1F7	Fixed Disk
1F8-21E	PC Bus

Address	Function
21F	Reserved
220-2F7	PC Bus
2F8-2FF	Com Port 2
300-377	PC Bus
378-37F	Printer Port
380-3AF	PC Bus
3B0-3BB	Monochrome Adapter (VGA)
3BC-3BF	PC Bus
3C0-3CF	VGA
3D0-3DF	Color Graphic Adapter (VGA)
3E0-3EF	PC Bus
3F0-3F7	Floppy Disk / Fixed Disk
3F8-3FF	Com Port 1
540-547	Enable VGA Memory Disable RAM/EPROM
548-54F	Disable VGA Memory Enable RAM/EPROM
550-55F	Flash Programming Voltage
5B0-5BF	Watchdog Strobe
740-747	SBX/MCS0
748-74F	SBX/MCS1
760	RAM/EPROM Bank Select
46E8	VGA I/O Wake-Up Port
4740-474F	SBX Interrupt Source

SBC I/O Map

3.3 SBC Interrupt Assignments

The following table lists the SBC interrupts. Interrupt 10, which is not assigned in the PC/AT, has been assigned to the SBX connectors.

Level	Function	Software Vector
NMI	Parity I/O Channel Check	2H
0	Timer 0	8H
1	Keyboard (buffer full)	9H
2	Controller 2 IRQ8-IRQ15	AH
3	Com Port 2	BH
4	Com Port 1	CH
5	PC Bus	DH
6	Diskette Controller	EH
7	Printer Port	FH
8	Real-Time Clock	70H
9	PC Bus IRQ 2 VGA (if SF-15 installed)	71H
10	SBX	72H
11	PC Bus	73H
12	PC Bus	74H
13	Math Co-processor	75H
14	Fixed Disk	76H
15	PC Bus	77H

SBC Interrupt Assignments

3.4 SBC DMA Assignments

The SBC DMA assignments are listed in the following tables. DMA Channels 0 through 3 support 8-bit transfers between 8-bit I/O devices and 8-bit or 16-bit memory. Each channel can transfer 64-kbyte blocks throughout the 16 Mbyte address range of the system on 64-kbyte boundaries. DMA Channel 4 is used to cascade DMA Channels 0 through 3. DMA Channels 5 through 7 support 16-bit transfers between 16-bit I/O and 16-bit memory. Each channel can transfer 128-kbyte blocks throughout the 32-Mbyte address range of the system on 64 kbyte boundaries.

DMA	Function
0	Not Used
1	Not Used
2	Floppy Diskette
3	Not Used
4	Cascade for Controller 1 DMA0-DMA3
5	Not Used
6	Not Used
7	Not Used

SBC DMA Assignments

Function	I/O Address
DMA 0	087
DMA 1	083
DMA 2	081
DMA 3	082
DMA 5	08B
DMA 6	089
DMA 7	08A
Refresh	08F

DMA Page Register Address

3.5 WatchDog Timer

A WatchDog timer monitors SBC activity to recover from software errors. After reset, ALE refreshes the WatchDog timer. WatchDog timer refresh may be transferred to program control by outputting to any I/O address from 5B0H through 5BFH. Your program must refresh the WatchDog timer at least once during the set period to avoid a Reset. The WatchDog timer period may be set to 150 milliseconds, 600 milliseconds, or 1.2 seconds depending on the setting of SF-4. (see section 2.4)

The WatchDog timer also monitors the power supply voltage. The voltage monitoring holds the SBC in Reset whenever the supply voltage drops below 4.75 Volts or 4.5 Volts, depending on the setting of SF-16.

3.6 RAM/ROM Disk Bank Select

As discussed in section 3.1, the RAM/ROM Disk occupies 128 kbytes at address A0000H. Bank selection increases the RAM/ROM Disk to up to 1 Mbyte. Writing to I/O port 760H (see section 4.4) sets the bank select register. The following table shows the bit layout of this register.

Bit	7	6	5	4	3	2	1	0
512k x 8 RAM	RAM/ROM Select1		A18	X	A17			RAM/ROM Select0
256k x 8 RAM	RAM/ROM Select1		A17	X	A17			RAM/ROM Select0
128k x 8 RAM	RAM/ROM Select1		X	1	X			RAM/ROM Select0
32k x 8 RAM	RAM/ROM Select1		X	X	X			RAM/ROM Select0
8k x 8 RAM	RAM/ROM Select1		X	X	X			RAM/ROM Select0
27080 EPROM	RAM/ROM Select1		A19	A18	A17			RAM/ROM Select0
27040 EPROM 28040 Flash	RAM/ROM Select1		1	A18	A17			RAM/ROM Select0
27020 EPROM 28020 Flash	RAM/ROM Select1		1	1	A17			RAM/ROM Select0
27010 EPROM 28010 Flash	RAM/ROM Select1		1	1	X			RAM/ROM Select0
27512 EPROM	RAM/ROM Select1		X	X	X			RAM/ROM Select0
27256 EPROM 28256 Flash	RAM/ROM Select1		X	X	X			RAM/ROM Select0

RAM/ROM Disk Bank Select Bits

Page Number	RAM/ROM1		RAM/ROM2		RAM/ROM3		
	RAM Select Bits (HEX)	EPROM Select Bits (HEX)	RAM Select Bits (HEX)	EPROM Select Bits (HEX)	RAM Select Bits (HEX)	EPROM Select Bits (HEX)	
00	01	01	80	80	81	81	Base Page for 27080
01	29	09	A8	88	A9	89	
02	21	11	A0	90	A1	91	
03	09	19	88	98	89	99	
04		21		A0		A1	Base Page for 27040, 27512 and 27256
05		29		A8		A9	
06		31		B0		B1	Base Page for 27020 and 27010
07		39		B8		B9	

RAM/ROM Page Select

Note:

That 8k x 8 RAMs have an address offset of 2000H. Also 27256 EPROMs have an address offset of 8000H. The Semiconductor Disk Software supplied with the SBC supports RAM/ROM devices greater than or equal to 32k. Support for smaller devices is the responsibility of the customer.

3.7 Automatic Wait States

The SBC automatically inserts wait states during accesses to on board peripherals and the AT expansion bus as shown in the following table. The on board peripherals are the floppy interface, hard disk interface, VGA, serial ports and printer port. The hard disk interface data port is 16 bit.

AT Expansion Memory VGA Memory		AT Expansion I/O On board I/O	
8 Bit	16 Bit	8 Bit	16 Bit
4	1	4	1

Wait States

3.8 Real-Time Clock RAM Usage

The battery-backed RAM in the real time clock stores system configuration. The following table shows the standard allocations. Reserved areas may be assigned functions within the BIOS. Use of this RAM may change without notice.

Address	Function
0H	Seconds
1H	Second Alarm
2H	Minutes
3H	Minute Alarm
4H	Hours
5H	Hour Alarm
6H	Day of week
7H	Day in month
8H	Month
9H	Year
0AH	Status Register A
0BH	Status Register B
0CH	Status Register C
0DH	Status Register D
0EH	Diagnostic Status
0FH	Shutdown Status
10H	Disk Drive Type
11H	RESERVED
12H	Fixed Disk Drive Type
13H	RESERVED
14H	Equipment Byte
15H	Low Base Memory
16H	High Base Memory 100H = 256k 200H = 512k 280H = 640k
17H	Low Expansion Memory
18H	High Expansion Memory 200H = 512k 400H = 1024k 600H-3C00H = up to 15360k
19H	Drive C Extended Byte
1AH	Drive D Extended Byte
1BH-2DH	RESERVED
2EH-2FH	Checksum based on 10H-2DH
30H	Low Expansion Memory
31H	High Expansion Memory 200H = 512k 400H = 1024k 600H-3C00H = up to 15360k
32H	Date Century (BCD)
33H	Information Flags
34H-3FH	RESERVED

Real-Time Clock RAM Usage

3.9 Flash Programming Voltage Port

The programming voltage for Flash memory (both BIOS and RAM/ROM) may be turned on or off by outputting to the Flash Programming Voltage port. Note that you must supply external +12V for the Flash programming voltage.

Bit	Function
7-1	Reserved
0	0 = Programming Voltage Disabled
	1 = Programming Voltage Enabled

Flash Programming Voltage Port

3.10 SBX Interrupt Port

The SBX interrupt port is used to determine which of the SBX interrupts has generated an interrupt. The port is at I/O address 4740H.

Bit	Function
7-4	Not used - no valid data
3-2	Always 0
1	SBX MINTR0
0	SBX MINTR1

SBX Interrupt Port

4. SOFTWARE OPERATION

Several good references are available that describe the hardware/software interface in the PC/AT (see Appendix B). This section provides information unique to the operation of the SBC and will not attempt to duplicate the material found in other references.

4.1 SBX I/O

The SBC provides access to up to 16 I/O ports using the SBX connector. These ports are from 740H to 74FH.

```
;*****
; SBC I/O Example
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
;
;*****

        SBX_PORT          EQU      740H

CODE    SEGMENT PARA PUBLIC

; Input and Output to an SBX port

        mov     dx, SBX_PORT          ; Select the port

        in     al, dx                ; read the port

        out    dx,al                 ; output to the port

CODE    ENDS

        END
```

Note:

An example showing a full implementation of this SBX function is in Appendix F.

4.2 SBX Interrupt Service

SBX maskable interrupts for the SBC are requested through SBC IRQ 10. The following code example demonstrates this process.

```
*****
; SBC Interrupt Service Example
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
;
*****

        SBX_INT      EQU      4740H

CODE    SEGMENT PARA PUBLIC

; Determine the SBX interrupt

        mov     dx, SBX_INT      ; Read the vector port

        in     al, dx           ; al now contains the vector read from
                                ; the SBX connector

CODE    ENDS

        END
```

Note:

An example showing a full implementation of this SBX function is in Appendix F.

4.3 WatchDog timer

The WatchDog timer provides a safe way to assure that your system continues to execute the program that you wanted it to run. Before activating the WatchDog timer, set up a routine that will periodically refresh the WatchDog timer. If the WatchDog timer is not refreshed often enough, it will reset the system. The WatchDog timer is refreshed by outputting any data to the WatchDog strobe port as shown in the following code example.

When your software takes control on the WatchDog refresh, the WatchDog LED on the SBC will turn on.

```
*****
; SBC WatchDog Refresh Example
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
;
*****

        WATCHDOG_STROBE      EQU            5B0H

CODE    SEGMENT PARA PUBLIC

; Refresh the WatchDog timer

        mov     dx, WATCHDOG_STROBE ; output anything to the strobe port

        out    dx,al

CODE    ENDS

        END
```

4.4 RAM/ROM Disk Bank Select

The RAM/ROM Disk bank may be selected by writing to the output port 750H as shown in the following code.

```

;*****
; SBC RAM/ROM Disk Bank Select
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
;
;*****

        Bank_SELECT      EQU          760H

CODE    SEGMENT PARA PUBLIC

; Set RAM/EPROM bank
; assumes AL contains the bank output pattern - see section 3.6

        mov     dx, Bank_SELECT

        out    dx,al

CODE ENDS

        END

```

Note:

An example showing a partial implementation of this function is in Appendix G.

4.5 Enable/Disable VGA Memory

The VGA memory may be disabled by writing to output port 548H and outputting a 0 to port 46E8H. This also enables the RAM/ROM memory.

```
*****
; SBC VGA Memory Disable
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
;
*****

        VGA_DISABLE          EQU      548H
        ADAPTER_ENABLE_REG   EQU      46E8H
        ADAPTER_DISABLE_DATA EQU      0H

CODE    SEGMENT PARA PUBLIC

; Disable VGA Memory - Enable RAM/EPROM Memory

        mov     dx, VGA_DISABLE          ; write anything to the VGA Disable port

        out    dx, al

        mov     dx, ADAPTER_ENABLE_REG   ; write Adapter Disable pattern to port
        mov     al, ADAPTER_DISABLE_DATA
        out    dx, al

CODE    ENDS

        END
```

Note:

An example showing a partial implementation of this function is in Appendix G.

The VGA memory may be enabled by writing to output port 540H and outputting an 8H to port 46E8H. This also disables the RAM/ROM memory.

```
*****
; SBC VGA Memory Enable
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
;
*****

        VGA_ENABLE           EQU        540H
        ADAPTER_ENABLE_REG   EQU        46E8H
        ADAPTER_ENABLE_DATA  EQU        8H

CODE    SEGMENT PARA PUBLIC

; Enable VGA Memory - Disable RAM/EPROM Memory

        mov     dx, VGA_ENABLE           ; write anything to the VGA
                                         Enable port

        out     dx, al

        mov     dx, ADAPTER_ENABLE_REG   ; write Adapter Enable
                                         pattern to port

        mov     al, ADAPTER_ENABLE_DATA
        out     dx, al

CODE    ENDS

        END
```

Note:

An example showing a partial implementation of this function is in Appendix G.

4.6 Flash Programming Voltage Enable/Disable

Flash programming voltage may be turned on and off by using the Flash Programming Voltage port. The Flash programming voltage is enabled by outputting a 1 to port 550H.

Flash programming algorithms vary between chip manufacturers and can become quite involved. They can cause damage to the Flash EPROM if programmed or erased improperly. Computer Dynamics provides a separate utility (ROMDISK.EXE) which will image a floppy diskette onto a Flash EPROM. This utility is part of the ROM Tools package. Contact Computer Dynamics for more information.

```
*****
; SBC Flash Programming Voltage Enable
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
*****

Flash_PORT      EQU    550H
Flash_ENABLE    EQU    1

CODE    SEGMENT PARA PUBLIC

; Enable Flash Programming Voltage

    mov     dx, Flash_PORT      ; write enable value to the Flash Programming PORT
    mov     al, Flash_ENABLE

    out     dx,al

CODE ENDS

END
```

The Flash programming voltage is disabled by outputting a 0 to port 550H.

```

;*****
; SBC Flash Programming Voltage Enable
;
; This source code may be used in applications using the
; Computer Dynamics SBC
;
; (c) Lines Unlimited 1989
;*****

Flash_PORT      EQU    550H
Flash_DISABLE   EQU    0

CODE    SEGMENT PARA PUBLIC

; Enable Flash Programming Voltage

PORT
    mov     dx, Flash_PORT      ; write disable value to the Flash Programming
    mov     al, Flash_ENABLE
    out    dx,al

CODE ENDS
    END

```

4.7 ACC 2178 Registers

The ACC-2178 is an advanced PC/AT-compatible single-chip 80386DX/80486DX chip set. The chip incorporates advanced memory management features which include page mode, support for 2/4-way interleaving and shadow RAM for system BIOS and VGA BIOS.

The ACC-2178 performs the CPU peripheral support functions of dual DMA Controllers, Memory Mapper, Timer/Counters, cascaded Interrupt Controllers, Bus Controller and supporting circuitry. An AT-Bus clock is generated by the ACC-2178. The frequency of the AT-Bus clock is programmable, and can range from 7 Mhz to 12 Mhz.

Access to the core logic registers is done via an indirect addressing scheme using I/O ports 0F2h and 0F3h. For example, writing E8h to register 02H would be done by outputting a 02H to I/O port 0F2h, then writing E8h to register 0F3h. Contact your CDI applications engineer for a more detailed description of the ACC-2178 core logic registers.

4.8 Super I/O Options

The Super I/O chip contains the floppy controller, two serial ports, printer port and a portion of the IDE interface. Port addresses for COM1, COM2, LPT1, floppy and IDE can be mapped and turned on/off at setup (see section 5.8.1).

4.9 SBC Cache Control

The i486 contained in the SBC contains an internal 8-kbyte unified instruction and data cache with a 16-byte line size. The cache is four-way set associative. Cache lines are allocated on read misses. A least recently used algorithm is used with the four-way set associative configuration. Cache lines fills are performed only for read misses, never for write misses.

The SBC does NOT cache I/O, locked bus cycles to memory, INTA bus cycles, or HALT/SHUTDOWN cycles. Non-cacheable regions can be specified in setup.

The cache is invalidated by the 486-compatible invalidate instructions (INVD and WBINVD) and by DMA or Bus Masters writing to system memory.

4.10 Memory Managers (DOS-Based SBC Systems Only)

Computer Dynamics now provides support for all major memory managers and for protected mode applications. This is provided via an installable device driver which is inserted into the CONFIG.SYS file.

4.10.1 CDIMEM.SYS

Provided on the SBC Utility diskette in the back of the manual is a device driver (CDIMEM.SYS) which will allow the SBC to make use of memory managers and protected mode programming. This device driver must be installed prior to any memory managers in the CONFIG.SYS file. It has no command line parameters and is installed simply as follows:

```
DEVICE=CDIMEM.SYS
```

The following third party memory managers have been tested successfully by Computer Dynamics. Also, the memory manager provided with the DOS versions listed below has been tested.

```
Quarter-deck QEMM Version 6  
Quadtel MM  
DR-DOS Version 5.0  
DR-DOS Version 6.0  
MS-DOS Version 5.0  
MS-DOS Version 6.22
```

4.10.2 Memory Mappings and Considerations

For the SBC, all of Computer Dynamics' ROM-based device drivers will reside between CB00:0000 and CFFF:0000. This memory range should be excluded manually on the memory manager command line if the ROM modules are not detected by the memory manager automatically. In addition, the exclusion of A000:0000 through BFFF:0000 is required.

Note:

If you have disabled the CDI Extensions via SF-13, these exclusions are not needed. You only need to exclude them if you are using CDI Extensions.

Note:

CDIMEM.SYS is only needed if the CDI Extensions are enabled via SF-13. It may be placed anywhere in the file as long as it is before any high memory, extended memory, or expanded memory operations or drivers. It is a good practice to make CDIMEM.SYS the first line of the CONFIG.SYS file.

4.11 Dither Support for Color TFT Flat Panels Displays

Standard VGA video uses a DAC to generate the color signals on an analog monitor. This DAC contains 256 registers. Each register contains a color formula comprised of 6 bits Red, 6 bits Green, and 6 bits Blue. The specific values contained in each register are then used to drive the analog outputs to the monitor.

Since TFT displays are digital, the SBC's video controller implements a "digital DAC". The front end of the DAC is the same as the standard DAC, but the output is provided in different digital formats suitable to driving various types of panels. In addition, many of the standard color TFT LCD displays are 9-bit interface (3 bits Red, 3 bits Green, 3 bits Blue). This means that the internal 6-bits-per-color in each DAC register must be cut down to 3-bits-per-color to drive the panel. The most obvious effect of this color loss is visible in an image in which an area consists of a single color that progresses from very light to very dark. In this type of picture you will be able to see "color bands" instead of smooth transitions from light to dark.

To minimize the effect of the digital interface, the SBC has the capability of dithering. Dithering is a process where specific pixels or groups of pixels are mapped to various intensities and patterns at the video controller level. Depending on the desired 6-bits-per-color in the DAC, the resulting 3-bits-per-color coming out of the controller are constantly modified in real time to give the effect of the desired color. This is somewhat similar to flashing the colors red and white on the screen. If the colors are flashed fast enough, the human eye will interpret them as pink.

4.11.1 The Dither TSR

The SBC is pre-configured with the optimum dither method for flicker-free operation in the DOS environment. However, the SBC Samples and Utilities program, dither.exe, allows the you to change the dither method or to disable it if desired.

The command line is as follows:

Usage: **dither [0-6] | [I] | [U]**

With no argument it reports the current dither mode

0	Read Current Dither Settings (same as with no argument)
1	2 Frame - 27K Color Dithering
2	2 Frame - 180K Color Dithering
3	2 & 3 Frame - 256K Color Dithering
4	NO Dither - 512 True Colors
5	NO Dither - 512 True Colors
6	Space Dithering - 27K Colors
I\U	Install\Unload TSR

As a TSR, [SHIFT-ALT-F1] through [SHIFT-ALT-F6] perform the same functions as the command line options 1 through 6. [SHIFT-ALT-R] will display the current dither mode if in text modes 3 or 7. Otherwise it will be ignored.

The dither program has two modes of operation:

1. command line operation, and
2. TSR operation.

When executed from the command line, the change is immediate and remains in effect until the system is re-booted or until dither is executed again with a different option. Below are some examples of using dither as a command line utility.

```
dither -0          read current dither mode
dither -4          disable dithering
dither -5          disable dithering (mode 5 is the same as mode 4)
dither -6          Space Dithering - 27K colors
dither -? Or dither /h  Help - displays a usage similar to that shown above.
```

Dither can also be installed as a TSR. This is done by typing:

```
dither -i
```

To unload the TSR simply type:

```
dither -u
```

When installed as a TSR there are 8 hotkeys. They are [ALT-SHIFT-F1] through [ALT-SHIFT-F6] and [ALT-SHIFT-R]. [ALT-SHIFT-F1] through [ALT-SHIFT-F6] have the same effect as typing dither -1 through dither -6. They simply set the specified mode. [ALT-SHIFT-R] has the same effect as dither -0 if in text modes 3 or 7. Otherwise it is ignored. The TSR capability allows the customer to load an application and then determine the most appropriate dither option without having to drop out to the command line between each dither mode change.

Note:

Dither TSR will not function on the Windows desktop. However, it can be executed as a command line utility from a full screen DOS window under Windows. Also any dither mode that is set prior to starting Windows will remain in effect during Windows unless it is changed in a full screen DOS window as stated earlier.

To determine the default dither mode for your system, power up the SBC unit. Then run dither -0. Dither.exe can be placed in the autoexec.bat file if the default dither mode is not acceptable in your environment.

4.12 FVBIOS

There are times when the only part of the BIOS ROM that needs changing or updating is the video BIOS portion. For such cases a utility has been developed called FVBIOS. This utility takes only a video BIOS image and updates a current BIOS Flash ROM with the video BIOS. The command line is:

```
FVBIOS <image file> [f1 | f2]
```

where:	f1	Force SBC-DX	(486SX / 486DX / 486DX/2)
	f2	Force SBC-SXE/SBC-486	(386SX / 486SLC)

Normally the only command line parameter needed is the input filename. FVBIOS will automatically detect the type of Computer Dynamics SBC board on which it is running. If CDI Extensions have been strapped out, FVBIOS will not be able to tell which board it is on, autodetection logic will fail and

FVBIOS will prompt the user to specify the appropriate board type from a choice list. If the user has CDI Extensions strapped out and does not wish to be prompted, the board type can be "forced" on the FVBIOS command line.

Note:

The SBC comes with a standard EPROM BIOS. The FVBIOS utility requires an optional Flash BIOS and the SBC must be restrapped for flash BIOS support prior to using this utility.

5. BIOS

Phoenix/Quadtel's AT-Compatible BIOS Version 4.05 for i486 is the culmination of years of effort on the part of clean room developers. This effort produced a standard BIOS product which can be easily customized for a variety of operating environments.

The 4.05 BIOS is comprised of three distinct parts: the AT-Compatible Core, the Hardware Adaptation Module, and the BIOS Configuration Parameters. The 4.05 BIOS was designed so that the AT-Compatible Core is identical, at the object and source code level, in all of the BIOS products.

Each BIOS product contains a unique Hardware Adaptation Module which works in conjunction with the AT-Compatible Core. This module contains the code which is matched to the system board architecture. It will support any special hardware features such as shadow RAM and memory configuration.

The BIOS Configuration Parameters consist of a block of data which control the operation of the BIOS itself. These parameters contain information about memory configuration, what error messages should be displayed, and other instructions for BIOS performance.

The method of building the BIOS from distinct components is entirely unique. Most vendors support different types of hardware by using "conditional assembly" or by maintaining different sets of program source code for each product. This means that each new version of the BIOS produced must be retested for AT compatibility. It also does not ensure that updates to the core functions of the BIOS are incorporated into all versions.

5.1 AT-Compatible Core Characteristics

Phoenix/Quadtel's BIOS products are developed using a clean room development methodology to ensure that no possibility of copyright infringement exists. After completion, the BIOS products are tested and approved by US Customs so that import of the BIOS is trouble-free.

The AT-Compatible Core is a fully functional BIOS with thousands of working units in the field. Since the Core is completely independent of the Hardware Adaptation Module, compatibility with the industry standard is assured for all of *Phoenix*/Quadtel's BIOS products.

All *Phoenix*/Quadtel's AT-Compatible BIOS products are designed to operate in both standard and highly custom environments. A proprietary technique allows the BIOS to operate independent of the CPU speed while continuing to provide consistent operation. The BIOS will perform from 6 MHz to 33 MHz without any change to internal time-outs or delays.

The 4.05 BIOS uses 64k of ROM from physical address F0000H to FFFFFH. The BIOS itself resides completely in the 32k portion of the BIOS from F8000-FFFFF.

During the power-on-self-test (POST) of the BIOS, additional external adapters will be initialized if they are presented in the proper format. The requirements are:

1. The code must reside from address C0000H to F0000H.
2. The code must reside on a 2k boundary.
3. The first two bytes of the code must be 55H and AAH.
4. The third byte must contain the number of 512 byte blocks. The length must be a multiple of 2k.
5. The fourth byte must contain a jump to the start of the initialization code.
6. The code must checksum to zero (byte sum).

Note:

The address space from C0000H to CC000H is reserved for external video adapters (e.g., EGA, VGA). The address space from D0000H to E0000H is typically used by expanded memory (EMS).

5.2 Fixed Disk Tables

The *Phoenix/Quadtel* 4.05 BIOS supports a total of 46 drive types and a user definable drive type (Type 47). The following table shows the standard 46 drive types provided.

The user definable drive type allows the user to create drive parameters for both fixed disk 0 or fixed disk 1. Using the built-in SETUP program, you can specify the cylinders, heads, sectors, write precompensation, and landing zone for a specific drive. With this feature, custom drive tables can be avoided and special software for supporting non-standard drives is usually unnecessary.

The user-defined drive information is stored in the clock CMOS RAM. Before the boot operation begins, fixed disk parameter tables are built at physical address 160H for drive 0 and 170H for drive 1 by default.

Type	Cylinders	Heads	Sectors	Write Precompensation	Landing Zone
1	306	4	17	128	305
2	615	4	17	300	615
3	615	6	17	300	615
4	940	4	17	512	940
5	940	6	17	512	940
6	615	4	17	-1	615
7	462	8	17	256	511
8	733	5	17	-1	733
9	900	15	17	-1	901
10	820	3	17	-1	820
11	855	5	17	-1	855
12	855	7	17	-1	855
13	306	8	17	128	319
14	733	7	17	-1	733
15	Reserved				
16	612	4	17	0	633
17	977	5	17	300	977
18	977	7	17	-1	977
19	1024	7	17	512	1023
20	733	5	17	300	732
21	733	7	17	300	732
22	733	5	17	300	733
23	306	4	17	0	336
24	612	4	17	305	663
25	612	2	17	300	612
26	614	4	17	-1	614
27	820	6	17	-1	820
28	977	5	17	-1	977
29	1218	15	36	-1	1218
30	1224	15	17	-1	1224
31	823	10	17	512	823
32	809	6	17	128	809
33	830	7	17	-1	830
34	830	10	17	-1	830
35	1024	5	17	-1	1024
36	1024	8	17	-1	1024
37	615	8	17	128	615

Type	Cylinders	Heads	Sectors	Write Precompensation	Landing Zone
38	1024	8	26	-1	1024
39	925	9	17	-1	925
40	1024	9	17	-1	1023

Type	Cylinders	Heads	Sectors	Write Precompensation	Landing Zone
41	918	15	17	-1	917
42	1024	15	17	-1	1023
43	823	10	34	-1	822
44	969	5	34	-1	968
45	969	7	34	-1	968
46	969	9	34	-1	968

IDE Drive Types

5.3 Recoverable POST Errors

Whenever a recoverable error occurs during power-on-self-test (POST), the BIOS displays an error message describing the problem. The BIOS also issues a beep code (one long tone followed by two short tones) during POST if the video configuration fails or if an external ROM module does not properly checksum to zero.

An external ROM module can also issue audible errors, usually consisting of one long tone followed by a series of short tones.

There are several POST routines that issue a **POST Terminal Error** and shut down the system if they fail. Before shutting down the system, the terminal-error handler issues a beep code signifying the test point error, writes the error to port 80H, attempts to initialize the video, and writes the error in the upper left corner of the screen (using both mono and color adapters).

The routine derives the beep code from the test point error as follows:

1. The 8-bit error code is broken down to four 2-bit groups.
2. Each group is made one-based (1 through 4) by adding 1.
3. Short beeps are generated for the number of times in each group.

Example:

Testpoint 01AH = 00 01 10 10 = 1-2-3-3 beeps

If the BIOS detects error 2C, 2E or 30, it displays an additional word of information reflecting the bit or address line that failed. For example, if "2C 0002" is displayed, address line 1 (represented by bit one) has failed. If "2E 1020" is displayed, then data bits 12 and 5 have failed in the upper 16 bits. The BIOS sends the same information to the port-80 LED display. The check point code is followed by a delay, the high order byte, another delay, and then the low order byte of the error. This will be repeated continuously.

5.4 BIOS Testpoints and Beep Codes

At the beginning of each POST routine, the BIOS outputs the test point error code to I/O address 80H. Use this code during trouble shooting to establish at what point the system failed and what routine was being performed.

If the BIOS detects a terminal error condition, it halts POST after issuing a terminal error beep code and attempts to display the error code on upper left corner of the screen. It attempts repeatedly to write the error to the screen, causing "hash" on some CGA displays. If the system hangs before the BIOS can process the error, the value displayed at the port 80H is the last test performed. In this case, the screen does not display the error code.

The following is a list of the checkpoint codes written at the start of each test and the beep codes issued for terminal errors:

Code	Beeps	POST Routine Description
02	1-1-1-3	Verify real mode
04	1-1-2-1	Get CPU type
06	1-1-2-3	Initialize system hardware
08	1-1-3-1	Initialize chipset registers with initial POST values
09	1-1-3-2	Set in POST flag
0A	1-1-3-3	Initialize CPU registers
0B		Enable CPU Cache
0C	1-1-4-1	Initialize cache to initial POST values
0E	1-1-4-3	Initialize I/O
10	1-2-1-1	Initialize power management
11	1-2-1-2	Load alternate registers with initial POST values
12	1-2-1-3	Jump to UserPatch0
14		Initialize keyboard controller
16	1-2-2-3	BIOS ROM checksum
18	1-2-3-1	8254 timer initialization
1A	1-2-3-3	8237 DMA controller initialization
1C	1-2-4-1	Reset programmable interrupt controller
20	1-3-1-1	Test DRAM refresh
22	1-3-1-3	Test 8742 keyboard controller
24	1-3-2-1	Set ES segment register to 4 GB
28	1-3-3-1	Autosize DRAM
2A	1-3-3-3	Clear 512k base RAM
2C	1-3-4-1	Test 512k base address lines
2E	1-3-4-3	Test 512k base memory
32	1-4-1-3	Test CPU bus-clock frequency
34		Test CMOS RAM
36		Warm Start Shutdown
37	1-4-2-4	Reinitialize the chipset
38	1-4-3-1	Shadow system BIOS ROM
39	1-4-3-2	Reinitialize the cache
3A	1-4-3-3	Autosize cache
3C	1-4-4-1	Configure advanced chipset registers
3D	1-4-4-2	Load alternate registers with CMOS values
40	2-1-1-1	Set Initial CPU speed
42	2-1-1-3	Initialize interrupt vectors
44	2-1-2-1	Initialize BIOS interrupts
46	2-1-2-3	Check ROM copyright notice
47	2-1-2-4	Initialize manager for PCI option ROMs
48	2-1-3-1	Check video configuration against CMOS
49	2-1-3-2	Initialize PCI bus and devices
4A	2-1-3-3	Initialize all video adapters in system
4B		Display Quiet Boot Screen
4C	2-1-4-1	Shadow video BIOS ROM

Code	Beeps	POST Routine Description
4E	2-1-4-3	Display copyright notice
50	2-2-1-1	Display CPU type and speed
52	2-2-1-3	Test keyboard
54	2-2-2-1	Set key click if enabled
56	2-2-2-3	Enable keyboard
58	2-2-3-1	Test for unexpected interrupts
5A	2-2-3-3	Display prompt "Press F2 to enter SETUP"
5C	2-2-4-1	Test RAM between 512 and 640k
60	2-3-1-1	Test extended memory

Code	Beeps	POST Routine Description
62	2-3-1-3	Test extended memory address lines
64	2-3-2-1	Jump to UserPatch1
66	2-3-2-3	Configure advanced cache registers
68	2-3-3-1	Enable external and CPU caches
6A	2-3-3-3	Display external cache size
6C	2-3-4-1	Display shadow message
6E	2-3-4-3	Display non-disposable segments
70	2-4-1-1	Display error messages
72	2-4-1-3	Check for configuration errors
74	2-4-2-1	Test real-time clock
76	2-4-2-3	Check for keyboard errors
7A		Test for Keylock ON
7C	2-4-4-1	Set up hardware interrupt vectors
7E	2-4-4-3	Test coprocessor if present
80	3-1-1-1	Disable onboard I/O ports
82	3-1-1-3	Detect and install external RS232 ports
84	3-1-2-1	Detect and install external parallel ports
86	3-1-2-3	Re-initialize onboard I/O ports
88	3-1-3-1	Initialize BIOS Data Area
8A	3-1-3-3	Initialize extended BIOS data area
8C	3-1-4-1	Initialize floppy controller
90	3-2-1-1	Initialize hard-disk controller
91	3-2-1-2	Initialize local-bus hard-disk controller
92	3-2-1-3	Jump to UserPatch2
94	3-2-2-1	Disable A20 address line
95		Install CD ROM for Boot
96	3-2-2-3	Clear huge ES segment register
98	3-2-3-1	Search for option ROMs
9A	3-2-3-3	Shadow option ROMs
9C	3-2-4-1	Set up power management
9E	3-2-4-3	Enable hardware interrupts
A0	3-3-1-1	Set time of day
A2	3-3-1-3	Check key lock
A4		Initialize typematic rate
A8	3-3-3-1	Erase F2 prompt
AA	3-3-3-3	Scan for F2 key stroke
AC	3-3-4-1	Enter SETUP
AE	3-3-4-3	Clear in-POST flag
B0	3-4-1-1	Check for errors
B2	3-4-1-3	POST done - prepare to boot operating system
B4	3-4-2-1	One beep
B5		Display MultiBoot Menu
B6	3-4-2-3	Check password (optional)
B8	3-4-3-1	Clear global descriptor table
BC	3-4-4-1	Clear parity checkers
BE	3-4-4-3	Clear screen (optional)
BF	3-4-4-4	Check virus and backup reminders
C0	4-1-1-1	Try to boot with INT 19
DO	4-2-1-1	Interrupt handler error
D2	4-2-1-3	Unknown interrupt error
D4	4-2-2-1	Pending interrupt error
D6	4-2-2-3	Initialize option ROM error
D8	4-2-3-1	Shutdown error
DA	4-2-3-3	Extended block move
DC	4-2-4-1	Shutdown 10 error

The following Testpoints and codes apply to the boot Block in FLASH ROM's only.

Code	Beeps	POST Routine Description
E2	4-3-1-3	Initialize the Chipset
E3	4-3-1-4	Initialize the refresh counter
E4	4-3-2-1	Check for forced FLASH
E5	4-3-2-2	Check HW status of ROM
E6	4-3-2-3	BIOS ROM is OK
E7	4-3-2-4	Do a complete RAM test
E8	4-3-3-1	Do OEM initialization
E9	4-3-3-2	Initialize the interrupt controller
EA	4-3-3-3	Read in bootstrap code
EB	4-3-3-4	Initialize all vectors
EC	4-3-4-1	Boot the FLASH program
ED	4-3-4-2	Initialize the boot device
EE	4-3-4-3	Boot code was read OK

5.5 BIOS Messages

The following is an alphabetic list of error and status messages which can be generated by the BIOS. Included is an explanation of each message. Many of the messages refer to the built-in SETUP program. For more information, see the SETUP section.

Diskette Drive A, B error - This error will be displayed if diskette drive A is present, but fails the BIOS POST diskette tests. Check to see that the proper diskette type is defined with SETUP and that the diskette drive is attached correctly. This error can occur for diskette drives A or B.

Entering SETUP - Starting the SETUP program.

Extended RAM Failed at offset: - This error will be displayed in the event that a RAM error is detected in the extended memory. Above this message, the k address which failed will be displayed. This message will show the offset in the 64k block at which the error was detected.

Extended RAM Passed - This message will be displayed to indicate that the extended memory test is occurring. The amount of RAM tested (in kbytes) will be displayed before this message. Not all systems contain extended memory.

Failing bits: - This message will show a hex number which is a map of the bits which have failed the memory test (System, Extended, or Shadow). Each bit of the number displayed which is a one is a failing bit.

Fixed Disk 0,1 Failure - This error will be displayed if fixed disk drive 0 is present, but fails the BIOS POST fixed disk tests. Check to see that the proper fixed disk type is defined with SETUP and that the fixed disk drive is attached correctly. This error can occur for fixed disk drives 0 or 1.

Fixed Disk Controller Failure - This error will be displayed if the fixed disk controller test in the BIOS POST fails. The fixed disk controller may need to be replaced.

Incorrect Drive A,B type - Run SETUP - Type of floppy drive is not correctly identified in SETUP.

Invalid NVRAM media type - Problem with NVRAM (CMOS) access.

Keyboard controller error - The keyboard controller test during the POST has failed if this message is displayed. The keyboard controller or the keyboard may need to be replaced or the system board may need to be repaired.

Keyboard error - This error will be displayed if a stuck key is found during the BIOS POST. The scan code of the key will be displayed before error message.

Keyboard locked - Unlock key switch - This message will be displayed if the system keylock is on. Unlock the system keylock to proceed.

Monitor type does not match CMOS - Run SETUP - This error will be displayed if the monitor adapter type does not match the type defined in the SETUP program and stored in CMOS memory. For example, if a CGA card is present and the SETUP is set to monochrome, this error will be displayed. Run the SETUP program and set the correct monitor type.

Operating system not found - The Operating system cannot be located on either Drive A: or Drive C:. Enter SETUP and see if fixed disk and Drive A: are properly identified.

Parity Check 1 - This error will be generated if a parity error is detected on the system board. The BIOS will attempt to locate the address which caused the parity error and display it on the screen. If it cannot locate the address causing the error, it will display "????".

Parity Check 2 - This error will be generated if a parity error is detected on the I/O channel. The BIOS will attempt to locate the address which caused the parity error and display it on the screen. If it cannot locate the address causing the error, it will display "????".

Press <F1> to resume, <F2> to SETUP - Displayed after any recoverable error message. Press <F1> to start the boot process or <F2> to enter SETUP and change any settings.

Previous boot was incomplete - This error will be displayed if the BIOS POST operation is terminated by a reset or power down before it is completed. On some systems (especially those with wait state control), this can be caused by an incorrect system configuration. Run SETUP and verify that the configuration is correct. This error will be cleared automatically the next time the system reboots.

Press <Esc> to skip memory test ... - This will be displayed during the memory tests. Pressing <Esc> will cause the memory tests to be aborted. Memory will be zeroed as it would be on a soft reset (Ctrl-Alt-Del). Do not press any key other than <Esc> or a keyboard error may result.

Real-time clock error - This error will be displayed if the real-time clock test fails during the BIOS POST operation. Your system board needs to be repaired.

Shadow RAM Failed at offset: - This error will be displayed in the event that a RAM error is detected in the shadow memory. Above this message, the k address which failed will be displayed. This message will show the offset in the 64k block at which the error was detected.

Shadow RAM Passed - This message will be displayed to indicate that the shadow memory test is occurring. The amount of RAM tested (in kbytes) will be displayed before this message. Not all systems contain shadow memory.

System CMOS checksum bad - Run SETUP - This error will be displayed if the CMOS has been modified incorrectly. This can be caused by a software program which modifies the CMOS memory or if the CMOS becomes corrupted. Run the SETUP program to reconfigure the system.

System battery is dead - Replace and run SETUP - This error will be displayed if the CMOS clock battery indicator shows that the battery is dead. You should contact CDI.

System RAM Failed at offset: - This error will be displayed in the event that a RAM error is detected in the system memory. Above this message, the k address which failed will be displayed. This message will show the offset in the 64k block at which the error was detected.

System RAM Passed - This message will be displayed to indicate that the system memory test is occurring. The amount of RAM tested (in kbytes) will be displayed before this message.

System timer error - This error will be displayed if the system timer test fails. Your system board needs to be repaired.

5.6 Boot-Up Display Screen

During Boot-Up, a full gray screen with a spinning star in the lower right-hand corner will be displayed. Across the bottom of the screen will be the following message printed in blue:

Press <ESC> to exit, or <F2> to enter SETUP

Pressing <ESC> will bypass the SETUP program and attempt to load the values and special features currently enabled in the SETUP program.

Pressing <F2> will clear the Gray screen to briefly display the DOS boot sequence and quickly bring the user into the Main Menu of the SETUP program. Within the Setup program, the end user can modify BIOS settings and control the special features of the system. The Setup program uses a number of menus for making changes and turning the special features on or off. The menus shown here are from a typical system, but the actual features displayed depend on the hardware installed in the system.

5.7 The Main Menu

The Main menu is displayed as follows:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.				
Main	Advanced	Exit		
System Time:	[14:37:12]		Item Specific Help	
System Date:	[11/13/1995]		<Tab>, <Shift-Tab>, or <Enter> selects field.	
Diskette A:	[1.44 MB, 3 1/2"]			
Diskette B:	[Not Installed]			
▶ IDE Adapter 0 Master:	(C: 213Mb)			
▶ IDE Adapter 1 Slave:	(None)			
Video System:	[EGA / VGA]			
▶ Memory Shadow:				
▶ Boot Sequence:	[A: then C:]			
▶ Numlock:	[On]			
System Memory:	640 KB			
Extended Memory:	7168 KB			
F1 Help	↑↓ Select Item	-/+ Change Values	F9 Setup Defaults	
ESC Exit	↔ Select Menu	Enter Select Sub-Menu	F10 Previous Values	

Note:

All menus contain extensive on-line Help which is typically displayed on the right side of the screen. Using this on-line help feature makes configuration of your system very easy.

5.7.1 The Menu Bar

The Menu bar at the top of the window lists these selections:

Main	Use this menu for basic system configuration.
Advanced	Use this menu to set the Advanced Features available on your system's chipset.
Exit	Exits the current menu.

Use the left/right ← → arrow keys to make a selection.

5.7.2 The Legend Bar

Use the keys listed in the legend bar on the bottom to make your selections or exit the current menu. The following chart describes the legend keys and their alternates:

Key	Function
<F1> or <Alt-H>	General Help window
<ESC>	Exit this menu.
→ or Ⓜ arrow keys	Select a different menu.
- or ← arrow keys	Move cursor up and down.
<Tab> or <Shift-Tab>	Cycle cursor up and down.
<Home> or <End>	Move cursor to top or bottom of window.
<PgUp> or <PgDn>	Move cursor to next or previous page.
<F5> or <->	Select the Previous Value for the field.
<F6> or <+> or <Space>	Select the Next Value for the field.
<F9>	Load the Default Configuration values for this menu.
<F10>	Load the Previous Configuration values for this menu.
<Enter>	Execute Command or Select P Submenu.
<Alt-R>	Refresh screen.

To select an item, use the arrow keys to move the cursor to the field you want. Then use the value keys to cycle through the values for that field. The Save Values commands in the Exit menu saves the values currently displayed in all the menus.

To display a sub menu, use the arrow keys to move the cursor to the sub menu you want. Then press **<Enter>**.

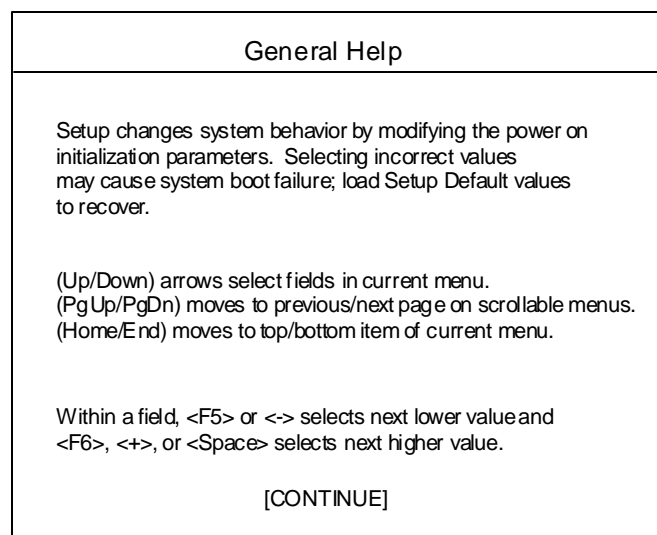
A pointer marks all sub menus.

5.7.3 The Field Help Window

The help window on the right side of each menu displays the help text for the currently selected field. It updates as you move the cursor to each field.

5.7.4 The General Help Window

Pressing <F1> or <Alt-H> on any menu brings up the General Help window that describes the legend keys and their alternates as shown here:



The scroll bar on the right of any window indicates that there is more than one page of information in the window. Use <PgUp> and <PgDn> to display all the pages. Pressing <Home> and <End> displays the first and last page. Pressing <Enter> displays each page and then exits the window. Press <ESC> to exit the current window.

5.7.5 Main Menu Selections

You can make the following selections directly at the Main menu itself. Use the sub menus for other selections.

Feature	Options	Description
System Time	HH:MM:SS	Set the system time
System Date	MM/DD/YYYY	Set the system date
Diskette A: Diskette B:	360kbytes, 5 1/4" 1.2Mbytes, 5 1/4" 720kbytes, 3 1/2" 1.44M, 3 1/2" 2.88Mbytes, 3 1/2" Not installed	Select the type of floppy-disk drive installed in your system
Video System	Monochrome EGA/VGA, CGA 80x25,	Select the default video device
Boot Sequence	A: then C: C: then A: C: only	Select the order in which to search for a boot disk.
Numlock	On Off Auto	Select the power-on state for the numlock feature.
System Memory	N/A	Displays amount of conventional memory detected during bootup
Extended Memory	N/A	Displays the amount of extended memory detected during bootup

5.7.6 IDE Adapters

The IDE adapters control the hard disk drives. *Phoenix*BIOS 4.05 supports up to two IDE hard disk drives in the following combinations:

- 1 Master
- 1 Master, 1 Slave

Use a separate sub menu to configure each hard-disk drive. Typically, the user will select a single hard-disk drive which is designated IDE adapter 0 Master.

To setup your IDE drive highlight the IDE adapter 0 Master submenu and hit Enter. The following menu will be displayed if the IDE adapter 0 Master has not yet been installed:

Phoenix BIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Main			
IDE Adapter 0 Master (None)		Item Specific Help	
Autotype Fixed Disk:	[Press Enter]	Attempts to automatically detect the drive type for drives that comply with ANSI specifications.	
Type:	[None]		
Cylinders:			
Heads:			
Sectors/Track:			
Write Precomp:			
Multi-Sector Transfers:	[Disabled]		
LBA Mode Control:	[Disabled]		
F1 Help	↓ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	↔ Select Menu	Enter Select Sub-Menu	F10 Previous Values

Use the legend keys listed on the bottom to make your selections and exit to the Main menu. Use the following chart to configure the hard disks.

Feature	Options	Description
Autotype Fixed Disk	N/A	Pressing <Enter> at this field attempts to read the hard disk parameters from the drive itself and sets the following options to their optimum settings.
Type	1 to 39 User	1 to 39 fills in all remaining fields with values for predefined disk type. "User" prompts user to fill in remaining fields.
Cylinders	1 to 65,536	Number of cylinders
Heads	1 to 16	Number of read/write heads
Sectors/Track	1 to 63	Number of sectors per track
Multi-Sector Transfers	Standard 2 sectors 4 sectors 8 sectors 16 sectors MAX n	Standard is 1 sector per block. If Block PIO is installed, MAX n indicates the largest sector possible (n = 32, 64 or 128 blocks). The MAX option is not always the fastest.
LBA mode Control	Enabled Disabled	Enables Logical Block Access. Default is Disabled.
32-Bit I/O	Enabled Disabled	Enables 32-Bit communication between the CPU and the IDE disk.
Transfer Mode	Standard FAST PIO 1 FAST PIO 2 FAST PIO 3	Selects the method for transferring the data between the hard disk and system memory. The SETUP menu only lists those options supported by the drive and platform.

* IDE drives do not require setting landing zone and write precomp.

WARNING

Incorrect settings can cause your system to malfunction.

The largest IDE hard drive *Phoenix*BIOS 4.05 supports is determined by:

$$(\# \text{ of sectors}) \times (\# \text{ of cylinders}) \times (\# \text{ of heads}) \times (128 \text{ Kbytes/sector})$$

$$(63) \times (65,536) \times (16) \times (128 \text{ K}) = 8.4 \text{ Gb}$$

As an example, using a Conner #CFS210A IDE hard drive connected up to the IDE port and highlighting "Autotype Fixed Disk" and pressing "ENTER" automatically reads the hard drive and sets the parameters as follows:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Main			
IDE Adapter 0 Master (C: 213Mb)		Item Specific Help	
Autotype Fixed Disk:	[Press Enter]	Attempts to automatically detect the drive type for drives that comply with ANSI specifications.	
Type:	[User] 213 Mb		
Cylinders:	[685]		
Heads:	[16]		
Sectors/Track:	[38]		
Write Precomp:	[None]		
Multi-Sector Transfers:	[16 Sectors]		
LBA Mode Control:	[Disabled]		
F1 Help	↕ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	↔ Select Menu	Enter Select Sub-Menu	F10 Previous Values

Note:

You can manually enter the Type, Cylinders, Heads and Sectors/Track information if you have an older vintage drive which does not support Autotyping.

5.7.7 Memory Shadow

Highlighting and entering "Memory Shadow" from the Main menu displays a menu like the one shown here. Use the legend keys to make your selections and exit to the Main menu. Defaults are shown:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Main			
Memory Shadow		Item Specific Help	
System Shadow:	[Enabled]	(Messages differ as the user scrolls through the options)	
Video Shadow:	[Enabled]		
Shadow Memory Regions			
C800 - CFFF:	[Disabled]		
E000 - EFFF:	[Disabled]		
F1 Help	↑↓ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	← Select Menu	Enter Select Sub-Menu	F10 Previous Values

WARNING

Incorrect settings can cause your system to malfunction.

5.7.8 Boot Sequence

Highlighting and entering "Boot Sequence" from the Main menu displays a menu like the one shown here. Use the legend keys to make your selections and exit to the Main menu. Default values are shown:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Main			
Boot Options		Item Specific Help	
Boot Sequence:	[A:then C:]	(Messages differ as the user scrolls through the options)	
Setup Prompt:	[Enabled]		
POST Errors:	[Enabled]		
Floppy Check:	[Enabled]		
Summary Screen:	[Disabled]		
F1 Help	↑↓ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	← Select Menu	Enter Select Sub-Menu	F10 Previous Values

Use the legend keys listed on the bottom to make your selections and exit to the Main menu. Use the following chart to configure the Boot Sequence:

Feature	Options	Description
Boot Sequence	A: then C: C: then A: C: only	The BIOS attempts to load the operating system from the disk drives in the sequence selected here.
SETUP Prompt	Enabled Disabled	Displays "Press <ESC> to exit or <F2> for Setup" during boot-up.
POST Errors	Enabled Disabled	At boot-up error, pauses and displays "Press <F1> to resume, <F2> to Setup"
Floppy Check	Enabled Disabled	Seeks diskette drives during boot-up. Disabling speeds boot time.
Summary Screen	Enabled Disabled	Displays system summary screen during boot-up.

5.7.9 Numlock

Highlighting and entering "Numlock" from the Main menu displays a menu like the one shown here. Use the legend keys to make your selections and exit to the Main menu. Default values are shown:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Main			
Keyboard Features		Item Specific Help	
Numlock:	[On]	(Messages differ as the user scrolls through the options)	
Key Click:	[Disabled]		
Keyboard auto-repeat rate:	[30/sec]		
Keyboard auto-repeat delay:	[1/2 sec]		
F1 Help	↑↓ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	← Select Menu	Enter Select Sub-Menu	F10 Previous Values

Use the legend keys listed on the bottom to make your selections and exit to the Main menu. Use the following chart to configure the Boot Sequence:

Feature	Options	Description
Numlock	Auto On Off	Turns Numlock ON or OFF during boot-up if it finds a numeric keypad.
Key Click	Enabled Disabled	Turns audible key click ON or OFF.
Keyboard auto-repeat rate	2/sec 6/sec 10/sec 13.3/sec 18.5/sec 21.8/sec 26.7/sec 30/sec	Sets the number of times per second to repeat a keystroke when you hold the key down.
Keyboard auto-repeat delay	1/4 sec 1/2 sec 3/4 sec 1 sec	Sets the delay time after the key is held down before it begins to repeat the keystroke.

5.8 The Advanced Menu

Highlighting the "Advanced" from menu bar on the Main menu displays a menu like this:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Main Advanced Exit			
<p style="text-align: center;">Warning!</p> <p style="text-align: center;">Setting items on this menu to incorrect values may cause your system to malfunction.</p> <p>▶ Integrated Peripherals</p> <p>▶ Advanced Chipset Control</p> <p>Large Disk Access Mode: [DOS]</p>		Item Specific Help	
F1 Help	↑↓ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	↔ Select Menu	Enter Select Sub-Menu	F10 Previous Values

5.8.1 Integrated Peripherals

Highlighting and selecting "Integrated Peripherals" from the menu displays the following screen:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Advanced			
Integrated Peripherals		Item Specific Help	
<p>COM Port [3F8, IRQ4]</p> <p>COM Port [2F8, IRQ3]</p> <p>LPT Port: [378, IRQ7]</p> <p>LPT Mode: [Output Only]</p> <p>Diskette Controller: [Enabled]</p> <p>IDE Controller: [Enabled]</p>		(Messages differ dependent upon selection made)	
F1 Help	↑↓ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	↔ Select Menu	Enter Select Sub-Menu	F10 Previous Values

5.8.2 Menu Selections

You can make the following selections at the Integrated Peripherals menu .

Feature	Options	Description
COM Ports	3F8, IRQ4 2F8, IRQ3 338, IRQ4 238, IRQ3 3E8, IRQ4 2E8, IRQ3 2E8, IRQ4 2E0, IRQ3 220, IRQ4 228, IRQ3 Auto Disabled	Set the port address and interrupt, if desired.
LPT Port	378, IRQ7 278, IRQ7 Auto Disabled	Set the port address and interrupt, if desired.
LPT Mode	Output Only Bi-Directional	Set the LPT Mode.
Diskette Controller	Enabled Disabled	Turn ON or OFF the Floppy Drive port.
IDE Adapter	Enabled Disabled	Turn ON or OFF the Hard Drive port.

5.8.3 Advanced Chipset Control

The chipset is an integrated circuit that acts as an interface between the CPU and much of the system's hardware. You can use this menu to change the values in the chipset registers and optimize your system's performance.

Selecting "Advanced Chipset Control" from the menu displays the following screen:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Advanced			
Advanced Chipset Control		Item Specific Help	
Slow Refresh:	[Disabled]	(Messages differ dependent upon selection made)	
RAS Precharge:	[2T]		
RAS to CAS Delay:	[2T]		
CAS Precharge:	[1T]		
CAS Read Width:	[1T]		
CAS Write Width:	[1T]		
/ADS Delay:	[Enabled]		
DRAM 0 Wait State:	[Disabled]		
Advanced IDE Drives Timing:	[Master]		
F1 Help	↵ Select Item	-/+ Change Values	F9 Setup Defaults
ESC Exit	↔ Select Menu	Enter Select Sub-Menu	F10 Previous Values

5.8.4 Menu Selections

You can make the following selections at the Advanced Chipset Control menu:

Feature	Options	Description
Slow Refresh	Disabled 30 us 60 us 120 us	Selects Refresh interval
RAS Precharge	2T 3T 4T 5T	Selects RAS Precharge time for the Write cycle.
RAS to CAS Delay	2T 3T	Selects RAS to CAS delay time for the Write cycle.
CAS Precharge	1T 2T	Selects CAS Precharge time for the Write cycle.
CAS Read Width	1T 2T 3T 4T	Selects CAS width for the Read cycle.,
CAS Write Width	1T 2T 3T 4T	Selects CAS width for the Write cycle.
/ADS Delay	Enabled Disabled	Selects /ADS delay at the front end.
DRAM 0 Wait State	Enabled Disabled	Selects DRAM Wait State.
Advanced IDE Drives Timing	Disabled Master Slave Both	Advanced IDE Drive timing for ISA devices.

5.8.5 Large Disk Access Mode

Selecting "Large Disk Access Mode" from the menu will allow the user to select either "DOS" or "OTHER" when highlighting this feature. Most often times the user will want to select "DOS".

5.9 The Exit Menu

Selecting "Exit" from the menu bar displays this menu:

PhoenixBIOS Setup - Copyright 1985-95 Phoenix Technologies, Ltd.			
Main	Advanced	Exit	
Save Changes & Exit Exit Without Saving Changes Get Default Values Load Previous Values Save Changes		Item Specific Help	(Messages differ dependent upon selections made)
F1 Help ESC Exit	↓ Select Item ↔ Select Menu	-/+ Change Values Enter Select Sub-Menu	F9 Setup Defaults F10 Previous Values

The following sections describe each of the options on this menu. Note that <ESC> does not exit this menu. You must select one of the items from the menu or menu bar to exit.

5.9.1 Save Changes & Exit

After making your selections on the Setup menus, always select Save Changes in order to make them operative. Settings are stored by an on-board battery and stays on after you turn your system off. After you save your selections, the program displays this message:

NOTICE
Changes Have Been Saved
[CONTINUE]

Hitting ENTER causes the system to re-boot and enable your new SETUP changes.

If you attempt to exit without saving, the program asks if you want to save before exiting. During bootup, *PhoenixBIOS* attempts to load the values you saved in NVRAM.

5.9.2 Exit Without Saving Changes

Use this option to exit Setup without recording any changes you may have made.

5.9.3 Get Default Values

To load all the default Setup values in the Setup menus, select Load ROM Default Values from the Main menu. The program displays this message:

```
ROM Default values have been loaded!  
Press <space> to continue
```

If, during bootup, the BIOS program detects a problem in the integrity of values stored in NVRAM, it displays these messages:

```
System CMOS checksum bad - run SETUP  
Press <F1> to resume, <F2> to Setup
```

The CMOS values have been corrupted or modified incorrectly, perhaps by an application program that changes data stored in CMOS. Press <F1> to resume the boot or <F2> to run Setup with the ROM default values already loaded into the menus. You can make other changes before saving the values to NVRAM.

5.9.4 Load Previous Values

If, during a Setup session, you change your mind about your selections and have not yet saved the values to NVRAM, you can restore the values you previously saved to NVRAM.

Selecting Load Previous Values on the Exit menu updates all the selections and displays this message:

```
CMOS values have been loaded!  
Press <space> to continue
```

5.9.5 Save Current Values

Saves all the selections without exiting Setup. You can return to the other menus if you want to review and change your selections.

5.10 BIOS Programmer's Interface

The following information describes all of the services available through function calls to the BIOS. Typically, the function number is placed into the AH register, and the appropriate interrupt call is made. All other sending register values and resultant values depend on the BIOS operation. All registers are preserved unless they are specifically used to return data or status.

5.11 Interrupt 10H Functions - Video Services

The INT 10 software interrupt will handle all standard video functions. The following outlines the video services in more detail with their corresponding register entry values:

AH = 00 **Set video mode**
 AL Mode value - Standard VGA Modes - see table below

Mode# Hex	Type	Colors	Alpha	Resolution	Font	Clock (MHz)	HSync (KHz)	VSync (Hz)	Memory (Min.)	Buffer Start	Page
0,1	A/N	16/256K	40x25	320x200	8x8	25.175	31.55	70.3	256K	B8000	8
0,1*	A/N	16/256K	40x25	320x350	8X14	25.175	31.55	70.3	256K	B8000	8
0,1**	A/N	16/256K	40x25	360x400	9x16	28.322	31.34	69.8	256K	B8000	8
2,3	A/N	16/256K	80x25	640x200	8x8	25.175	31.55	70.3	256K	B8000	8
2,3*	A/N	16/256K	80x25	640x350	8X14	25.175	31.55	70.3	256K	B8000	8
2,3**	A/N	16/256K	80x25	720x400	9x16	28.322	31.34	69.8	256K	B8000	8
4,5	APA	4/256K	40x25	320x200	8x8	25.175	31.55	70.3	256K	B8000	1
6	APA	2/256K	80x25	640x200	8x8	25.175	31.55	70.3	256K	B8000	1
7	A/N	MONO	80x25	720x350	9x14	28.322	31.34	69.8	256K	B8000	8
7*	A/N	MONO	80x25	720x400	9x16	28.322	31.34	69.8	256K	B8000	8
D	APA	16/256	40x25	320x200	8x8	25.175	31.55	70.3	256K	A0000	8
E	APA	16/256K	80x25	640x200	8x8	25.175	31.55	70.3	256K	A0000	4
F	APA	MONO	80x25	640x350	8X14	25.175	31.55	70.3	256K	A0000	2
10	APA	16/256K	80x25	640x350	8X14	25.175	31.55	70.3	256K	A0000	2
11	APA	2/256K	80x30	640x480	8x16	25.175	31.55	60.1	256K	A0000	1
12	APA	16/256K	80x30	640x480	8x16	25.175	31.55	60.1	256K	A0000	1
13	APA	256/256K	40x25	320x200	8x8	25.175	31.55	70.3	256K	A0000	1

Note:

For Modes 3 and 7, 8-dot fonts are used when driving a panel.

AH = 01 **Set cursor type (start and end)**

CL Cursor end line
 CH Cursor start line

AH = 02 **Set cursor position**

BH Page to set cursor
 DL Character column position
 DH Character row position

AH = 03 **Get cursor position of page**

BH	Page to return cursor
Exit:	
DL	Character column position
DH	Character row position
CL	Cursor end line
CH	Cursor start line
AH = 04	Return light pen position
Exit:	
DL	Char column of light pen
DH	Char row of light pen
CH	Raster line
BX	Pixel column
AH	Light pen trigger (1 pressed)
AH = 05	Change displayed (active) page
AL	Page number to display
AH = 06	Scroll active page up
CL	Column of scroll upper left
CH	Row of scroll upper left
DL	Column of scroll lower right
DH	Row of scroll lower right
BH	Attribute for blanked space
AL	Number of lines to scroll
AH = 07	Scroll active page down
CL	Column of scroll upper left
CH	Row of scroll upper left
DL	Column of scroll lower right
DH	Row of scroll lower right
BH	Attribute for blanked space
AL	Number of lines to scroll
AH = 08	Read character and attribute
BH	Video page to read character
Exit:	
AL	Character
AH = 09	Write character and attribute
AL	Character to write
BL	Character attribute (alpha)
	Character color (graphics)
BH	Page to write character
CX	Count of characters to write
AH = 0A	Write character at cursor
BH	Page to write character
AL	Character to write
CX	Count of characters to write
AH = 0B	Write color palette
BL	Color value for palette
BH	Palette color ID (01)

AH = 0C Write graphics pixel

AL Color value for pixel
(XORed if bit7=1)
CX Column to write pixel
DX Row to write pixel

AH = 0D Read graphics pixel

CX Column to read pixel
DX Row to read pixel
Exit:
AL Value of pixel read

AH = 0E Teletype write character

AL Character to write
BL Foreground color (graphics only)

AH = 0F Return Current Video Parameters

Exit:
AL Current video mode
AH Number of character columns
BH Active page

AH = 13 Write string

ES:BP Pointer to string
CX Length of string to display
DH Character row for display
DL Character column for display
BL Display attribute
AL Write string mode
 0 Chars only, no cursor update
 1 Chars only, update cursor
 2 Char, Attrib, no cursor update
 3 Char, Attrib, update cursor

5.11.1 Interrupt 10H Functions - Western Digital Paradise Extensions

In addition to the standard Video Functions, the Computer Dynamics SBC family provides Western Digital Paradise Super VGA Enhancements via INT 10 BIOS calls.

AH = 00 Set video mode

AL = Western Digital Mode number, per tables below

VESA VBE-Compliant Super VGA Modes

VESA Mode# (Hex)	WD Mode	Type	Colors	Alpha	Resolution	Font	Memory (Min.)	Buffer Start	Page
101	5F	APA	256/256K	80x30	640x480	8x16	512K	A0000	1
102	58/6A	APA	16/256K	100x75	800x600	8x8	256K	A0000	1
103	5C	APA	256/256K	700x75	800x600	8x8	512K	A0000	1
104	5D	APA	16/256K	128x48	1024x768	8x16	512K	A0000	1
105	60	APA	256/256K	128x48	1024x768	8x16	1M	A0000	1
109	55	A/N	16/256K	132x25	1056x400	8x16	256K	B8000	4
10A	54	A/N	16/256K	132x43	1056x344	9x9	256K	B8000	2
10D ⁽¹⁾	68	APA	32,768	40x25	320x200	8x8	256K	A0000	1
10E ⁽¹⁾	78	APA	65,536	40x25	320x200	8x8	256K	A0000	1
110 ⁽¹⁾	62	APA	32,768	80x30	640x480	8x16	1M	A0000	1
111 ⁽¹⁾	72	APA	65,536	80x30	640x480	8x16	1M	A0000	1

(1) These modes are only supported by Rev J or later SBC.

132 Column Modes (Supported on CRT only)

Mode# (Hex)	Type	Colors	Alpha	Resolution	Font	Clock (MHz)	HSync (KHz)	VSync (Hz)	Memory (Min.)	Buffer Start	Page
54	A/N	16/256K	132x43	1056x344	9x9	44.9	31.06-	69.2+	256K	B8000	2
55	A/N	16/256K	132x25	1056x400	8x16	44.9	31.06-	69.2+	256K	B8000	4

640x480x256 Color Modes

Mode# (Hex)	Type	Colors	Alpha	Resolution	Font	Clock (MHz)	HSync (KHz)	VSync (Hz)	Memory (Min.)	Buffer Start	Page
5F	APA	256/256K	80x30	640x480	8x16	25.175	31.55-	60.1-	512K	A0000	1

800x600 modes

Mode# (Hex)	Type	Colors	Alpha	Resolution	Font	Clock (MHz)	HSync (KHz)	VSync (Hz)	Memory (Min.)	Buffer Start	Page
58/6A	APA	16/256K	100x75	800x600	8x8	36.0	35.12-	56.2-	256K	A0000	1
58/6A	APA	16/256K	100x75	800x600	8x8	40.0	37.84+	60.3+	256K	A0000	1
58/6A	APA	16/256K	100x75	800x600	8x8	50.0	47.67+	71.7+	256K	A0000	1
5C	APA	256/256K	100x75	800x600	8x8	36.0	35.12-	56.2-	512K	A0000	1
5C	APA	256/256K	100x75	800x600	8x8	40.0	37.84+	60.3+	512K	A0000	1
5C	APA	256/256K	100x75	800x600	8x8	50.0	47.67+	71.8+	1M	A0000	1

1024x768 modes

Mode# (Hex)	Type	Colors	Alpha	Resolution	Font	Clock (MHz)	HSync (KHz)	VSync (Hz)	Memory (Min.)	Buffer Start	Page
5D	APA	16/256K	128X48	1024X768	8X16	44.9	35.6+	87.1+	512K	A0000	1
5D	APA	16/256K	128X48	1024X768	8X16	65.0	48.37-	60.0-	512K	A0000	1
60 ⁽¹⁾	APA	256/256K	128X48	1024X768	8X16	44.9	35.6+	87.1+	1M	A0000	1
60 ⁽¹⁾	APA	256/256K	128X48	1024X768	8X16	65.0	48.37-	60.0-	1M	A0000	1

(1) This mode has an alternate mode number of 30 for use in environments which have a conflicting Mode 60.

32K Color modes

Mode# (Hex)	Type	Colors	Alpha	Resolution	Font	Clock (MHz)	HSync (KHz)	VSync (Hz)	Memory (Min.)	Buffer Start	Page
62 ^(1,2)	APA	32,768	80x30	640x480	8x16	25.2	31.50-	70.29	1M	A0000	1
68 ⁽²⁾	APA	32,768	40x25	320x200	8x8	25.2	31.60-	70.2+	256K	A0000	1

(1) This mode has an alternate mode number of 32 for use in environments which have a conflicting Mode 62.

(2) These modes are only supported by the Rev J and later SBC.

64K Color modes

Mode# (Hex)	Type	Colors	Alpha	Resolution	Font	Clock (MHz)	HSync (KHz)	VSync (Hz)	Memory (Min.)	Buffer Start	Page
72 ⁽¹⁾	APA	65,536	80x30	640x480	8x16	25.2	31.50-	70.29	1M	A0000	1
78 ⁽¹⁾	APA	65,536	40x25	320x200	8x8	25.2	31.60-	70.2+	256K	A0000	1

(1) These modes are only supported by the Rev J and later SBC.

Additional Video services are supported via a "Gateway Function" (identified by ax=7F7FH).

Note:

All numbers shown in the BIOS calls are HEX.

AX = 7F7F

BH = 2 Return Special Status Info

Returns: CH = Total display memory (VRAM) in 64K units

CL = Display memory (VRAM) unused

AX = 7F7F

BX = 4101 Toggle Auto-Centering ON/OFF

Note:

Not available when Vertical Expansion is enabled. Vertical Expansion is the default setting for most SBC flat panel products with 480 vertical lines or greater. Some panels less than 480 lines high are implemented with Vertical Expansion disabled.

AX = 7F7F

BX = 4102 Toggle Vertical Expansion ON/OFF

Note:

Text modes and graphics modes use different hardware algorithms for expanding to fill the screens. Text modes use "enhanced vertical expansion" which only expands the lines between the text characters and so the text characters do not look choppy like most flat panel displays.

AX = 7F7F

BX = 4103 Toggle Enhanced Vertical Expansion Type

Types available: Expand three lines below each text row.

Expand one line above and two lines below each text row.

Note:

Enhanced Vertical expansion is only available in text modes. A different hardware algorithm is used in graphics modes.

AX = 7F7F

BX = 4104 Three Way Display Toggle: Flat Panel/CRT/Simultaneous

Note:

During Three-way Display Switching, the switch from CRT-only to Simultaneous is inhibited if the current Video Mode is a high-resolution mode which is not supported on the flat panel.

AX = 7F7F

BX = 4105 Switch to CRT

Note:

This BIOS call may clear the video memory when using a split screen (dual screen) panel.

AX = 7F7F

BX = 4106 Switch to LCD

Note:

This BIOS call may clear the video memory when using a split screen (dual screen) panel.

AX = 7F7F

BX = 4107 Toggle Between LCD/CRT; No Mode Set (VGA Mode Only)

AX = 7F7F

BX = 4108 Toggle Between Simultaneous/Non-Simultaneous Display (VGA Mode Only)

AX = 7F7F

BX = 410A Toggle between Normal/Reverse Video for Text/Graphics Modes.

Note:

This toggling feature is for MONO LCD only and has the following sequence: Text Reverse and Graphics Normal, then Text Normal and Graphics Normal, then Text Reverse and Graphics Reverse.

AX = 7F7F
BX = 410B **Toggle between 32-Bit and 16-Bit Memory Data Path.**

AX = 7F7F
BX = 410D **Switch To Simultaneous mode.**

Note:

This BIOS call may clear the video memory when using a split screen (dual screen) panel.

AX = 7F7F
BX = 4200 **Get Extended Information**
Returns: BX = Status Information
CX = Status Information (see below)

STATUS BITS	BX	CX
0	00 = No CRT Attached 01 = Color CRT Attached 10 = Mono CRT Attached 11 = Reserved	Reserved
1		
2	0 = Display is CRT 1 = Display is LCD	0 = 32-bit memory data path 1 = 16-bit memory data path
3	0 = Auto Center is On 1 = Auto Center is Off	Reserved
4	00 = Graphics Normal - Text Reverse 01 = Graphics Normal - Text Normal 10 = Graphics Reverse - Text Reverse 11 = Reserved	
5		Memory Configuration 000 = Reserved 001 = Reserved 100 = 256Kx16x2 101 = 256Kx16x1 111 = Reserved
6	0 = Vertical Expansion is On 1 = Vertical Expansion is Off	
7	Reserved	
8	00 = 800x600 @ 56Hz 01 = 800x600 @ 60Hz 10 = 800x600 @ 72Hz 11 = 800x600x16 @ 72Hz & 800x600x256 @ 72Hz	00 = Dual Panel 01 = Plasma Panel 10 = Reserved
9		
10	00 = 1024x768x16 Interlaced 01 = 1024x768x16 @ 60Hz 10 = 1024x768x16 @ 70Hz 11 = Reserved	Reserved
11		
12	00 = 1024x768x256 Interlaced 01 = 1024x768x256 @ 60Hz 10 = 1024x768x256 @ 70Hz 11 = Reserved	
13		0 = AT Bus 1 = Local Bus
14	0 = Non-simultaneous Display 1 = Simultaneous Display	Reserved
15	Reserved	0 = 90c24 1 = 90c24A

AX = 7F7F

BX = 4201 Set Extended Information

CH = Status Information

STATUS BITS	CH
0	00 = 800x600 @ 56Hz 01 = 800x600 @ 60Hz 10 = 800x600 @ 72Hz 11 = 800x600x16 @ 72Hz & 800x600x256 @ 72Hz
1	
2	00 = 1024x768x16 Interlaced 01 = 1024x768x16 @ 60Hz 10 = 1024x768x16 @ 70Hz 11 = 1024x768x16 @ 72Hz
3	
4	00 = 1024x768x256 Interlaced 01 = 1024x768x256 @ 60Hz 10 = 1024x768x256 @ 70Hz 11 = 1024x768x256 @ 72Hz
5	
6	Reserved
7	

AX = 7F7F

BX = 4700 Do Monitor Detection

Returns: BL = 0 - No Monitor
BL = 1 - Color Monitor
BL = 2 - Mono Monitor

AX = 7F7F

BH = 48 Set Power Down Status

BL = 0 - System Power-Down Mode

DRAM refresh option
CH = 0 - DRAM self refresh
CH = 1 - CAS before RAS refresh
CH = 2 - CKIN / PR71
CH = 3 - 32 KHz pass-through

BL = 1 - Display Idle Mode

DRAM refresh option
CH = 0 - DRAM self refresh
CH = 1 - CAS before RAS refresh

BL = 2 - General Power-Down Mode

CL = EXT Clock Divide Value

BL = 3 - General Power-Down Mode

CL = INT Clock Divide Value

BL = 4 - Deep Sleep Mode (save registers)

CH = DRAM refresh option
ES:DX = 1020 byte buffer

BL = 5 - Deep Sleep Mode (restore registers)

ES:DX = 1020 byte buffer

BL = 6 - Save Display Memory

CX = 64K display memory bank

BL = 7 - Restore Display Memory

CX = 64K display memory bank

AX = 7F7F

BH = 49 To-Suspend

AX = 7F7F

BH = 4A From-Suspend

AX = 7F7F

BH = 60 Return BIOS Status Info

Returns: AX = Offset to date_time
CX = offset to part_no/version
BP = Segment of BIOS
DI = Offset to BIOS Internal Tables

5.11.2 Interrupt 10H Functions - VESA VBE Compliant SVGA Extensions

In addition to the standard Video Functions and the Western Digital Paradise Video Extensions, the Computer Dynamics SBC family supports a subset of the VESA[®] Super VGA BIOS Extensions (VBE Standard VS911022 Version 1.2) in the BIOS ROM itself. No additional VESA device driver is required in the config.sys file.

The Video Electronics Standards Association (VESA[®]) is the international, non-profit organization which sets and supports industry-wide video graphics standards for the benefit of the end user. VESA's membership includes manufacturers of monitors, peripherals, systems and software products. The VESA[®] offices are located at 2150 North First Street, Suite 440, San Jose, California 95131-2029, and can be reached at 408-435-0333, FAX 408-435-8225.

Note:

The following information is per the VESA Super VGA Standard #VS911022 - VBE Version 1.2. For more detailed information than is provided below, contact the Video Electronics Standards Association at the address or phone numbers listed above.

The VESA VBE-Compliant Super VGA modes are listed below for reference. This is the same table that is found in the Western Digital Extended section of this manual. It is duplicated here for convenience.

VESA VBE-Compliant Super VGA Modes

VESA Mode# (Hex)	Equivalent WD Mode	Type	Colors	Alpha	Resolution	Font	Memory (Min.)	Buffer Start	Page
101	5F	APA	256/256K	80x30	640x480	8x16	512K	A0000	1
102	58/6A	APA	16/256K	100x75	800x600	8x8	256K	A0000	1
103	5C	APA	256/256K	700x75	800x600	8x8	512K	A0000	1
104	5D	APA	16/256K	128x48	1024x768	8x16	512K	A0000	1
105	60	APA	256/256K	128x48	1024x768	8x16	1M	A0000	1
109	55	A/N	16/256K	132x25	1056x400	8x16	256K	B8000	4
10A	54	A/N	16/256K	132x43	1056x344	9x9	256K	B8000	2
10D ⁽¹⁾	68	APA	32,768	40x25	320x200	8x8	256K	A0000	1
10E ⁽¹⁾	78	APA	65,536	40x25	320x200	8x8	256K	A0000	1
110 ⁽¹⁾	62	APA	32,768	80x30	640x480	8x16	1M	A0000	1
111 ⁽¹⁾	72	APA	65,536	80x30	640x480	8x16	1M	A0000	1

(1) These modes are only supported by Rev J or later SBC.

The subset of VESA® Super VGA compliant calls that are supported directly from the BIOS ROMs provided with the SBC are listed below. The VESA call is made via the VESA video extension function 4Fh (AH=4Fh) in the standard INT 10 video BIOS call.

INT 10 Function 4FH Sub-Function	Description
00H	Return Super VGA Information
01H	Return Super VGA Mode Information
02H	Set Super VGA Video Mode
03H	Return Current Video Mode
04H	Save/Restore Super VGA State
05H	CPU Video Memory Window Control

VESA VBE Return Information:

All VBE functions return status information in the AX register. The format is as follows:

AL == 4F	Function is supported by current version
AL != 4F	Function is not supported by current version
AH == 00	Function call successful
AH == 01	Function call failed

Software should treat any non-zero value in AH as a failure condition as later versions of the VBE could define additional error codes.

VESA VBE Super VGA BIOS Calls

AH = 4F

AL = 00 Return Super VGA Information

ES:DI = Pointer to 256 byte buffer with format specified below.

Return:

AX = Status

All other registers are preserved

```
VgaInfoBlock struct
  VESASignature db 'VESA'          ; 4 signature bytes
  VESAVersion   dw ?                ; VESA Version number
                                   ; Hi Byte = Major # / Lo Byte = Minor #
  OEMStringPtr  dd ?                ; Pointer to OEM String
                                   ; Far Ptr to a NULL terminated string
  Capabilities  db 4 dup(?)         ; capabilities of the video environment
                                   ; Bit 0 = DAC Switchable ?
                                   ; 0=DAC width is fixed @ 6 bits per
primary color                                     ; 1=DAC width is switchable
                                   ; Bits 1-31 = Reserved for future expansion
  VideoModePtr  dd ?                ; pointer to supported Super VGA Modes
```

```

; List of supported SVGA mode numbers. Each
mode is one word
; in length. The list is -1
terminated(0FFFFh).
TotalMemory dw ? ; Number of 64kbytes memory blocks installed
Reserved db 236 dup(?) ; remainder of VgaInfoBlock
VgaInfoBlock ends

```

AH = 4F

AL = 01 Return Super VGA Mode Specific Information

CX = Super VGA Video Mode (Must be a valid mode that was returned by function 00)

ES:DI = Pointer to 256 byte buffer

Return:

AX = Status

All other registers are preserved

The buffer pointed to by ES:DI has the following format:

```

ModeInfoBlock struct
  ModeAttributes dw ? ; mode attributes
; Bit 0 = Mode Supported in current hardware
configuration
; 0 = not supported
; 1 = supported
; Bit 1 =1 (Reserved)
; Bit 2 = Output functions supported (tty out,
scroll, etc)
; 0 = not supported
; 1 = supported
; Bit 3 = Monochrome/color mode
; 0 = Monochrome
; 1 = Color
; Bit 4 = Mode Type
; 0 = Text mode
; 1 = Graphics Mode
; Bits 5-15 = Reserved
;
; IF WIN A & B ARE NOT SUPPORTED THEN ASSUME THAT DISPLAY BUFFER
RESIDES AT THE
; STANDARD CPU ADDRESS APPROPRIATE FOR THE MemoryModel OF THE MODE
;
  WinAAttributes db ? ; Window A Attributes
; Bit 0 = CPU Window Supported
; 0 = Not Supported
; 1 = Supported
; Bit 1 = CPU Window Readable (if supported)
; 0 = Not Readable
; 1 = Readable
; Bit 2 = CPU Window Writeable (if supported)
; 0 = Not Writeable
; 1 = Writeable
  WinBAttributes db ? ; Window B Attributes
; Same as Window A Attributes but for Window B

  WinGranularity dw ? ; Window Granularity
; Smallest boundry, in Kbytes, on which the window
can be placed
; in video memory. Undefined if the appropriate
Win?Attributes

```

```

; field is not set.
WinSize          dw ? ; Window Size in Kbytes
WinASegment      dw ? ; Window A Start Segment in CPU Address Space
WinBSegment      dw ? ; Window B Start Segment in CPU Address Space
WinFuncPtr       dd ? ; pointer to windowing function
; Address of function invoked by calling VESA BIOS
Function 05.
; This pointer is provided to allow hi-end apps to
access the
; paging registers quicker than via the INT 10h
when the extra
; speed is required. If this field is NULL, then
Function 05
; must be used.
BytesPerScanLine dw ? ; bytes per scan line
XResolution       dw ? ; Horizontal Resolution
; Value is in pixels in graphics modes, characters
in text mode.
YResolution       dw ? ; Vertical Resolution
; Value is in pixels in graphics modes, characters
in text mode.
XCharSize         db ? ; character cell width in pixels
YCharSize         db ? ; character cell height in pixels
NumberOfPlanes    db ? ; number of memory planes available to software in
that mode. For
; standard 16-color VGA graphics, this would be 4;
for std packed
; pixel modes, this would be 1.
BitsPerPixel      db ? ; bits per pixel
; Number of bytes that describe 1 pixel. Std 4
plane 16-color VGA
; would be 4; packed pixel 256-color would be 8.
NumberOfBanks     db ? ; number of banks in which the scan lines are grouped. CGA
modes have
; 2 banks, Hercules graphics have 4, VGA modes 0D-
13h do NOT have
; scan line banks and should be set to 1.
MemoryModel       db ? ; memory model type
; 00 = Text Mode
; 01 = CGA Graphics
; 02 = Hercules Graphics
; 03 = 4 plane planar
; 04 = packed pixel
; 05 = Non-chain 4,256 color
; 06 = Direct Color
; 07 = YUV
; 08-0f = Reserved, to be defined by VESA
; 10-ff = Reserved, to be defined by OEM
BankSize          db ? ; bank (group of scan lines)size in kbytes
; For CGA & Herc. graphics modes this is 8. For
modes that don't
; have scanline banks (such as VGA modes 0D-13h)
this is 0.
NumberOfImagePages db ? ; Number of images
; This is the number of ADDITIONAL complete display
images that
; will fit into the VGA's memory at one time in
this mode. If this
; field is non-zero, the app can load multiple
images into memory
; and flip between them.

```

```

Reserved          db 1    ; Reserved for page function - Always set to 1
;
; The following ___MaskSize variables:
;   These define the size, in bits, of the red, green, and blue
components of a direct ;   color pixel.  A bit mask can be constructed from the ___MaskSize
fields using shift ;   arithmetic.  i.e.  the ___MaskSize values for a Direct Color
5:6:5 mode would be ;   5, 6, 5, 0 for the red, green, blue, and reserved fields,
respectively.  Note that in ;   YUV memory modes, red is used for V, green for Y, and the blue
is used for U.  The ;   ___MaskSize fields should be set to 0 for modes using a
MemoryModel that does not ;   have pixels with component fields.
;
; The following ___FieldPosition variables:
;   These define the position within the direct color pixel or YUV
pixel of the least ;   significant bit of the respective color component.  A color
value can b aligned ;   with its pixel field by shifting the value left by the
___FieldPosition. ;   For example, the ___FieldPosition values for a Direct Color
5:6:5 mode would ;   be 11, 5, 0, and 0, for red, green, blue, and reserved fields
respectively.  Note ;   that in YUV memory modes, red is used for V, green for Y, and
the blue is used ;   for U.  The ___FieldPosition fields should be set to 0 for modes
using a ;   MemoryModel that does not have pixels with component fields.
;
RedMaskSize  db ?    ; size of direct color red mask in bits
RedFieldPosition  db ?    ; bit position of LSB of red mask
GreenMaskSize db ?    ; size of direct color green mask in bits
GreenFieldPosition db ?    ; bit position of LSB of green mask
BlueMaskSize  db ?    ; size of direct color blue mask in bits
BlueFieldPosition db ?    ; bit position of LSB of blue mask
RsvdMaskSize  db ?    ; size of direct color reserved mask in bits
RsvdFieldPosition db ?    ; bit position of LSB of reserved mask
DirectColorInfo db ?    ; Direct Color Mode Attributes
;   Bit 0 = Color ramp is fixed/programmable
;   0 = fixed
;   1 = programmable.  It is assumed that the red,
green,
;   and blue lookup tables can be loaded
using a standard ;   VGA DAC color register BIOS call (AX =
1012h).
;   Bit 1 = Rsvd field of the direct color pixel
usable
;   0 = RsvdMaskSize and RsvdFieldPosition are
reserved
;   1 = RsvdMaskSize and RsvdFieldPosition are
usable by the app.
Reserved          db 216 dup(?) ; Reserved for future expansion
VgaInfoBlock ends

```

AH = 4F
AL = 02 Set Super VGA Video Mode
BX = Super VGA Video Mode
 Bits 0-14 = video mode
 Bit 15 = Clear memory flag
 0=Clear Video Memory
 1=Don't Clear Video Memory

Return:
AX = Status
All other registers are preserved

AH = 4F
AL = 03 Return Current Video Mode

Return:
AX = Status
BX = Current Video Mode
All other registers are preserved

Note:
This function does not return the state of the "clear memory bit" (Bit 7 of 40:87). If the application wants to obtain the "clear memory bit" it must call the standard VGA BIOS function 0Fh.

AH = 4F
AL = 04 Save/Restore Super VGA Video State Functions

Note:
These functions are a superset of the three subfunctions under the standard VGA BIOS function 1Ch (Save/Restore Video State).

DL = 00h Return Save/Restore State Buffer Size
CX = Requested States
 Bit 0 = Save/Restore Video Hardware State
 Bit 1 = Save/Restore Video BIOS Data State
 Bit 2 = Save/Restore Video DAC State
 Bit 3 = Save/Restore Super VGA State

Return:
AX = Status
BX = Number of 64-byte blocks to hold the state buffer
All other registers are preserved

DL = 01h Save Super VGA Video State
CX = Requested States (see sub-function 00 above for definition of states)
ES:BX = Pointer to buffer where the data is to be stored

Return:
AX = Status
All other registers are preserved

DL = 02h Restore Super VGA Video State
CX = Requested States (see sub-function 00 above for definition of states)
ES:BX = Pointer to buffer containing previously stored data

Return:
AX = Status

All other registers are preserved

AH = 4F
AL = 05 CPU Video Memory Window Control

BH = 00 Select Super VGA Video Memory Window

BL = Window Number

0 = Window A

1 = Window B

DX = Window position in video memory (in window granularity units)

Return:

AX = Status

All other registers are preserved (see notes below for "Far Call" implementation)

BH = 01 Return Super VGA Video Memory Window

BL = Window Number

0 = Window A

1 = Window B

Return:

AX = Status

DX = Window position in video memory (in window granularity units)

All other registers are preserved (see notes below for "Far Call" implementation)

Note:

For high performance application requirements, this function can be accessed directly via a far call.

The address to call may be obtained using VESA BIOS function 01h, **Return Super VGA Mode Information**. This function may differ for different modes so the application should get the function's address after each mode set.

Note:

The following rules apply when accessing this function via a "far call" implementation instead of via the VESA BIOS call.

- 1) No status information is returned to the application.
- 2) Registers AX and DX will be destroyed. It is the applications responsibility to preserve them if they must be preserved.
- 3) BH and BL (and DX for sub-function 00h) must be set prior to calling the function.
- 4) AH and AL do not need to be loaded for the "far call" implementation.

VESA VBE Programming example

Below is a skeleton framework for VESA mode programming. Contact the Video Electronics Standards Association at the address listed at the beginning of this section for further support of VESA® Super VGA programming.

1) Application allocates a 256 byte buffer and makes the Return Super VGA Information call (VBE function 00). The application can determine the level of VESA compliance by checking the VESA Signature and Version returned in the Buffer. This call also provides a list of available video modes.

2) The application will normally know one of the following: 1) the VESA mode it needs, or 2) the parameters of the video environment that are required. In the first case, the available modes will have been returned in step 1 above. The application should then request, via Return Super VGA Mode Information call (VBE function 01), the details of the required mode. In the second case, the application should request the details of each of the available modes returned in step 1 until it finds a mode with acceptable parameters. In

either case a valid 256 byte buffer must be provided for each Return Super VGA Mode Information function (VBE function 01) called.

3) Application would then store the current video mode number via Get Super VGA Mode call (VBE function 03) for use on exit.

4) Application would then use Set Super VGA Mode (VBE function 02) to initialize the video hardware. The application now has full access and information required to manipulate the video as needed.

5) Just prior to terminating the application, the application should restore the original video mode (retrieved in step 3) via the Set Super VGA Mode (VBE function 02). The application would then do any additional cleanup that might be required (such as to free the allocated buffers from steps 1 and 2 if they were dynamically allocated) and then exit.

5.11.3 Interrupt 11H Functions - Return System Information

Return the equipment installed determined by the BIOS power on diagnostics.

Exit:

AX	Equipment information
0	Diskette drives attached
1	Math coprocessor installed
2,3	Planer RAM size in 16k increments
4,5	Initial video mode
	00 - Unused
	01 - 40x25
	10 - 80x25
	11 - Monochrome
6,7	Diskette drives
	00 - 1 drive
	01 - 2 drives
	10 - 3 drives
	11 - 4 drives
8	Not used
9-11	Number of serial adapters
12	Game Adapter installed
13	Not used
14,15	Number of parallel adapters

5.11.4 Interrupt 12H Functions - Return System Memory Size

Return the amount of system memory determined during the power on diagnostics. The value returned is expressed in kbytes.

Exit:

AX	Number of 1k memory blocks
----	----------------------------

5.11.5 Interrupt 13H Functions - Diskette Services

Interrupt 13 is the BIOS software interface for access to the 5 1/4 and 3 1/2 inch diskette drives. If an error occurs during the diskette operation, the AH register will contain the error code. The error codes are defined in the following table.

Code	Description
80	Time out occurred
40	Seek failed
20	NEC failed
10	CRC failed
0C	Media not found
09	DMA crossed 64k boundary
08	DMA failed
06	Media changed
04	Sector not found
03	Write protect occurred
02	Bad address mark
01	Illegal command
00	No error occurred

Diskette Error Codes

All values above are hexadecimal

The following table contains the combinations of drive types and media types which are supported by the diskette routines.

Media	Drive	Sectors/Track	Tracks
360k	360k	8-9	40
360k	1.2M	8-9	40
1.2M	1.2M	15	80
720k	720k	9	80
720k	1.44M	9	80
1.44M	1.44M	18	80

Diskette Types

The following outlines the diskette services in more detail with their corresponding register entry values.

Interrupt 13 Functions - Diskette

AH = 00 **Reset diskette system**

Exit:

AH Operation status

C Operation failed
(AH=error code)

AH = 01 Return last operation status

Exit:

AH Operation status
C Operation failed
 (AH=error code)

AH = 02 Read sectors from diskette

ES:BX Buffer address
DL Drive number (0-1)
DH Head number
CH Track number
CL Sector number
AL Number of sectors

Exit:

AL Number of sectors transferred
AH Operation status
C Operation failed
 (AH=error code)

AH = 03 Write sectors to diskette

ES:BX Buffer address
DL Drive number (0-1)
DH Head number
CH Track number
CL Sector number
AL Number of sectors

Exit:

AL Number of sectors transferred
AH Operation status
C Operation failed
 (AH=error code)

AH = 04 Verify sectors on diskette

DL Drive number (0-1)
DH Head number
CH Track number
CL Sector number
AL Number of sectors

Exit:

AL Number of sectors transferred
AH Operation status
C Operation failed
 (AH=error code)

AH = 05 Format track on diskette

ES:BX Buffer address
DL Drive number (0-1)
DH Head number
CH Track number
CL Sector number
AL Number of sectors

Exit:

AL Number of sectors transferred

AH Operation status
C Operation failed
 (AH=error code)

AH = 08 Return diskette parameters

DL Drive number
Exit:
DL Number of diskette disks
DH Maximum head number
CH Maximum cylinder number

AH = 15H Read drive type

Exit:
C flag not set then AH contains:
 00 Drive not present
 01 No change line available on drive
 02 Change available on drive
 03 Fixed disk
Otherwise:
AH Operation status
C Operation failed
 (AH=error code)

AH = 16H Disk change line status

DL Drive Number (0-1)
Exit:
AH Disk change status
 00 Disk change not active
 06 Disk change line active
 and C flag set

AH = 17H Set DASD drive type for format

AL DASD Type
 00 Not Used
 01 360kbytes floppy in 360kbytes drive
 02 360kbytes floppy in 1.2Mbytes drive
 03 1.2Mbytes floppy in 1.2Mbytes drive
 04 1.44Mbytes floppy in 1.44Mbytes
DL Drive Number (0-1)
Exit:
AH Operation status
C Operation failed
 (AH=error code)

AH = 18H Set drive type for format

CH Number of tracks
CL Number of sectors
DL Drive Number (0-1)
Exit:
ES:DI Pointer to parameter table
AH Operation Status
C Operation failed
 (AH=error code)

5.11.6 Interrupt 13H Functions - Fixed Disk Services

The AH register is used to determine the BIOS fixed disk command to be performed. The following are the error codes that can be returned by the fixed disk routines:

Fixed Disk Error Codes

Description	Code
Sense failure	FF
Status error	E0
Write fault	CC
Undefined error	BB
Drive not ready	AA
Time out occurred	80
Seek failed	40
Controller failure	20
ECC failed	10
Bad track	0B
Bad sector	0A
Parameter init failed	07
Reset failed	05
Sector not found	04
Write protect occurred	03
Bad address mark	02
Illegal command	01
No error occurred	00

All values above are hexadecimal

The following outlines the fixed disk services in more detail, with their corresponding register entry values.

Interrupt 13 Functions - Fixed Disk

AH = 00 **Reset fixed disk system**

Exit:

AH Operation status
C Operation failed
 (AH=error code)

AH = 01 **Return last operation status**

Exit:

AH Operation status
C Operation failed
 (AH=error code)

AH = 02 **Read sectors from fixed disk**

ES:BX Buffer address
DL Drive number (80H-81H)
DH Head number
CH Track number

CL Sector number/Cyl high
AL Number of sectors
Exit:
AL Number of sectors transferred
AH Operation status
C Operation failed
(AH=error code)

AH = 03 Write sectors to fixed disk

ES:BX Buffer address
DL Drive number (80H-81H)
DH Head number
CH Track number
CL Sector number/Cyl high
AL Number of sectors
Exit:
AL Number of sectors transferred
AH Operation status
C Operation failed
(AH=error code)

AH = 04 Verify sectors on fixed disk

DL Drive number (80H-81H)
DH Head number
CH Track number
CL Sector number/Cyl high
AL Number of sectors
Exit:
AL Number of sectors transferred
AH Operation status
C Operation failed
(AH=error code)

AH = 05 Format track on fixed disk

ES:BX Buffer address
DL Drive number (80H-81H)
DH Head number
CH Track number
CL Sector number/Cyl high
AL Number of sectors
Exit:
AL Number of sectors transferred
AH Operation status
C Operation failed
(AH=error code)

AH = 08 Return fixed disk parameters

DL Drive number
Exit:
DL Number of fixed disks
DH Maximum head number
CH Maximum cylinder number
CL Maximum sector number/cylinder

AH = 09 Initialize drive parameters

Exit:

AH Operation status
C Operation failed
 (AH=error code)

AH = 0A Read long

ES:BX Buffer address
DL Drive number (80H-81H)
DH Head number
CH Cylinder number
CL Sector number/Cyl high
AL Number of sectors

Exit:

AL Number of sectors transferred
AH Operation status
C Operation failed
 (AH=error code)

AH = 0B Write long

ES:BX Buffer address
DL Drive number (80H-81H)
DH Head number
CH Cylinder number
CL Sector number/Cyl high
AL Number of sectors

Exit:

AL Number of sectors transferred
AH Operation status
C Operation failed
 (AH=error code)

AH = 0C Seek drive

ES:BX Buffer address
DL Drive number (80H-81H)
DH Head number
CH Cylinder number
CL Cylinder high

Exit:

AH Operation status
C Operation failed
 (AH=error code)

AH = 0D Alternate disk reset

DL Drive number (80H-81H)

Exit:

AH Operation status
C Operation failed
 (AH=error code)

AH = 10 Test drive ready

DL Drive number (80H-81H)

Exit:

AH Operation status

C	Operation failed (AH=error code)
AH = 11	Recalibrate drive
DL	Drive number (80H-81H)
Exit:	
AH	Operation status
C	Operation failed (AH=error code)
AH = 14	Controller diagnostic
DL	Drive number (80H-81H)
Exit:	
AH	Operation status
C	Operation failed (AH=error code)
AH = 15	Read DASD type
DL	Drive number
Exit:	
AH	0 Not present 1 Diskette without change line 2 Diskette with change line 3 Fixed disk
CXDX	Number of 512 blocks

5.11.7 Interrupt 14H Functions - Serial Services

The INT 14 software interrupt will handle serial I/O function requests. This code will use the AH register to decide which of the interrupt 14 functions are to be invoked. The following outlines the serial communication services:

AH = 00	Set communication parameters
AL	Init parameters
Bit 1,0 =	10 - 7 data bits 11 - 8 data bits
Bit 2 =	0 - 1 stop bit 1 - 2 stop bits
Bit 4,3 =	00 - No parity 10 - No parity 01 - Odd parity 11 - Even parity
Bit 7-5 =	000 - 110 Baud-417 divisor 001 - 150 Baud-300 divisor 010 - 300 Baud-180 divisor 011 - 600 Baud-0C0 divisor 100 - 1200 Baud-060 divisor 101 - 2400 Baud-030 divisor 110 - 4800 Baud-018 divisor 111 - 9600 Baud-00C divisor
DX	Serial port
Exit:	

AL Modem status
AH Line status (Bit 7 - timeout)

AH = 01 Transmit character

AL Character to transmit
DX Serial port
Exit:
AH Line status (Bit 7 - timeout)

AH = 02 Receive character

DX Serial port
Exit:
AL Character received
AH Status (Bit 7 - timeout)

AH = 03 Return status

DX Serial port
Exit:
AL Modem status
AH Line status (Bit 7 - timeout)

Bit Modem Status

0 Delta clear to send
1 Delta data set ready
2 Trailing edge ring indicator
3 Delta (RLSD)
4 Clear to send
5 Data set ready
6 Ring indicator
7 Received line signal detect

Bit Line Status

0 Data ready
1 overrun error
2 Parity error
3 Framing error
4 Break detect
5 Trans holding register empty
6 Trans shift register empty
7 Time out error

5.11.8 Interrupt 15H Functions - Miscellaneous Hardware Services

The INT 15 software interrupt will handle a variety of functions which are particular to the hardware. These include functions for multi-tasking, joystick functions, and extended memory functions. This code will use the AH register to decide which of the interrupt 15 functions are to be invoked. The following outlines the miscellaneous services in more detail with their corresponding register entry values:

AH = 00-7F **Cassette functions**
Exit:
AH 86
C Set

AH = 80 **Device open**
BX Device identifier
CX Process identifier

AH = 81 **Device close**
BX Device identifier
CX Process identifier

AH = 82 **Program termination**
BX Device identifier

AH = 83 **Event wait**
AL 00 Set event timer
ES:BX Pointer to post byte
CXDX Microseconds before post
AL 01 Cancel event timer

AH = 84 **Joystick support**
DL 00 Read switch settings
Exit:
AL Switch settings

DL 01 Return resistive inputs
Exit:
AX Resistive input bit 0
BX Resistive input bit 1
CX Resistive input bit 2
DX Resistive input bit 3

AH = 85 **System request key pressed**
AL 00 System request key pressed
AL 01 System request key released

AH = 86 **Wait microseconds**
CX:DX Number of microseconds to wait

AH = 87 **Move block**
CX Number of words to move
ES:SI Pointer to global descriptor table

AH = 88 **Extended memory size**
Exit:
AX Amount of extended memory in kbytes

AH = 89 **Enter protected mode**
ES:SI Pointer to descriptor
BH Offset into IDT for IRQ 00-07
BL Offset into IDT for IRQ 08-0F

AH = 90 **Device busy**
 AL Type code

AH = 91 **Interrupt completion**
 AL Type code

AH = C0 **Return system parameters**
 Exit:
 ES:BX Pointer to configuration table
 AH 00

5.11.9 Interrupt 16H Functions - Keyboard Services

The INT 16 software interrupt will handle keyboard I/O function requests. This code will use the AH register to decide which of the interrupt 16 functions are to be invoked. The following outlines the keyboard services:

AH = 00 **Read character from buffer**
 Exit:
 AL ASCII keystroke pressed
 AH Scan code of key

AH = 01 **Return buffer status**
 Exit:
 AL ASCII keystroke pressed
 AH Scan code of key
 ZF No keystroke available
 NZ Keystroke in buffer

AH = 02 **Return shift status**
 Exit:
 AL Current shift status

AH = 03 **Set type-a-matic rate.**
 AL 05 (subfunction number)
 BL 00H through 1FH, Type-a-matic rate
 (30 chars/sec to 2 char/sec)
 BH 00H through 03H, Delay rate
 (0-250mills,1-500mills.)
 Exit: None

AH = 05 **Add key to Keyboard buffer.**
 CL ASCII code
 CH Scan code
 Exit:
 AL 0 if insert successful
 1 if keyboard buffer full

AH = 10 **Read extended character from buffer.**
 Exit:
 AL ASCII keystroke pressed
 AH Scan code of key

AH = 11 Return extended buffer status.

Exit:
AL ASCII keystroke pressed
AH Scan code of key
ZF No keystroke available
NZ Keystroke in buffer

AH = 12 Return extended shift status.

Exit:
AL Shift status
AH Extended shift status

5.11.10 Interrupt 17H Functions - Parallel Printer Services

The INT 17 software interrupt will handle the parallel printer I/O function requests. This code will use the AH register to decide which of the interrupt 17 functions is to be invoked. The following outlines the printer services:

AH = 00 Print character

AL Character to print
DX Printer port
Exit:
AH Status

AH = 01 Initialize printer port

DX Printer port
Exit:
AH Status

AH = 02 Return printer status

DX Printer port
Exit:
AH Status

Bit	Status
0	Time out
1	Reserved
2	Reserved
3	I/O error
4	Selected
5	Out of paper
6	Acknowledge
7	Not busy

Printer Status

5.11.11 Interrupt 1AH Functions - Time of Day Services

The INT 1AH software interrupt will handle the time of day I/O function requests. This code will use the AH register to decide which of the interrupt 1AH functions are to be invoked. The following outlines the time of day services in more detail with their corresponding register entry values:

AH = 00 Read current time

Exit:
CX High count word
DX Low count word
AL Day rollover

AH = 01 Set current time

CX High time word
DX Low time word

AH = 02 Read real time clock

Exit:
CH BCD hours
CL BCD minutes
DH BCD seconds

AH = 03 Set the real time clock

CH BCD hours
CL BCD minutes
DH BCD seconds
DL 1 if daylight saving;
 0 otherwise

AH = 04 Read date from real time clock

Exit:
CH BCD century
CL BCD year
DH BCD month
DL BCD date

AH = 05 Set date in real time clock

CH BCD century
CL BCD year
DH BCD month
DL BCD date

AH = 06 Set alarm

CH BCD hours to alarm
CL BCD minutes to alarm
DH BCD seconds to alarm

AH = 07 Reset the alarm

5.12 BIOS Data Area

The BIOS Data Area contains information about the current operating environment of the AT system. The BIOS Data Area is located from physical address 400H to 501H. The following tables define the variables in the data area and a brief description of their function.

Offset	Size	Description																						
00	2	Com1 address																						
02	2	Com2 address																						
04	2	Com3 address																						
06	2	Com4 address																						
08	2	Lpt1 address																						
0A	2	Lpt2 address																						
0C	2	Lpt3 address																						
0E	2	Lpt4 address (note 1)																						
10	2	Equipment installed																						
		<table border="1"> <thead> <tr> <th>Bit</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Diskette drives attached</td> </tr> <tr> <td>1</td> <td>Math coprocessor installed</td> </tr> <tr> <td>2,3</td> <td>Planer RAM size in 16k increments</td> </tr> <tr> <td>4,5</td> <td>Initial video mode 00 - Unused 01 - 40x25 10 - 80x25 11 - Monochrome</td> </tr> <tr> <td>6,7</td> <td>Diskette drives 00 - 1 drive 01 - 2 drives 10 - 3 drives 11 - 4 drives</td> </tr> <tr> <td>8</td> <td>Not used</td> </tr> <tr> <td>9-11</td> <td>Number of serial adapters</td> </tr> <tr> <td>12</td> <td>Game Adapter installed</td> </tr> <tr> <td>13</td> <td>Not Used</td> </tr> <tr> <td>14,15</td> <td>Number of parallel adapters</td> </tr> </tbody> </table>	Bit	Definition	0	Diskette drives attached	1	Math coprocessor installed	2,3	Planer RAM size in 16k increments	4,5	Initial video mode 00 - Unused 01 - 40x25 10 - 80x25 11 - Monochrome	6,7	Diskette drives 00 - 1 drive 01 - 2 drives 10 - 3 drives 11 - 4 drives	8	Not used	9-11	Number of serial adapters	12	Game Adapter installed	13	Not Used	14,15	Number of parallel adapters
Bit	Definition																							
0	Diskette drives attached																							
1	Math coprocessor installed																							
2,3	Planer RAM size in 16k increments																							
4,5	Initial video mode 00 - Unused 01 - 40x25 10 - 80x25 11 - Monochrome																							
6,7	Diskette drives 00 - 1 drive 01 - 2 drives 10 - 3 drives 11 - 4 drives																							
8	Not used																							
9-11	Number of serial adapters																							
12	Game Adapter installed																							
13	Not Used																							
14,15	Number of parallel adapters																							
12	1	Interrupt flag (POST)																						
13	2	Memory size (k bytes)																						
15	1	Reserved																						
16	1	Control flag																						
17	1	Keyboard flag 0																						
		<table border="1"> <thead> <tr> <th>Bit</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Right shift key pressed</td> </tr> <tr> <td>1</td> <td>Left shift key pressed</td> </tr> <tr> <td>2</td> <td>Control key pressed</td> </tr> <tr> <td>3</td> <td>Alt key pressed</td> </tr> <tr> <td>4</td> <td>Scroll lock on</td> </tr> <tr> <td>5</td> <td>Num lock on</td> </tr> <tr> <td>6</td> <td>Caps lock on</td> </tr> <tr> <td>7</td> <td>Insert mode on</td> </tr> </tbody> </table>	Bit	Definition	0	Right shift key pressed	1	Left shift key pressed	2	Control key pressed	3	Alt key pressed	4	Scroll lock on	5	Num lock on	6	Caps lock on	7	Insert mode on				
Bit	Definition																							
0	Right shift key pressed																							
1	Left shift key pressed																							
2	Control key pressed																							
3	Alt key pressed																							
4	Scroll lock on																							
5	Num lock on																							
6	Caps lock on																							
7	Insert mode on																							

Offset	Size	Description												
18	1	Keyboard flag 1												
		<table border="1"> <thead> <tr> <th>Bit</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>3</td> <td>Freeze state</td> </tr> <tr> <td>4</td> <td>Scroll lock pressed</td> </tr> <tr> <td>5</td> <td>Num lock pressed</td> </tr> <tr> <td>6</td> <td>Caps lock pressed</td> </tr> <tr> <td>7</td> <td>Insert mode pressed</td> </tr> </tbody> </table>	Bit	Definition	3	Freeze state	4	Scroll lock pressed	5	Num lock pressed	6	Caps lock pressed	7	Insert mode pressed
Bit	Definition													
3	Freeze state													
4	Scroll lock pressed													
5	Num lock pressed													
6	Caps lock pressed													
7	Insert mode pressed													
19	1	Keypad input byte												
1A	2	Key buffer head												
1C	2	Key buffer tail												
1E	20	Key buffer												
3E	1	Seek/recalibrate status												
3F	1	Drive motor status												
40	1	Motor on time												
41	1	Diskette status												
42	7	Controller status												
49	1	Video mode												
4A	2	Video columns												
4C	2	Video length												
4E	2	Video start												
50	10	Cursor locations												
60	2	Cursor size												
62	1	Active page												
63	2	6845 address												
65	1	Mode register value												
66	1	Video palette												
67	4	ROM check address												
6B	1	CPU rate control												
6C	2	Timer count low word												
6E	2	Timer count high word												
70	1	Timer overflow byte												
71	1	Break pressed flag												
72	2	Soft reset flag												
74	1	Fdisk status												
75	1	Number of fixed disks												
76	1	Fixed disk control												
77	1	Reserved												
78	4	Lpt1-4 timeout values												
7C	4	Com1-4 timeout values												
80	2	Key buffer start												
82	2	Key buffer end												
84	1	Number of video rows												

Offset	Size	Description																
85	2	Character Length																
87	1	EGA Status A																
88	1	EGA Status B																
89	1	VGA Status A																
8A	1	Diskette Control																
8B	1	Last diskette data rate																
8C	1	FDisk status																
8D	1	FDisk error value																
8E	1	FDisk Interrupt flag																
8F	1	Floppy info nibbles																
90	4	Floppy state information																
94	2	Floppy cylinder number																
96	1	Keyboard control																
97	1	Keyboard flag 2																
		<table border="1"> <thead> <tr> <th>Bit</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Scroll LED on</td> </tr> <tr> <td>1</td> <td>Num lock LED on</td> </tr> <tr> <td>2</td> <td>Caps lock LED on</td> </tr> <tr> <td>4</td> <td>Ack code received</td> </tr> <tr> <td>5</td> <td>Resend received</td> </tr> <tr> <td>6</td> <td>LED being updated</td> </tr> <tr> <td>7</td> <td>Keyboard error</td> </tr> </tbody> </table>	Bit	Definition	0	Scroll LED on	1	Num lock LED on	2	Caps lock LED on	4	Ack code received	5	Resend received	6	LED being updated	7	Keyboard error
Bit	Definition																	
0	Scroll LED on																	
1	Num lock LED on																	
2	Caps lock LED on																	
4	Ack code received																	
5	Resend received																	
6	LED being updated																	
7	Keyboard error																	
98	4	RTC user flag																
9C	2	RTC time low word																
9E	2	RTC time high word																
A0	1	RTC wait flag																
A1	7	Reserved																
A8	4	EGA/VGA environment pointer																
AC	21	Reserved																
DB	2	Day counter																
DE	22	Reserved																
100	1	Print screen flag																

BIOS Data Area Description

Note 1:

The Lpt4 address in the BIOS data area (Offset 0E) is used for the Extended BIOS data area segment under CDI Extensions.

5.13 Interrupt Vectors

The table on the following page describes the AT system interrupt vectors. Status indicates whether the BIOS supports the interrupt.

INT	Description	Status
00	Divide by zero	Not Supported
01	Single step	Not Supported
02	Non-Maskable interrupt	Supported
03	Breakpoint	Not Supported
04	Overflow	Not Supported
05	Print Screen Interrupt	Supported
06	Reserved	Not Supported
07	Reserved	Not Supported
08	IRQ0 - System Timer Interrupt	Supported
09	IRQ1 - Keyboard Interrupt	Supported
0A	IRQ2 - Reserved	Not Supported
0B	IRQ3 - Secondary Serial Adapter	Not Supported
0C	IRQ4 - Primary Serial Adapter	Not Supported
0D	IRQ5 - Secondary Parallel Interrupt	Not Supported
0E	IRQ6 - Floppy Disk Interrupt	Supported
0F	IRQ7 - Primary Parallel Interrupt	Not Supported
10	BIOS Video Interface	Supported
11	BIOS Request system information	Supported
12	BIOS Request system memory size	Supported
13	BIOS Fixed Disk/Diskette Interface	Supported
14	BIOS Serial Interface	Supported
15	BIOS System Functions Interface	Supported
16	BIOS Keyboard Interface	Supported
17	BIOS Parallel Printer Interface	Supported
18	BIOS Secondary Boot Request	Supported
19	BIOS Primary Boot Request	Supported
1A	BIOS System Timer Interface	Supported
1B	BIOS Control Break Interrupt	Supported
1C	BIOS User System Timer Interrupt	Supported
1D	BIOS Video init parameters	Supported
1E	BIOS Diskette Parameters	Supported
1F	BIOS Video Graphic Characters	Supported
40	BIOS Diskette (when fixed disk present)	Supported
41	BIOS Fixed disk 0 parameters	Supported
46	BIOS Fixed disk 1 parameters	Supported
70	IRQ8 - Real time clock interrupt	Supported
71	IRQ9 - IRQ2 redirection	Supported
72	IRQA - Reserved	Not Supported
73	IRQB - Reserved	Not Supported
74	IRQC - Reserved	Not Supported
75	IRQD - 80387SX interrupt	Supported
76	IRQE - Fixed Disk interrupt	Supported
77	IRQF - Reserved	Not Supported

BIOS Interrupt Support

All values above are hexadecimal

6. SEMICONDUCTOR DISK REFERENCE

6.1 Overview

Semiconductor disks are RAM and/or ROM devices which are effectively presented to the system as extra "floppy diskettes" when booting with either MS-DOS or DR-DOS. These "ROM disks" and "RAM disks" are recognized, controlled, and handled by the BIOS adapter module device driver, SCDISK.ADP, which is provided with ROMTools. SCDISK.ADP treats a RAM disk as a standard, MS-DOS/DR-DOS floppy diskette. This means you can read and write files, create and delete files, execute programs, format, and even "boot" your system from the RAM disk. A ROM disk, however, operates identically to a floppy diskette with the "Write Protect Tab" installed. You may perform any read operation with a ROM disk that you can do with a RAM disk but it cannot be written. Once a ROM disk is created, it cannot be modified without being erased and reprogrammed.

The SCDISK.ADP adapter module included with ROMTools provides support for RAM or ROM disks with any of Computer Dynamics' complete PC compatible processor line. DOS-based embedded systems may boot from a Floppy Drive, ROM Disk, RAM Disk, or Hard disk. Computer Dynamics' ROMTools package, (purchased separately), is required to generate ROM based applications.

If you wish to use semiconductor support please contact your Computer Dynamics' salesperson and request the ROMTools package. This package contains all the software and documentation you will need to create your own semiconductor disk and enter the world of diskless computing.

APPENDICES

ADDITIONAL READING

IBM PC Technical Reference, IBM Corp., 1983. - Complete reference to the PC. There are versions for both the PC/XT and PC/AT.

Microprocessor and Peripheral Handbook Volume 1 - Microprocessor, Intel, 1989. This volume contains register definitions for the parts duplicated in the ACC2168DT.

The Programmer's PC Sourcebook, Thom Hogan, Microsoft Press, 1988. This volume contains many tables of useful information on the PC family. It is very handy for the assembly language programmer.

The Peter Norton Programmer's Guide to the IBM PC, Peter Norton, Microsoft Press, 1985. This book is an excellent introduction to the logical organization of the PC family. It highlights differences between different versions.

Microsoft MS-DOS User's Guide and User's Reference, Microsoft, 1988. This volume is included with each purchase of MS-DOS. It will explain DOS commands and some important operations such as formatting.

Programmer's Guide to the EGA and VGA Cards, Richard F. Ferraro, Addison-Wesley Publishing Company, 1988. This volume explains in good detail the function and use of the VGA registers and BIOS calls.

WD90C24 Windows Accelerated High Resolution VGA LCD Controller for Low Power Applications Data Sheet, Western Digital Corporation, 1993. It provides additional details about the registers used for panels and generating gray scale output.

Super VGA BIOS Extension VBE Version 1.2, Video Electronics Standards Association (VESA), 1991. This document provides the standardized software interface for VESA VBE compliant Super VGA hardware.

Chips and Technologies F82C735 I/O Peripheral Controller with Printgine (Dual Buffered UART, Floppy Disk Controller and Parallel Port with EPP capability), Chips and Technologies 1993. This data sheet provides information on bit assignments and register assignments for the COM ports, configuration of standard/bidirectional parallel ports as well as detailed information on Enhanced Printer Port configuration.

C Programmers Guide to Serial Communications, Joe Campbell, Howard W. Sams & Company, 1987. A complete reference to programming asynchronous serial communications.

SBC I/O OPTIONS

The SBC has a 100% compatible parallel printer port. That means that all output pins can be read back at the pin. Additionally, four pins have open collector outputs allowing them to be used as inputs or outputs. Compatibility also means IBM defined connectors, pinouts, port mapping, and logic conventions.

If not used for a parallel printer, the parallel printer port can be used as general purpose parallel I/O. The outputs can drive LEDs, intelligent LCD displays, motor controllers or higher voltage devices through solid state relays. The input lines can be used to read switches, multiplexed keypads, encoders, or interface to higher voltages through solid state relay racks. The parallel printer port can be used as follows:

8 bits output only with read back

4 bits open collector outputs which are programmable as input or output with read back

4 bits input only

1 bit programmable interrupt and readable input

All outputs have read back. Read back keeps you from having to store the output value in memory when changing individual bits. An input buffer actually senses the level at the output and lets you read it. Unfortunately, if the outputs are heavily loaded, the output voltage may not be within TTL voltage limits causing the read back value to be incorrect. LEDs are current devices and will light up even though the pin is greater than 0.8V. However, the read back buffer may see that pin as a high. In these cases, store a copy of what you output in memory.

The bits are accessed by reading or writing to I/O port locations. The following tables show the I/O port locations and list the connector pins affected. The board has a 26-pin header connector which is normally cabled to a DB-25 connector. The tables and connector diagrams list both the header and DB-25 connector pin numbers.

	D7	D6	D5	D4	D3	D2	D1	D0
Header Pin Number	17	15	13	11	9	7	5	3
DB-25 Pin Number	9	8	7	6	5	4	3	2

Write Address 378H (Use as outputs only.) Reset Condition: Undetermined

These pins are connected to driver outputs with direct read back via port 378H. Each output can source 2.6 mA and sink 24 mA while maintaining TTL voltage levels. This port is not reset on power-up. External devices must not drive these lines or damage may occur.

	D7	D6	D5	D4	/D3	D2	/D1	/D0	
Header Pin Number				IRQ7 Enable	8	6	2	1	
DB-25 Pin Number				IRQ7 Enable	17	16	14	1	
Reset Condition				0	0	0	0	0	
Write for Input	0	0	0	0	0	1	0	0	= 04H

Write Address 37AH (These pins may be outputs or inputs.)

These four pins are connected to open collector outputs with 4.7k pull-up resistors. Each line can sink about 7 mA while maintaining TTL levels. Each pin is also connected to an input buffer for read back. With the open collector output off, the pin can be used as an input. The drawing at the end of this section illustrates how to use these outputs. Notice that D2 is not inverted.

When D4 is set, an interrupt request #7 (IRQ7) will be generated when pin 10 makes a high-to-low transition. Do not enable the interrupt unless pin 10 is properly driven (pin 10 has no pull-up resistor). The status of pin 10 may be read.

	D7	D6	D5	D4	D3	D2	D1	D0
Header Pin Number	17	15	13	11	9	7	5	3
DB-25 Pin Number	9	8	7	6	5	4	3	2

Read Address 378H

The 8-bit output port 378H may also be read back at the same address. This allows you to read the data you have written. External devices must not drive these lines or damage may occur. To simplify your code, OR this byte with the bits you wish to set.

	D7	D6	D5	D4	/D3	D2	/D1	/D0
Header Pin Number				IRQ7 Enable	8	6	2	1
DB-25 Pin Number				IRQ7 Enable	17	16	14	1

Read Address 37AH (These pins may be outputs or inputs.)

These four pins are connected to open collector outputs with 4.7k pull-up resistors. If the appropriate value is written to the open collector outputs (see the following figure), then the pins may be used as inputs. Switches to ground may be connected directly to these pins since 4.7k pull-up resistors are provided on the board. Notice that some inputs are inverted while D2 is not.

	/D7	D6	D5	D4	D3	D2	/D1	/D0
Header Pin Number	21	19	23	25	4			
DB-25 Pin Number	11	10	12	13	15			

Read Address 379H

These five pins are inputs only and have no internal pull-up resistors. Any device connected to these pins must ensure that the logic level goes to a true low (<0.8V) as well as high (>2.0V). An external pull-up resistor must be connected from the input to +5V when open collector devices or switches are used to ground the input.

	DB-25	Header	Header	DB-25	
*Write 37AH bit /D0	1	1	2	14	*Write 37AH bit /D1
Write 378H bit D0	2	3	4	15	
Write 378H bit D1	3	5	6	16	*Write 37AH bit D2
Write 378H bit D2	4	7	8	17	*Write 37AH bit /D3
Write 378H bit D3	5	9	10	18	Ground
Write 378H bit D4	6	11	12	19	Ground
Write 378H bit D5	7	13	14	20	Ground
Write 378H bit D6	8	15	16	21	Ground
Write 378H bit D7	9	17	18	22	Ground
	10	19	20	23	Ground
	11	21	22	24	Ground
	12	23	24	25	Ground
	13	25	26		

Outputs

* Indicates an open collector output which can also be used as an input.

	DB-25	Header	Header	DB-25	
*Read 37A bit /D0	1	1	2	14	*Read 37AH bit /D1
Read back 378H bit D0	2	3	4	15	Read 379H bit D3
Read back 378H bit D1	3	5	6	16	*Read 37AH bit D2
Read back 378H bit D2	4	7	8	17	*Read 37AH bit /D3
Read back 378H bit D3	5	9	10	18	Ground
Read back 378H bit D4	6	11	12	19	Ground
Read back 378H bit D5	7	13	14	20	Ground
Read back 378H bit D6	8	15	16	21	Ground
Read back 378H bit D7	9	17	18	22	Ground
Read 379H bit D6 /INT	10	19	20	23	Ground
Read 379H bit D7	11	21	22	24	Ground
Read 379H bit D5	12	23	24	25	Ground
Read 379H bit D4	13	25	26		

Inputs

* Indicates an open collector output which can also be used as an input.

Using the IDE hard disk Interface for Parallel I/O

The SBC has an AT IDE hard disk interface. This consists of a bi-directional data port and some address and control lines. If the IDE disk drive is not used, then the data port can be used for parallel input by reading port location 1F0H. Port 1F0H may be used as a single byte wide port or as a 16 bit port when executing a 16 bit input. The port will not function correctly if you try to read the upper 8 bits at 1F1H. The standard version of this port has frequent data outputs. Contact your CDI applications engineer for custom versions that are input only.

The SBC BIOS contains the IDE hard disk driver software. You will want to tell the BIOS that you have no drive installed.

The input bits follow TTL level conventions. They have no pull-up resistors, so any device connected to these pins must ensure that the logic level goes to a true low (<0.8V) as well as high (>2.0V). An external pull-up resistor must be connected from the input to +5V when open collector devices or switches are used to ground the input.

An interrupt request line (IRQ14) is also brought to the IDE hard disk interface connector. It can be used to signal the CPU to accept data on the data bus. The interrupt can also be used to alert the CPU when an important external event happens. An interrupt will be generated when pin 10 transitions from low to high.

Two address lines are available letting you multiplex momentary (normally off) switches. They can be used to strobe the 2 rows of a bank of 16 normally open switches. See the following section for information on reading multiplexed (X-Y) keypads.

	Header	Header	
	1	2	Ground
Read 1F0H bit D7	3	4	Read 1F0H bit D8
Read 1F0H bit D6	5	6	Read 1F0H bit D9
Read 1F0H bit D5	7	8	Read 1F0H bit D10
Read 1F0H bit D4	9	10	Read 1F0H bit D11
Read 1F0H bit D3	11	12	Read 1F0H bit D12
Read 1F0H bit D2	13	14	Read 1F0H bit D13
Read 1F0H bit D1	15	16	Read 1F0H bit D14
Read 1F0H bit D0	17	18	Read 1F0H bit D15
Ground	19	20	
	21	22	Ground
	23	24	Ground
	25	26	Ground
	27	28	
	29	30	Ground
Interrupt IRQ14	31	32	/IOCS16
(Address bit A1)	33	34	
(Address bit A0)	35	36	(Address bit A2)
/CS0	37	38	/CS1
	39	40	Ground

IDC Connector Pinout

Do not connect to undefined pins.

If you will be using the port for 16 bit inputs, the /IOCS16 signal will have to be driven low during the input. If you are not using the PC bus expansion you can directly connect /CS0 to /IOCS16. If you are using the PC bus expansion, you will need to drive the /IOCS16 with a open collector or tri-state driver.

Multiplexed Keypad Pseudocode

This pseudocode illustrates, first, how to force all rows low and read the columns to determine if any key has been pressed. Then the program shows how to step through the rows, one at a time, testing until the exact key pressed is determined. This is shown for simplicity and is by no means optimized. Delays between keys, key rollover and bounce delays have not been addressed.

----- SUBROUTINE TO TEST FOR ANY KEYPRESS -----

OUTPUT 0 (Force all rows low)
READ COL INPUT (Read column signals, assign to COL)
COL < FH? (If NO, then no key was pressed and return to the calling program.)

----- SUBROUTINE TO QUALIFY COLUMN D0 KEYS -----

OUTPUT EH (Force row D0 low, all others high)
READ COL INPUT (Read column signals, assign to COL)
COL = EH? (If YES, then "F" key pressed, and exit)
COL = DH? (If YES, then "B" key pressed, and exit)
COL = BH? (If YES, then "7" key pressed, and exit)
COL = 7H? (If YES, then "3" key pressed, and exit)

----- SUBROUTINE TO QUALIFY COLUMN D1 KEYS -----

OUTPUT DH (Force row D1 low, all others high)
READ COL INPUT (Read column signals, assign to COL)
COL = EH? (If YES, then "E" key pressed, and exit)
COL = DH? (If YES, then "A" key pressed, and exit)
COL = BH? (If YES, then "6" key pressed, and exit)
COL = 7H? (If YES, then "2" key pressed, and exit)

----- SUBROUTINE TO QUALIFY COLUMN D2 KEYS -----

OUTPUT BH (Force row D2 low, all others high)
READ COL INPUT (Read column signals, assign to COL)
COL = EH? (If YES, then "D" key pressed, and exit)
COL = DH? (If YES, then "9" key pressed, and exit)
COL = BH? (If YES, then "5" key pressed, and exit)
COL = 7H? (If YES, then "1" key pressed, and exit)

----- SUBROUTINE TO QUALIFY COLUMN D3 KEYS -----

OUTPUT 7H (Force row D3 low, all others high)
READ COL INPUT (Read column signals, assign to COL)
COL = EH? (If YES, then "C" key pressed, and exit)
COL = DH? (If YES, then "8" key pressed, and exit)
COL = BH? (If YES, then "4" key pressed, and exit)
COL = 7H? (If YES, then "0" key pressed, and exit)
RETURN (All possibilities should have been covered and the program should never hit this
return.)

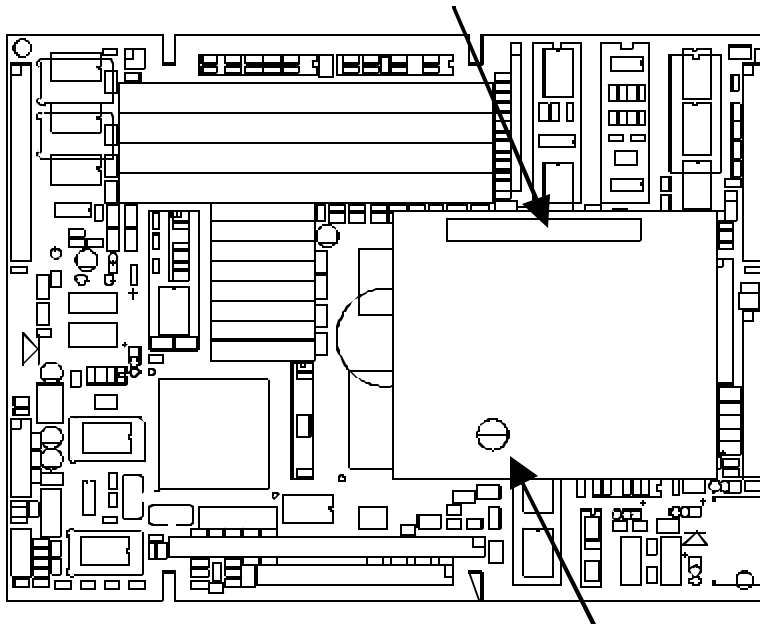
SBX EXPANSION BOARDS

The SBC can be expanded through either its PC/AT bus connector or through its SBX connectors. Although the PC bus is a viable possibility, it leaves the engineer with questions of how to mount a PC-compatible board having no standard mounting holes. Even the best of choices rarely offers a happy solution.

The SBX connector allows you to add a daughterboard and function without a mechanical engineering degree. Each SBX board snaps onto the base board connector and is then secured with a nylon screw.

Hundreds of different SBX boards are available from dozens of different suppliers. Most boards are single-width (3.70 x 2.85 in) modules and plug into 8bit processors. Most modules are designed for industrial environments and tested to 50 degrees C. If you have unique needs, SBX modules are very easily designed requiring no chip select circuitry or buffering.

SBX connector provides a positive snap mount.



Nylon mounting screw and spacer section

SBX Mounting

SBX Origins

The SBX (Small Board expansion) concept was designed by Intel to add single functions to its line of Multibus I boards. A simple and inexpensive (\$100 - \$500) expansion method was needed for its expensive (\$3000) bus boards. Intel designed the modules in both single- and double-width modules and 8- and 16-bit data paths. 8-bit modules can even plug into 16-bit sockets.

To broaden the line of SBX boards, Intel turned it into an open architecture, publishing the timing and physical specifications in their manual: #142686-002.

SBX Modules From Computer Dynamics

Computer Dynamics offers a line of cost-effective expansion modules designed for industrial and data acquisition applications. These boards are designed for wide temperature swings and offer multiple functions. The following boards are currently offered by Computer Dynamics.

CDX-P48	48 TTL parallel I/O lines software programmable as input or output in four 8-bit groups and four 4-bit groups. Directly interfaces to OPTO-22 solid-state relay racks. Uses two 82C55A ICs.
CDX-CTC/PIO	Ten 16-bit counter/timers which are fully programmable to measure events, pulse width, or frequency to 7 MHz. They may also generate frequencies, variable duty cycles, or programmable delays. 24 TTL parallel I/O lines are also provided which are software programmable as input or output in 2-byte and 2-nibble groups. CTCs and PIO are directly OPTO-22 relay rack compatible.
CDX-AD8/16	This analog-to-digital expansion module allows the CPU to read inputs in either 5 or 12 ms with 12-bit resolution. The on-board instrumentation amplifier lets you select from 1-1000 gain. Order either the 8 differential input or 16 single-ended input version.
CDX-MEM/CLK	This module allows you to add up to 384 kbytes of RAM or 256 kbytes of EPROM as well as a real-time clock.

Designing Your Own SBX Board

You will normally be able to find the expansion module you need without resorting to building it yourself. However, cost sensitive applications with unique functions may need customized I/O. The SBX technique is the easiest method for the do-it-yourselfer. SBX modules have the following attributes:

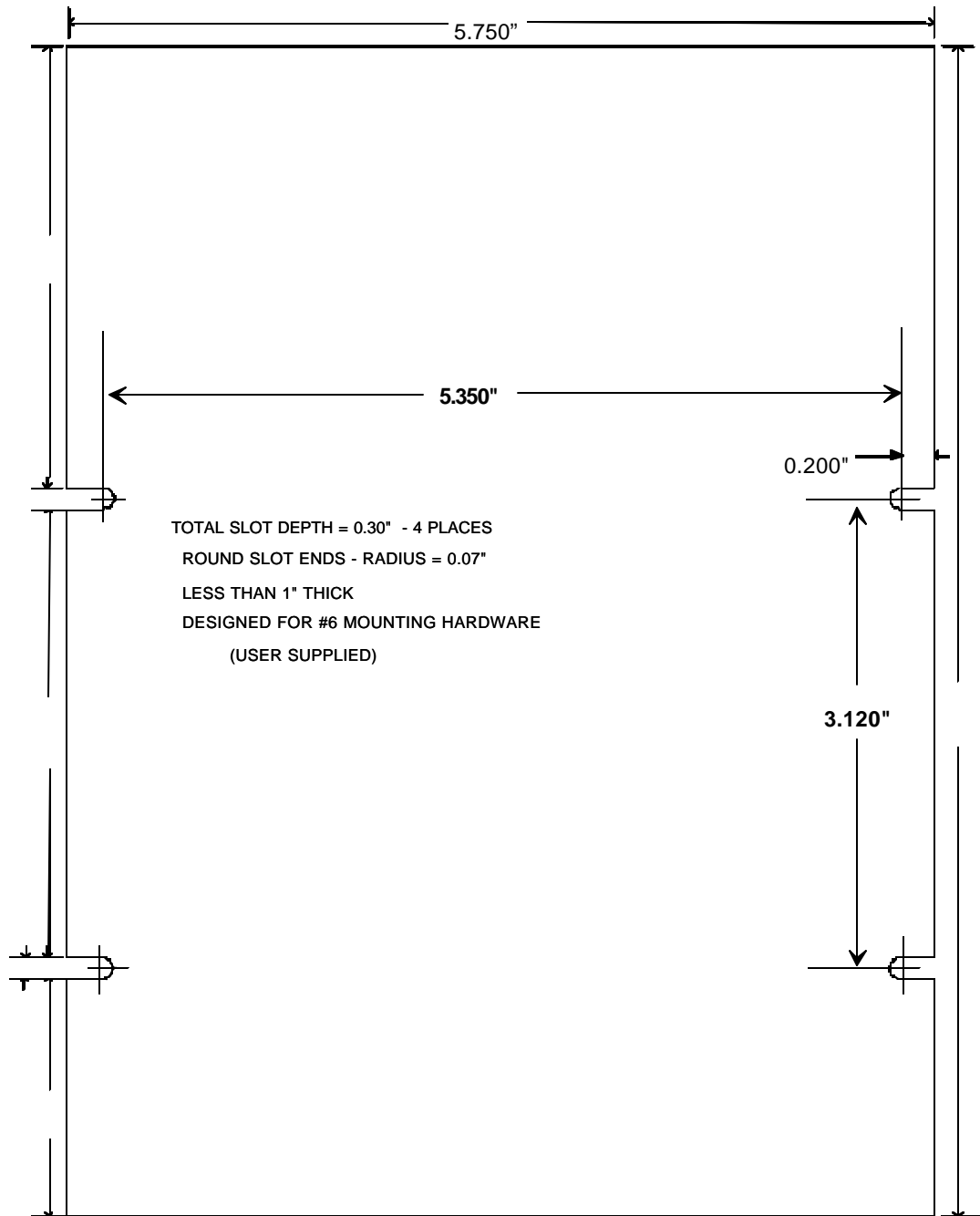
Buffering	Data and address lines are rarely buffered since most TTL and HCT devices can easily drive the bus. It is difficult to put enough loads on the board to tax the system.
Decoding	Two chip select lines (MCS0 and MCS1) are provided by the host board. Each chip select line is active for 8 consecutive addresses. A0-A2 can be used by on-board chips for further selection or they can be ignored. Read and write signals should be used to clock signals into and out of the module.
Power	Ground and +5 Vdc are provided on three sets of pins. Additionally, +12V and -12V (if provided) are sent from the base board.
Mounting	The SBX connector rigidly holds the daughterboard in place with a snapping action. This allows the daughterboard to maintain a 0.50" board-to-board spacing. Additionally, a nylon spacer and screw holds the other side of the board in place.
Size	These small boards (3.70" x 2.85") offer enough room for two 40-pin ICs, four 20-pin ICs, and connectors.

MATING CONNECTORS

Reference Designator	Function	CDI Part Number	Manufacturer's Part Number
J1	PC Bus Expansion	7CRF0-0020-6400	AMP 1-499997-2
J2	AT Bus Expansion	7CRF0-0020-4000	AMP 499997-9
J3	Power	7CRF0-0004-2000	Molex 15-24-2008
J4	Keyboard/Speaker	7CRF0-0020-1000	AMP 499997-1
J5	SBX	7CBM0-0036-0100	Viking 000292-0003
J6	Reset Switch	7CRF0-2100-0200 (shell) * 7CRF0-2100-0000 (pin)	AMP 640440-2
J7	IDE Hard Disk Drive	7CRF0-0020-4000	AMP 499997-9
J8	Printer	7CRF0-0020-2600	AMP 499997-6
J9	VGA Monitor	7CRF0-0020-1600	AMP 499997-3
J10	Floppy Drive	7CRF0-0020-3400	AMP 499997-8
J11	COM2	7CRF0-0020-1000	AMP 499997-1
J12	COM1	7CRF0-0020-1000	AMP 499997-1
J13	Panels	7CRF0-0020-4000	AMP 499997-9
J14	External Power	7CRF0-2100-0600 (shell) * 7CRF0-2100-0000 (pin)	AMP 640440-6
J16	Fanned Heat Sink	7CRF0-2100-0200 (shell) * 7CRF0-2100-0000 (pin)	AMP-640440-2
J17	External Battery	7CRF0-2100-0300 (shell) * 7CRF0-2100-0000 (pin)	AMP 640440-3

* Note these connectors have separate pins which must be crimped to the signal wire and then inserted into the shell.

MECHANICAL OUTLINE




```

** Author          : GSA
**
** Creation Date   : Fri May 29 15:35:59 1992
**
** Compiler Env.   : Turbo C++
**
** Build Cmd Line  : Make
**
** Usage           : mode_a
**
**
**                (c) Copyright Lines Unlimited
**                1992 All Rights Reserved
**
** This software is copyrighted by Lines Unlimited, and shall remain the
** property of Lines Unlimited. The licensee may make copies only for
** backup purposes. There is no run-time license fee when the software
** accesses only products manufactured by Computer Dynamics, Inc.
**
**
** =====
*/

#include <stdio.h>
#include <dos.h>          /* for the outportb() command */

#define P_EOI_REG  0x20      /* Primary End Of Interrupt Register */
#define S_EOI_REG  0xA0      /* Secondary End of Interrupt Register */
#define S_8259_MASK 0xA1     /* Secondary 8259 mask register */

/*
** Per the SBC manual (interrupt mapping section), the SBX interrupt
** line is connected to IRQ 10.
*/
#define SBX_IRQ      10      /* SBX on IRQ 10 */
#define SBX_INT_VEC  0x72     /* Interrupt Vector for IRQ 10 */

/*
** The following is defined in the SBC manual
*/
#define SBX_INT_SOURCE 0x4740

/*
** The following is defined in the SBC manual I/O mapping tables.
** With the CDX-COM board, Channel 0 is the decoded by SBX /MCS1 and
** channel 1 is decoded by SBX /MCS0.
*/
#define SBX_C0_BASE 0x748     /* SBX Channel 0 base port is at 0x748 */
#define SBX_C1_BASE 0x740     /* SBX Channel 1 base port is at 0x740 */

/*
** definitions of port addresses where the channel 0 UART is decoded.
** Base address is per the SBC manual. Registers decoded below
** are per the CDX-COM manual. C0_ is for serial channel 0 and C1_ is
** for serial channel 1.
*/
#define C0_RBR      (SBX_C0_BASE + 0) /* Rx buf(RO) - if LCR(7) == 0 */
#define C0_THR      (SBX_C0_BASE + 0) /* Tx hold(WO)- if LCR(7) == 0 */
#define C0_DLL      (SBX_C0_BASE + 0) /* Dev Lth LSB- if LCR(7) == 1 */
#define C0_DLM      (SBX_C0_BASE + 1) /* Dev Lth MSB- if LCR(7) == 1 */

```

```

#define C0_IER      (SBX_C0_BASE + 1)  /* Int Enable - if LCR(7) == 0 */
#define C0_IIR      (SBX_C0_BASE + 2)  /* Int ID register */
#define C0_LCR      (SBX_C0_BASE + 3)  /* Line Control Register */
#define C0_MCR      (SBX_C0_BASE + 4)  /* Modem Control Register */
#define C0_LSR      (SBX_C0_BASE + 5)  /* Line Status Register */
#define C0_MSR      (SBX_C0_BASE + 6)  /* Modem Status Register */
#define C0_SCRATCH  (SBX_C0_BASE + 7)  /* scratch / unused */

/*
** definitions of port addresses where the channel 1 UART is decoded
*/
#define C1_RBR      (SBX_C1_BASE + 0)  /* Rx buf(RO) - if LCR(7) == 0 */
#define C1_THR      (SBX_C1_BASE + 0)  /* Tx hold(WO)- if LCR(7) == 0 */
#define C1_DLL      (SBX_C1_BASE + 0)  /* Dev Lth LSB- if LCR(7) == 1 */
#define C1_DLM      (SBX_C1_BASE + 1)  /* Dev Lth MSB- if LCR(7) == 1 */
#define C1_IER      (SBX_C1_BASE + 1)  /* Int Enable - if LCR(7) == 0 */
#define C1_IIR      (SBX_C1_BASE + 2)  /* Int ID register */
#define C1_LCR      (SBX_C1_BASE + 3)  /* Line Control Register */
#define C1_MCR      (SBX_C1_BASE + 4)  /* Modem Control Register */
#define C1_LSR      (SBX_C1_BASE + 5)  /* Line Status Register */
#define C1_MSR      (SBX_C1_BASE + 6)  /* Modem Status Register */
#define C1_SCRATCH  (SBX_C1_BASE + 7)  /* scratch / unused */

/*
** Function prototypes
*/
void interrupt far SBX_isr( void); /* our ISR for SBX interrupts */
void (interrupt far *Original_isr)(); /* pointer to original SBX isr */
/* prior to us taking it over */

/*
** Global variables
*/
int counter = 0;
int channel = 0;
char small_buf;

/*
** =====
** name : main()
**
** synopsis: main routine for sample program for CDI CDX-COM board
** mounted on an SBC
**
** entry : void
**
** return : void
**
** globals : none
**
** =====
*/

void main( void)
{
    int original_mask = 0;

    /*
    ** DISABLE INTERRUPTS BEFORE SETTING ANYTHING UP.

```

```

*/
disable();

/*
**
** initialize UART on CDX-COM Board for channel 0
**
** -----
**
** enable DLL & DLM. Then set baud rate via DLL & DLM. The
** CDX-COM has a 1,8432Mhz Crystal so writing a 12 to DLL
** (and a 0 to DLM) sets the baud rate to 9600. (Per Appendix A
** in the CDX-COM manual)
*/
outportb( C0_LCR, 0x80); /* enable access to DLL & DLM */
outportb( C0_DLM, 0 );
outportb( C0_DLL, 12 );

/*
** set data width = 8, stop bits = 2, parity = even,
** and disable DLL & DLM (bit mapping is given in Appendix A
** of the CDX-COM Manual)
*/
outportb( C0_LCR, 0x3F );

/*
** for this example we only want the RX data Ready interrupt
** enabled. (bit mapping is given in Appendix A
** of the CDX-COM Manual)
*/
outportb( C0_IER, 0x01 );

/*
** enable interrupt mode for the chip, channel 0. Also enable
** DTR and RTS. (bit mapping is given in Appendix A
** of the CDX-COM Manual)
*/
outportb( C0_MCR, 0x0b );

/*
** Chip setup is now complete for channel 0
** -----
**
** Now set channel 1 the same way. Comments in this
** section have not been reproduced since the only thing that
** changes is the channel port locations.
**
** initialize everything the same as channel 0
*/
outportb( C1_LCR, 0x80);          /* enable access to DLL & DLM */
outportb( C1_DLM, 0 );          /* baud rate */
outportb( C1_DLL, 12 );
outportb( C1_LCR, 0x3F );       /* data bits, stop bits, etc */
outportb( C1_IER, 0x01 );       /* receive int only enabled */
outportb( C1_MCR, 0x0b );       /* set master interrupt enable */

/*

```

```

** Now setup is done for Channel 0 and for Channel 1 of the
** CDX-COM board in Mode A.
** -----
*/

/*
**
** setup new interrupt vectors for this board and save the old
** ones as follows:
**
**
** get current interrupt controller mask for the secondary
** interrupt controller and save it
*/
original_mask = inportb( S_8259_MASK);

/*
** enable (via clearing) the bitmask for the IRQ we are using
** We are using IRQ 10 so the correct mask is 0xFB. A generic
** method to get the mask is as follows:
**
**      outportb( S_8259_MASK, (original_mask & ~(1 << (SBX_IRQ - 8))))
**
** i.e.   normalize the IRQ on the secondary controller to the
**        appropriate bit on a local interrupt controller. Since
**        there are 8 interrupt lines on the primary controller,
**        we need to subtract 8 from our IRQ number.
**
**        (SBX_IRQ - 8)
**
**        So IRQ 10 - 8 == 2. To get a bit enabled in bit 2 (3rd
**        bit) we shift a 1 twice.
**
**        (1 << (SBX_IRQ - 8))
**
**        Now that we have an enable bit in the correct position,
**        we have to logically NOT the value ( a 0 enables each
**        interrupt bit. a 1 disables the interrupt)
**
**        ~(1 << (SBX_IRQ - 8))
**
**        Now we AND it with the original mask to get the new
**        mask.
**
**        (original_mask & ~(1 << (SBX_IRQ - 8)))
**
**        At the same time, we output it to the correct port and
**        we don't need a temporary variable.
**
**      outportb( S_8259_MASK, (original_mask & ~(1 << (SBX_IRQ - 8))))
**
**
** The same can be achieved with the following sequence:
**
** temp_mask =      SBX_IRQ - 8;
** temp_mask =      1 << temp_mask;
** temp_mask =      ~(temp_mask);
** temp_mask &=     original_mask;
**
** outportb( S_8259_MASK, temp_mask);

```

```

**
*/
outportb( S_8259_MASK, (original_mask & 0xFB) );

/*
** Get and save the original ISR vector address
*/
Original_isr = getvect( SBX_INT_VEC);

/*
** Set our new ISR Vector
*/
setvect( SBX_INT_VEC, SBX_isr);

/*
** ISR Vector setup and 8259 Interrupt Controller setup complete
** -----
**
**
** ENABLE INTERRUPTS NOW THAT SETUP IS DONE.
*/
enable();

/*
** -----
**
** Actual Program goes here.
**
*/
clrscr(); /* clear screen */

puts("This demo allows you to echo characters received from CDX-COM");
puts("channel 0 and 1 to the screen. As the receive buffer is only 1");
puts("character deep, anything other than moderate typing may drop");
puts("characters. The added complexity of a fully implemented ring");
puts("buffer is not the goal in this sample. However, it would be");
puts("required in most applications.\n");
puts("CHARACTERS RECEIVED IN MODE A - Hit Q or q to exit\n");

while( 1)
{
    if( small_buf)
    {
        if( (small_buf == 'q') || (small_buf == 'Q') )
            break;
        else
        {
            printf(" Channel %d - %c\n",channel, small_buf);
            small_buf = '\0';
        }
    }
}

puts("\n\nDone.\n");

/*
** -----

```

```

** DISABLE INTERRUPTS SO WE CAN RESET THE INT VECTORS AND EXIT CLEANLY
*/
disable();

/*
** restore original mask to secondary interrupt controller
*/
outportb( S_8259_MASK, original_mask);

/*
** restore original ISR vector
*/
setvect( SBX_INT_VEC, Original_isr);

/*
** disable interrupt mode on CDX-COM channel 0 UART chip
*/
outportb( C0_MCR, 0);

/*
** disable interrupt mode on CDX-COM channel 1 UART chip
*/
outportb( C1_MCR, 0);

/*
** ENABLE INTERRUPTS NOW THAT ORIGINAL STATUS IS RESTORED
*/
enable();

} /* end main() */

```

```

/*
** =====
**
** name      :   SBX_isr()
**
** synopsis:   This is the interrupt subroutine which is hooked into
**             IRQ 10.  It checks for valid interrupts on channel 0
**             and channel 1 directly as specified by mode A of the
**             CDX-COM Board.
**
**             Comments below show how to nest into the ISR vector
**             instead of taking over the vector.
**
** NOTE:      Under NO circumstances should any calls to functions
**             that use the DOS interrupts be made within an
**             interrupt service routine.  This is a safeguard
**             to prevent stack corruption which would occur
**             if the ISR was called when the main program was
**             executing a DOS interrupt.  As the DOS functions
**             are not re-entrant, the DOS stack would be reset on
**             the re-entrant call, and would not be where it
**             should be on return to the interrupted DOS call.
**
** entry     :   void

```

```

**
** return : void
**
** globals : channel, small_buf
**
** =====
*/

void interrupt far SBX_isr( void)
{
/*
** For the SBC-SXE the SBX interrupt Source Port is bit mapped
** as follows:
**
** 76543210
** |  |||||
** |  |||||+--- SBX MINTR1
** |  |||||
** |  |||||+---- SBX MINTR0
** |  |||||
** |  |||||++----- Always 0
** |  |||||
** +---+----- Not Used - No valid data
**
** Depending on the interrupt strapping of the CDX-COM board (SF-3)
** the data returned in the SBX interrupt Source Port will Vary.
**
** This Sample assumes the following strapping on the CDX-COM Board:
**
**          ( SF-3 )
**
** SBX-INTR0 #-----0 INT CHANNEL 0
** SBX-INTR0 0  +--0 INT CHANNEL 1
** SBX-INTR1 0--+ 0 INT PTR
**
** i.e. channel 0 is strapped for INTR0 and channel 1 is
** strapped for INTR1.
**
** THEREFORE, FOR THIS EXAMPLE, THE SBX INTERRUPT SOURCE (lower
** two bits) RETURNS:
**
** 1 = INTERRUPT ON CHANNEL 1
** 2 = INTERRUPT ON CHANNEL 0
**
** first check channel 0 for interrupt pending
**
*/
if( inportb( SBX_INT_SOURCE) & 2)
{
/*
** valid interrupt on this channel per CDX-COM board. Verify
** that there is valid data in the UART and, if so, or this
** example put the character into a 1 character deep global
** buffer. Also set the channel variable to the channel
** that character was read from for the print out in
** the main program
**
*/
if( inportb( C0_LSR) & 1) /* data ready signal */
{
small_buf = inportb( C0_RBR);

```

```

        channel = 0;
    }
}

/*
** Now check for interrupt pending on channel 1.  There could be an
** interrupt on both channels.
*/
if( inportb( SBX_INT_SOURCE) & 1)
{
    /*
    ** for this example put the character into a 1 character deep
    ** global buffer.
    */
    if( inportb( C1_LSR) & 1) /* data ready signal */
    {
        small_buf = inportb( C1_RBR);
        channel = 1;
    }
}

/*
** issue a non-specific EOI (end of interrupt) to the interrupt
** controllers. Note that the secondary interrupt controller must
** get the EOI PRIOR to the primary interrupt controller.
**
** NOTE: if your program is not controlling both channels on the SBX
** then you may need to handle interrupt sharing.  To do this
** you will need to nest into the vector instead of taking
** over the interrupt vector.  This can be implemented as
** follows.
**
** 1) Only initialize the channel you will use in main().
** 2) Only check the channel (above) which you are using.
** 3) instead of the two outportb() commands below to issue
** the EOI's, use the following code sample:
**
** if( Original_isr)
**     (Original_isr)();
** else
**     {
**         outportb( S_EOI_REG, 0x20);
**         outportb( P_EOI_REG, 0x20);
**     }
**
** If the original ISR vector was non-zero (i.e. the vector
** is being used) then call the original ISR and allow it to
** handle the other channel and then it will issue the EOI's.
** If the original ISR vector was zero (i.e. the vector was not
** being used) then you will issue the EOI's.
**
*/
outportb( S_EOI_REG, 0x20);
outportb( P_EOI_REG, 0x20);

} /* end SBX_isr() */

```

SOFTWARE EXAMPLE - SRAM IMPLEMENTATION

This example illustrates the use of the following by writing and reading from an 8K x 8 SRAM.

- RAM/ROM Disk Bank Select
- Enable/Disable VGA Memory

```
/*
** =====
**
** Module Name      :   direct.c
**
** Module Synopsis :   This module is an example of how to directly access
**                     the battery backed SRAM for Computer Dynamics'
**                     SBC-SXE.  It will implement the video/SRAM bank
**                     swapping and the SRAM Page Select as referenced in
**                     the SBC manual.
**
**
**                     This sample assumes an 8K x 8 SRAM is in the RAM/ROM1
**                     socket.  If a test signature exists from a previous
**                     run, it will dump the data in the SRAM.  Then it
**                     will query the user for new input values and store
**                     them into the SRAM.  These will also be printed.
**                     If no signature is present upon the sample startup,
**                     it will simply prompt the user for data, write
**                     it to the SRAM, and then dump it back to the screen
**                     from the SRAM.
**
**
** Author           :   GSA
**
** Creation Date    :   Mon Jun 01 18:26:27 1992
**
** Compiler Env.    :   Turbo C++
**
** Build Cmd Line   :   tcc direct
**
** Usage           :   direct
**
**
**                 (c) Copyright Lines Unlimited
**                 1992 All Rights Reserved
**
** =====
*/

#include <stdio.h>
#include <dos.h>      /* for the outportb() & inportb() commands */

/*
** VGA enable/disable are the same port address for all REV's of
** the SBC per the SBC manual
*/
#define VGA_ENABLE      0x540
#define VGA_DISABLE     0x548
#define ADP_ENABLE_REG  0x46E8
#define ADP_DISABLE_DATA  0
#define ADP_ENABLE_DATA  8

/*
```

```

** The following information may change based on SRAM size and RAM/ROM
** socket number. This sample shows a system with an 8K x 8 SRAM in
** RAM/ROM1 socket. In the #defines below, if this does not fit
** your configuration, modify as needed. Alternate configurations are
** listed but commented out. All this information can be found in the
** SBC Manual.
**
**
**
** for SBC
*/
#define SRAM_START      0xA000
#define PAGE_SELECT    0x760
#define PAGE_0         0x01

/*
** Offset from SRAM_START depends on SRAM size. This information
** is based on the SBC manual, in the section titled: RAM/ROM Page
** Select
**
** for 8k x 8 SRAMS
*/
#define SRAM_OFFSET    0x2000      /* 8k x 8          */

/*
** for 32k x 8 or above SRAMS
*/
/* -----
| #define SRAM_OFFSET    0x0000      /* 32k X 8 or above */
|-----*/

/*
** prototypes
*/
void    SRAM_on( void);
void    SRAM_off( void);

/*
** This is a sample structure to use for testing. Since SRAM memory
** that is being paged out doesn't need to be "allocated", we can define
** what the data format is and then define a pointer to somewhere in SRAM
** and use the pointer.
*/
struct test
{
    int    signature;
    char   type;
    int    value1;
    int    value2;
} far *SRAM;

/*
** this is a test signature for the sample. Its
** primary purpose is to show the battery backup capability
** if the board is strapped appropriately.
*/
#define CDI_SIG      1234

```

```

/*
** =====
**
** name      :   main()
**
** synopsis:   see module header
**
** entry     :   void
**
** return    :   void
**
** globals  :   none
**
** =====
*/

void main( void)
{
    /*
    ** This is a local variable used for copying the SRAM data into
    ** prior to printing to the screen.  Since we can't access SRAM
    ** and video at the same time, data which needs to be printed
    ** must be copied to a temp variable while SRAM is enabled, and then
    ** the temp can be printed after the video is re-enabled (SRAM
    ** disabled).
    */
    struct test local;

    clrscr();          /* clear screen */

    /*
    ** Initialize the structure pointer to the first byte of SRAM in the
    ** absolute memory location specified in the SBC-SXE Manual
    */
    SRAM = MK_FP( (unsigned)SRAM_START, (unsigned)SRAM_OFFSET);

    /*
    ** disable interrupts, enable SRAM, disable video
    */
    SRAM_on();

    /*
    ** If a signature is in the SRAM from a previous run of this
    ** sample then copy it to the local struct so it can be printed.
    */
    if( SRAM->signature == CDI_SIG)
        {
            local.signature = SRAM->signature;
            local.type      = SRAM->type;
            local.value1    = SRAM->value1;
            local.value2    = SRAM->value2;
        }

    /*
    ** disable SRAM, enable video
    */
    SRAM_off();
}

```

```

/*
** print previous values if a signature was found
*/
if( local.signature == CDI_SIG)
{
printf("\nSignature found from previous run of this sample.\n");
printf("\nsignature = %d\ntype      = %c\nvalue1   = %d"
      "\nvalue2   = %d\n",
      local.signature,
      local.type,
      local.value1,
      local.value2 );
}

/*
** prompt user for test data
*/
printf("\n\n");
printf("Please enter a character-->");
local.type = getchar();

printf("\nPlease enter an integer value-->");
scanf( "%d", &local.value1);

printf("\nPlease enter another integer value-->");
scanf( "%d", &local.value2);

/*
** disable interrupts, enable SRAM, disable video
*/
SRAM_on();

/*
** write the users test data into the structure located in SRAM.
*/
SRAM->signature = CDI_SIG;
SRAM->type      = local.type;
SRAM->value1    = local.value1;
SRAM->value2    = local.value2;

/*
** disable SRAM, enable video
*/
SRAM_off();

/*
** clear the temp variables.  Since for this demo we are
** printing SRAM data to the screen, the data to be printed
** must be loaded into temp variables because we cannot access
** the video and the SRAM at the same time.
*/
local.signature = 0;
local.type      = '\0';
local.value1    = 0;
local.value2    = 0;

/*
** switch into SRAM and copy users input to temp variables
**

```

```

** NOTE: No video access may be done while SRAM is
**       enabled (video disabled)
*/
SRAM_on();

/*
** write SRAM data to temp variables so they can be printed
*/
local.signature = SRAM->signature;
local.type      = SRAM->type;
local.value1    = SRAM->value1;
local.value2    = SRAM->value2;

/*
** SRAM off, video on
*/
SRAM_off();

printf("\n\nThe following NEW data has been written to and read back\n");
printf("from the SRAM...\n");
printf("\nsignature = %d\ntype      = %c\nvalue1    = %d"
       "\nvalue2    = %d\n",
       local.signature,
       local.type,
       local.value1,
       local.value2 );

} /* end main() */

/*
** =====
**
** name      :   SRAM_on()
**
** synopsis:   Enable the SRAM, Disable the Video, and select page 0 of
**             the SRAM. Also disables interrupts.
**
**           NOTE: this does NOT re-enable interrupts when finished.
**                 A call to SRAM_off will re-enable the interrupts.
**                 While the SRAM is enabled, all interrupts must be
**                 disabled.
**
** entry    :   void
**
** return   :   void
**
** globals :   none
**
** =====
*/

void SRAM_on( void)
{
/*
** disable interrupts; ANYTIME ACCESS TO SRAM VIA THE VGA-DISABLE
** PAGE SWAPPING IS DONE, INTERRUPTS MUST BE DISABLED. ALSO NO VIDEO

```

```

** MAY BE USED.
*/
disable();          /* CLI (assembly) */

/*
** disable video (enable SRAM) by writing anything
** out to the VGA_DISABLE port
*/
outportb( VGA_DISABLE, 0);

/*
** disable Western Digital video adapter. HERE, THE DATA WRITTEN
** TO THE PORT DOES MATTER.
*/
outportb( ADP_ENABLE_REG, ADP_DISABLE_DATA);

/*
** switch the SRAM to the correct bank, (in an 8K x 8 there is only
** one Page - see the SBC manual, RAM/ROM Page Select table)
*/
outportb( PAGE_SELECT, PAGE_0);

} /* end SRAM_on() */

/*
** =====
**
** name      :   SRAM_off()
**
** synopsis:   disables the SRAM, enables the video, and re-enables the
**             interrupts.
**
** entry     :   void
**
** return    :   void
**
** globals  :   none
**
** =====
*/

void SRAM_off( void)
{
/*
** switch back to video (disable SRAM) by writing anything out
** to the VGA_ENABLE port.
*/
outportb( VGA_ENABLE, 0xff);

/*
** re-enable the Western Digital video adapter.  HERE, THE VALUE
** WRITTEN TO THE PORT MATTERS.
*/
outportb( ADP_ENABLE_REG, ADP_ENABLE_DATA);

/*

```

```
** re-enable interrupts
*/
enable();          /* STI (assembly) */

} /* end SRAM_off() */
```



COMPUTER DYNAMICS INCORPORATED
 7640 Pelham Rd., Greenville, SC 29615
 Phone: (864) 627-8800

WARRANTY

CDI products are warranted for a period of one year from the date of purchase against all defects in materials and workmanship provided they are properly used and not modified by non-CDI personnel. Subassemblies and items not manufactured by CDI (power supplies, disk drives, etc.) are warranted for the period established by their original manufacturer. CDI will repair or replace the product, provided that it is returned promptly to CDI at the owner's expense. Prior to returning a component or subsystem, the purchaser must obtain a Return Material Authorization number (RMA#) from CDI. All board level products are shipped in an antistatic bag to prevent damage to the electronic components due to electrostatic discharge. Failure to use the bag in shipment will VOID the warranty. No other warranty is expressed or implied.

DISCLAIMER

CDI makes no representation or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, CDI reserves the right to revise the prices or specifications and to make any changes from time to time in the contents hereof without obligation of CDI to notify any person of such revisions or changes.

To Our Customers:

It is our intention to provide you with accurate and useful information about our product. Although the information is correct to the best of our knowledge, we cannot assume responsibility for inaccuracies within the manual.

We request that you inform us of any errors found, areas difficult to understand or suggestions to improve this manual. Please fill out the bottom portion (using additional sheets if necessary) with your comments and return it to CDI.

Thank you.

Name: _____
 Company: _____
 Address: _____

 Phone: _____
 Product Type: _____

Computer Dynamics, Inc.
 7640 Pelham Rd.
 Greenville, S.C. 29615
 Phone: (864) 627-8800

Card Serial No. _____

COMMENTS: