# Sun™ Ultra™ 60 Product Note

THE NETWORK IS THE COMPUTER™

**Sun Microsystems Computer Company**
A Sun Microsystems, Inc. Business
901 San Antonio Road
Palo Alto, CA 94303-4900 USA
650 960-1300 fax 650 969-9131

Part No.: 805-3683-10
Revision A, November 1997

# Sun Ultra 60 Product Note

This product note contains changes to Sun Ultra 5/10 documentation that became known after release of the documentation. It contains the following sections:

# ShowMe How Audio Button Problem

This section of the product note describes a problem you may experience when using the audio buttons in the ShowMe™ How™ multimedia documentation on some Ultra™ 60 systems. The problem is caused by alias settings in the system's `.cshrc` file.

## Problem Description

The audio buttons in the ShowMe How multimedia documentation might not work correctly if the shell from which you invoked ShowMe How has an alias that contains file name range-substitution functions.

For example, adding the line

```
alias lh ls -a .[a-z] [a-z]*
```

to your `.cshrc` file could cause the audio buttons in ShowMe How not to work correctly.

## Problem Resolution

To fix this problem:

1. **Check which alias statements are included in your shell's `.cshrc` file:**

   At a command line prompt within the shell from which you are trying to run ShowMe How, type `alias` and press Return.

   All alias statements in the `.cshrc` file are listed.

2. **Use an editor to remove any alias statements that contain file name range substitutions from the `.cshrc` file (and from any additional files it sources).**

3. **Save your changes and close the `.cshrc` file.**

4. **Type `source .cshrc` at the command prompt.**

5. **Restart ShowMe How.**

# POST Screen Changes

This section of the product note contains changes to the power-on self-test (POST) screens that were made after release of the *Sun Ultra 60 Service Manual* (805-1709-10).

This section of the product note contains updated Sections 3.4.1 and 3.4.2 from the *Sun Ultra 60 Service Manual.*

## 3.4.1 `diag-level` Variable Set to `max`

When the `diag-level` variable is set to `max`, POST enables an extended set of diagnostic-level tests. This mode requires approximately 2 minutes and 15 seconds to complete (with 128 Mbytes of DIMM installed). CODE EXAMPLE 3-1 identifies a typical serial port A POST output with the `diag-level` variable set to `max`.

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max`

```
Executing Power On SelfTest

0>
0>@(#) Sun Ultra 60(UltraSPARC-II 2-way) UPA/PCI POST x.x.x xx/xx/
xxxx xx:xx PM
0>INFO: Processor 0 is master.
0>
0> <00> Init System BSS
0> <00> NVRAM Battery Detect Test
0> <00> NVRAM Scratch Addr Test
0> <00> DMMU TLB Tag Access Test
0> <00> DMMU TLB RAM Access Test
0> <00> IMMU TLB Tag Access Test
0> <00> IMMU TLB RAM Access Test
0> <00> Probe Ecache
0>INFO:CPU 296 MHz: 2048KB Ecache
0> <00> Ecache RAM Addr Test
0> <00> Ecache Tag Addr Test
0> <00> Ecache Tag Test
0> <00> Invalidate Ecache Tags
0>INFO: Processor 2 - UltraSPARC-II.
0> <00> Init SC Regs
0> <00> SC Address Reg Test
0> <00> SC Reg Index Test
0> <00> SC Regs Test
0> <00> SC Dtag RAM Addr Test
```

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max` *(Continued)*

```
 0> <00> SC Cache Size Init
 0> <00> SC Dtag RAM Data Test
 0> <00> SC Dtag Init
 0> <00> Probe Memory
 0>INFO:128MB Bank 0
 0>INFO:  0MB Bank 1
 0>INFO:  0MB Bank 2
 0>INFO:  0MB Bank 3
 0> <00> Malloc Post Memory
 0> <00> Init Post Memory
 0> <00> Post Memory Addr Test
 0> <00> Map PROM/STACK/NVRAM in DMMU
 0> <00>Memory Stack Test
 2> <00> DMMU TLB Tag Access Test
 2> <00> DMMU TLB RAM Access Test
 2> <00> IMMU TLB Tag Access Test
 2> <00> IMMU TLB RAM Access Test
 2> <00> Probe Ecache
 2>INFO:CPU 296 MHz: 2048KB Ecache
 2> <00> Ecache RAM Addr Test
 2> <00> Ecache Tag Addr Test
 2> <00> Ecache Tag Test
 2> <00> Invalidate Ecache Tags
 2> <00> Map PROM/STACK/NVRAM in DMMU
 2> <00> Update Slave Stack/Frame Ptrs
 0> <00> DMMU Hit/Miss Test
 0> <00> IMMU Hit/Miss Test
 0> <00> DMMU Little Endian Test
 0> <00> IU ASI Access Test
 0> <00> FPU ASI Access Test
 2> <00> DMMU Hit/Miss Test
 2> <00> IMMU Hit/Miss Test
 2> <00> DMMU Little Endian Test
 2> <00> IU ASI Access Test
 2> <00> FPU ASI Access Test
 2> <00> Dcache RAM Test
 2> <00> Dcache Tag Test
 2> <00> Icache RAM Test
 2> <00> Icache Tag Test
 2> <00> Icache Next Test
 2> <00> Icache Predecode Test
 0> <1f> Init Psycho
 0> <1f> PIO Read Error, Master Abort Test
 0> <1f> PIO Read Error, Target Abort Test
 0> <1f> PIO Write Error, Master Abort Test
 0> <1f> PIO Write Error, Target Abort Test
 0> <1f> Timer Increment Test
```

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max` *(Continued)*

```
0> <1f> Consistent DMA UE ECC Rd Err Lpbk Test
0> <1f> Pass-Thru DMA UE ECC Rd Err Lpbk Test
0> <00> Copy Post to Memory
0> <00> Ecache Thrash Test
0> <00> Init Memory
0> <00> Memory Addr w/ Ecache Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> Block Memory Addr Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> ECC Memory Addr Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> Memory Status Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> V9 Instruction Test
0> <00> CPU Tick and Tick Compare Reg Test
0> <00> CPU Soft Trap Test
0> <00> CPU Softint Reg and Int Test
2> <00> V9 Instruction Test
2> <00> CPU Tick and Tick Compare Reg Test
0> <1f> Init Psycho
0> <1f> Psycho Cntl and UPA Reg Test
0> <1f> Psycho DMA Scoreboard Reg Test
0> <1f> Psycho Perf Cntl Reg Test
0> <1f> PIO Decoder and BCT Test
0> <1f> PCI Byte Enable Test
0> <1f> Counter/Timer Limit Regs Test
0> <1f> Timer Reload Test
0> <1f> Timer Periodic Test
0> <1f> Mondo Int Map (short) Reg Test
0> <1f> Mondo Int Set/Clr Reg Test
0> <1f> Psycho IOMMU Regs Test
0> <1f> Psycho IOMMU RAM Address Test
0> <1f> Psycho IOMMU CAM Address Test
0> <1f> IOMMU TLB Compare Test
0> <1f> IOMMU TLB Flush Test
```

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max` *(Continued)*

```
 0> <1f> Stream Buff A Control Reg Test
 0> <1f> Psycho ScacheA Page Tag Addr Test
 0> <1f> Psycho ScacheA Line Tag Addr Test
 0> <1f> Psycho ScacheA RAM Addr Test
 0> <1f> Psycho ScacheA Error Status NTA Test
 0> <1f> Psycho ScacheB Page Tag Addr Test
 0> <1f> Psycho ScacheB Line Tag Addr Test
 0> <1f> Psycho ScacheB RAM Addr Test
 0> <1f> Psycho ScacheB Error Status NTA Test
 0> <1f> PBMA PCI Config Space Regs Test
 0> <1f> PBMA Control/Status Reg Test
 0> <1f> PBMA Diag Reg Test
 0> <1f> PBMB PCI Config Space Regs Test
 0> <1f> PBMB Control/Status Reg Test
 0> <1f> PBMB Diag Reg Test
 0> <00> FPU Regs Test
 0> <00> FPU Move Regs Test
 0> <00> FPU State Reg Test
 0> <00> FPU Functional Test
 0> <00> FPU Trap Test
 0> <00> DMMU Primary Context Reg Test
 0> <00> DMMU Secondary Context Reg Test
 0> <00> DMMU TSB Reg Test
 0> <00> DMMU Tag Access Reg Test
 0> <00> DMMU VA Watchpoint Reg Test
 0> <00> DMMU PA Watchpoint Reg Test
 0> <00> IMMU TSB Reg Test
 0> <00> IMMU Tag Access Reg Test
 0> <00> DMMU TLB Tag Access Test
 0> <00> DMMU TLB RAM Access Test
 0> <00> Dcache RAM Test
 0> <00> Dcache Tag Test
 0> <00> Icache RAM Test
 0> <00> Icache Tag Test
 0> <00> Icache Next Test
 0> <00> Icache Predecode Test
 2> <00> FPU Regs Test
 2> <00> FPU Move Regs Test
 2> <00> FPU State Reg Test
 2> <00> FPU Functional Test
 2> <00> FPU Trap Test
 2> <00> DMMU Primary Context Reg Test
 2> <00> DMMU Secondary Context Reg Test
 2> <00> DMMU TSB Reg Test
 2> <00> DMMU Tag Access Reg Test
 2> <00> DMMU VA Watchpoint Reg Test
 2> <00> DMMU PA Watchpoint Reg Test
```

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max` *(Continued)*

```
 2> <00> IMMU TSB Reg Test
 2> <00> IMMU Tag Access Reg Test
 2> <00> DMMU TLB Tag Access Test
 2> <00> DMMU TLB RAM Access Test
 0> <00> CPU Addr Align Trap Test
 0> <00> DMMU Access Priv Page Test
 0> <00> DMMU Write Protected Page Test
 0> <1f> Init Psycho
 0> <1f> Pri CE ECC Error Test
 0> <1f> Pri UE ECC Error Test
 0> <1f> Pri 2 bit w/ bit hole UE ECC Err Test
 0> <1f> Pri 3 bit UE ECC Err Test
 0> <1f> Streaming DMA UE ECC Rd Err Ebus Test
 0> <1f> Streaming DMA CE ECC Rd Err Ebus Test
 0> <1f> Streaming DMA CE ECC Rd Err Lpbk Test
 0> <1f> Consistent DMA UE ECC Rd Error Ebus Test
 0> <1f> Consistent DMA UE ECC R/M/W Err Ebus Test
 0> <1f> Consistent DMA UE ECC R/M/W Err Lpbk Test
 0> <1f> Consistent DMA CE ECC Rd Err Ebus Test
 0> <1f> Consistent DMA CE ECC Rd Err Lpbk Test
 0> <1f> Consistent DMA CE ECC R/M/W Err Ebus Test
 0> <1f> Consistent DMA CE ECC R/M/W Err Lpbk Test
 0> <1f> Consistent DMA Wr Data Parity Err Lpbk Test
 0> <1f> Pass-Thru DMA UE ECC Rd Err Ebus Test
 0> <1f> Pass-Thru DMA UE ECC R/M/W Err Ebus Test
 0> <1f> Pass-Thru DMA UE ECC R/M/W Err Lpbk Test
 0> <1f> Pass-Thru DMA CE ECC Rd Err Ebus Test
 0> <1f> Pass-Thru DMA CE ECC Rd Err Lpbk Test
 0> <1f> Pass-Thru DMA CE ECC R/M/W Err Ebus Test
 0> <1f> Pass-Thru DMA CE ECC R/M/W Err Lpbk Test
 0> <1f> Pass-Thru DMA Write Data Parity Err, Lpbk Test
 0> <1f> Init Psycho
 0> <1f> Mondo Generate Interrupt Test
 0> <1f> Timer Interrupt Test
 0> <1f> Timer Interrupt w/ periodic Test
 0> <1f> Psycho Stream Buff A Flush Sync Test
 0> <1f> Psycho Stream Buff B Flush Sync Test
 0> <1f> Psycho Stream Buff A Flush Invalidate Test
 0> <1f> Psycho Stream Buff B Flush Invalidate Test
 0> <1f> Psycho Merge Buffer w/ Scache A Test
 0> <1f> Psycho Merge Buffer w/ Scache B Test
 0> <1f> Consist DMA Rd, IOMMU miss Ebus Test
 0> <1f> Consist DMA Rd, IOMMU miss Lpbk Test
 0> <1f> Consist DMA Rd, IOMMU hit Ebus Test
 0> <1f> Consist DMA Rd, IOMMU hit Lpbk Test
 0> <1f> Consist DMA Wr, IOMMU miss Ebus Test
 0> <1f> Consist DMA Wr, IOMMU miss Lpbk Test
```

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max` *(Continued)*

```
0> <1f> Consist DMA Wr, IOMMU hit Ebus Test
0> <1f> Consist DMA Wr, IOMMU hit Lpbk Test
0> <1f> Stream DMA Rd, IOMMU miss, Scache Miss Ebus Test
0> <1f> Stream DMA Rd, IOMMU miss, Scache Miss Lpbk Test
0> <1f> Stream DMA Rd, IOMMU hit, Scache Miss Ebus Test
0> <1f> Stream DMA Rd, IOMMU hit, Scache Miss Lpbk Test
0> <1f> Stream DMA Rd, IOMMU Miss, Scache(prev rd) Hit Ebus Test
0> <1f> Stream DMA Rd, IOMMU Miss, Scache Hit (prev rd) Lpbk Test
0> <1f> Stream DMA Rd, IOMMU Hit, Scache Hit Ebus Test
0> <1f> Stream DMA Rd, IOMMU Hit, Scache Hit (prev rd) Lpbk Test
0> <1f> Stream DMA Rd, IOMMU Miss, Scache Hit(prev wr) Ebus Test
0> <1f> Stream DMA Rd, IOMMU Miss, Scache Hit (prev wr) Lpbk Test
0> <1f> Stream DMA Rd, IOMMU Hit, Scache Hit(prev wr) Ebus Test
0> <1f> Stream DMA Rd, IOMMU Hit, Scache Hit (prev wr) Lpbk Test
0> <1f> Stream DMA Wr, IOMMU miss, Scache Miss Ebus Test
0> <1f> Stream DMA Wr, IOMMU miss, Scache Miss Lpbk Test
0> <1f> Stream DMA Wr, IOMMU hit, Scache Miss Ebus Test
0> <1f> Stream DMA Wr, IOMMU hit, Scache Miss Lpbk Test
0> <1f> Stream DMA Wr, IOMMU Miss, Scache(prev rd) Hit Ebus Test
0> <1f> Stream DMA Wr, IOMMU Miss, Scache(prev rd) Hit Lpbk Test
0> <1f> Stream DMA Wr, IOMMU Hit, Scache(prev rd) Hit Ebus Test
0> <1f> Stream DMA Wr, IOMMU Hit, Scache(prev rd) Hit Lpbk Test
0> <1f> Stream DMA Wr, IOMMU Miss, Scache(prev wr) Hit Ebus Test
0> <1f> Stream DMA Wr, IOMMU Miss, Scache(prev wr) Hit Lpbk Test
0> <1f> Stream DMA Wr, IOMMU Hit, Scache(prev wr) Hit Ebus Test
0> <1f> Stream DMA Wr, IOMMU Hit, Scache(prev wr) Hit Lpbk Test
0> <1f> Pass-Thru DMA Rd, Ebus device Test
0> <1f> Pass-Thru DMA Rd, Loopback Mode Test
0> <1f> Pass-Thru DMA Wr, Ebus device Test
0> <1f> Pass-Thru DMA Wr, Loopback Mode Test
0> <1f> Consist DMA Rd, IOMMU LRU Lock Ebus Test
0> <1f> Consist DMA Rd, IOMMU LRU Lock Lpbk Test
0> <1f> Stream DMA Rd, IOMMU LRU Lock, Scache LRU Lock Ebus Test
0> <1f> Stream DMA Rd, IOMMU LRU Lock, Scache LRU Lock Lpbk Test
0> <1f> Stream DMA Rd, IOMMU miss, Scache LRU Lock Ebus Test
0> <1f> Stream DMA Rd, IOMMU Miss, Scache LRU Lock Lpbk Test
0> <1f> Stream DMA Rd, IOMMU Hit, Scache LRU Lock Ebus Test
0> <1f> Stream DMA Rd, IOMMU Hit, Scache LRU Lock Lpbk Test
0> <1f> Stream DMA Rd, IOMMU LRU Lock, Scache Miss Ebus Test
0> <1f> Stream DMA Rd, IOMMU LRU Lock, Scache Miss Lpbk Test
0> <1f> Consist DMA Wr, IOMMU LRU Locked Ebus Test
0> <1f> Consist DMA Wr, IOMMU LRU Lock Lpbk Test
0> <1f> Stream DMA Wr, IOMMU LRU Lock, Scache LRU Lock Ebus Test
0> <1f> Stream DMA Wr, IOMMU LRU Lock, Scache LRU Lock Lpbk Test
0> <1f> Stream DMA Wr, IOMMU Miss, Scache LRU Lock Ebus Test
0> <1f> Stream DMA Wr, IOMMU Miss, Scache LRU Lock Lpbk Test
0> <1f> Stream DMA Wr, IOMMU Hit, Scache LRU Lock Ebus Test
```

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max` *(Continued)*

```
0> <1f> Stream DMA Wr, IOMMU Hit, Scache LRU Lock Lpbk Test
0> <1f> Stream DMA Wr, IOMMU LRU Lock, Scache Miss Ebus Test
0> <1f> Stream DMA Wr, IOMMU LRU Lock, Scache Miss Lpbk Test
0> <1f> Stream DMA Wr, IOMMU LRU Lock, Scache(prev rd) Hit Ebus
Test
0> <1f> Stream DMA Wr, IOMMU LRU Lock, Scache(prev rd) Hit Lpbk
Test
0> <00> Init Memory
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> Memory w/ Ecache Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> Block Memory Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> ECC Blk Memory Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> UltraSPARC-2 Prefetch Instructions Test
0> <00>Test 0: prefetch_mr
0> <00>Test 1: prefetch to non-cacheable page
0> <00>Test 2: prefetch to page with dmmu misss
0> <00>Test 3: prefetch miss does not check alignment
0> <00>Test 4: prefetcha with asi 0x4c is noped
0> <00>Test 5: prefetcha with asi 0x54 is noped
0> <00>Test 6: prefetcha with asi 0x6e is noped
0> <00>Test 7: prefetcha with asi 0x76 is noped
0> <00>Test 8: prefetch with fcn 5
0> <00>Test 9: prefetch with fcn 2
0> <00>Test 10: prefetch with fcn 12
0> <00>Test 11: prefetch with fcn 16 is noped
0> <00>Test 12: prefetch with fcn 29 is noped
0> <00>Test 13: prefetcha with asi 0x15 is noped
0> <00>Test 14: prefetch with fcn 3
0> <00>Test 15: prefetcha14 with fcn 2
0> <00>Test 16: prefetcha80_mr
0> <00>Test 17: prefetcha81_1r
0> <00>Test 18: prefetcha10_mw
```

**CODE EXAMPLE 3-1** `diag-level` Variable Set to `max` *(Continued)*

```
0> <00>Test 19: prefetcha80_17 is noped
0> <00>Test 20: prefetcha10_6: illegal instruction trap
0> <00>Test 21: prefetcha11_1w
0> <00>Test 22: prefetcha81_31
0> <00>Test 23: prefetcha11_15: illegal instruction trap
2> <00> UltraSPARC-2 Prefetch Instructions Test
2> <00>Test 0: prefetch_mr
2> <00>Test 1: prefetch to non-cacheable page
2> <00>Test 2: prefetch to page with dmmu misss
2> <00>Test 3: prefetch miss does not check alignment
2> <00>Test 4: prefetcha with asi 0x4c is noped
2> <00>Test 5: prefetcha with asi 0x54 is noped
2> <00>Test 6: prefetcha with asi 0x6e is noped
2> <00>Test 7: prefetcha with asi 0x76 is noped
2> <00>Test 8: prefetch with fcn 5
2> <00>Test 9: prefetch with fcn 2
2> <00>Test 10: prefetch with fcn 12
2> <00>Test 11: prefetch with fcn 16 is noped
2> <00>Test 12: prefetch with fcn 29 is noped
2> <00>Test 13: prefetcha with asi 0x15 is noped
2> <00>Test 14: prefetch with fcn 3
2> <00>Test 15: prefetcha14 with fcn 2
2> <00>Test 16: prefetcha80_mr
2> <00>Test 17: prefetcha81_1r
2> <00>Test 18: prefetcha10_mw
2> <00>Test 19: prefetcha80_17 is noped
2> <00>Test 20: prefetcha10_6: illegal instruction trap
2> <00>Test 21: prefetcha11_1w
2> <00>Test 22: prefetcha81_31
2> <00>Test 23: prefetcha11_15: illegal instruction trap
0>STATUS =PASSED


Power On Selftest Completed
```

## 3.4.2 `diag-level` Variable Set to `min`

When the `diag-level` variable is set to `min`, POST enables an abbreviated set of diagnostic-level tests. This mode requires approximately 1 minute and 30 seconds to complete (with 128 Mbytes of DIMM installed). CODE EXAMPLE 3-2 identifies a serial port A POST output with the `diag-level` NVRAM variable set to `min`.

**CODE EXAMPLE 3-2** `diag-level` Variable Set to `min`

```
Executing Power On SelfTest

0>
0>@(#) Sun Ultra 60(UltraSPARC-II 2-way) UPA/PCI POST x.x.x xx/xx/
xxxx xx:xx PM
0>INFO: Processor 0 is master.
0>
0> <00> Init System BSS
0> <00> NVRAM Battery Detect Test
0> <00> NVRAM Scratch Addr Test
0> <00> DMMU TLB Tag Access Test
0> <00> DMMU TLB RAM Access Test
0> <00> IMMU TLB Tag Access Test
0> <00> IMMU TLB RAM Access Test
0> <00> Probe Ecache
0>INFO:CPU 296 MHz: 2048KB Ecache
0> <00> Ecache RAM Addr Test
0> <00> Ecache Tag Addr Test
0> <00> Ecache Tag Test
0> <00> Invalidate Ecache Tags
0>INFO: Processor 2 - UltraSPARC-II.
0> <00> Init SC Regs
0> <00> SC Address Reg Test
0> <00> SC Reg Index Test
0> <00> SC Regs Test
0> <00> SC Dtag RAM Addr Test
0> <00> SC Cache Size Init
0> <00> SC Dtag RAM Data Test
0> <00> SC Dtag Init
0> <00> Probe Memory
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> Malloc Post Memory
0> <00> Init Post Memory
0> <00> Post Memory Addr Test
0> <00> Map PROM/STACK/NVRAM in DMMU
0> <00>Memory Stack Test
```

**CODE EXAMPLE 3-2** `diag-level` Variable Set to `min` *(Continued)*

```
 2> <00> DMMU TLB Tag Access Test
 2> <00> DMMU TLB RAM Access Test
 2> <00> IMMU TLB Tag Access Test
 2> <00> IMMU TLB RAM Access Test
 2> <00> Probe Ecache
 2>INFO:CPU 296 MHz: 2048KB Ecache
 2> <00> Ecache RAM Addr Test
 2> <00> Ecache Tag Addr Test
 2> <00> Ecache Tag Test
 2> <00> Invalidate Ecache Tags
 2> <00> Map PROM/STACK/NVRAM in DMMU
 2> <00> Update Slave Stack/Frame Ptrs
 0> <00> DMMU Hit/Miss Test
 0> <00> IMMU Hit/Miss Test
 0> <00> DMMU Little Endian Test
 0> <00> IU ASI Access Test
 0> <00> FPU ASI Access Test
 2> <00> DMMU Hit/Miss Test
 2> <00> IMMU Hit/Miss Test
 2> <00> DMMU Little Endian Test
 2> <00> IU ASI Access Test
 2> <00> FPU ASI Access Test
 2> <00> Dcache RAM Test
 2> <00> Dcache Tag Test
 2> <00> Icache RAM Test
 2> <00> Icache Tag Test
 2> <00> Icache Next Test
 2> <00> Icache Predecode Test
 0> <1f> Init Psycho
 0> <1f> PIO Read Error, Master Abort Test
 0> <1f> PIO Read Error, Target Abort Test
 0> <1f> PIO Write Error, Master Abort Test
 0> <1f> PIO Write Error, Target Abort Test
 0> <1f> Timer Increment Test
 0> <1f> Consistent DMA UE ECC Rd Err Lpbk Test
 0> <1f> Pass-Thru DMA UE ECC Rd Err Lpbk Test
 0> <00> Copy Post to Memory
 0> <00> Ecache Thrash Test
 0> <00> Init Memory
 0> <00> Memory Addr w/ Ecache Test
 0>INFO:128MB Bank 0
 0>INFO:  0MB Bank 1
 0>INFO:  0MB Bank 2
 0>INFO:  0MB Bank 3
 0> <00> Block Memory Addr Test
 0>INFO:128MB Bank 0
 0>INFO:  0MB Bank 1
```

**CODE EXAMPLE 3-2** `diag-level` Variable Set to `min` *(Continued)*

```
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> ECC Memory Addr Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> Memory Status Test
0>INFO:128MB Bank 0
0>INFO:  0MB Bank 1
0>INFO:  0MB Bank 2
0>INFO:  0MB Bank 3
0> <00> V9 Instruction Test
0> <00> CPU Tick and Tick Compare Reg Test
0> <00> CPU Soft Trap Test
0> <00> CPU Softint Reg and Int Test
2> <00> V9 Instruction Test
2> <00> CPU Tick and Tick Compare Reg Test
0> <1f> Init Psycho
0> <1f> Psycho Cntl and UPA Reg Test
0> <1f> Psycho DMA Scoreboard Reg Test
0> <1f> Psycho Perf Cntl Reg Test
0> <1f> PIO Decoder and BCT Test
0> <1f> PCI Byte Enable Test
0> <1f> Counter/Timer Limit Regs Test
0> <1f> Timer Reload Test
0> <1f> Timer Periodic Test
0> <1f> Mondo Int Map (short) Reg Test
0> <1f> Mondo Int Set/Clr Reg Test
0> <1f> Psycho IOMMU Regs Test
0> <1f> Psycho IOMMU RAM Address Test
0> <1f> Psycho IOMMU CAM Address Test
0> <1f> IOMMU TLB Compare Test
0> <1f> IOMMU TLB Flush Test
0> <1f> Stream Buff A Control Reg Test
0> <1f> Psycho ScacheA Page Tag Addr Test
0> <1f> Psycho ScacheA Line Tag Addr Test
0> <1f> Psycho ScacheA RAM Addr Test
0> <1f> Psycho ScacheA Error Status NTA Test
0> <1f> Psycho ScacheB Page Tag Addr Test
0> <1f> Psycho ScacheB Line Tag Addr Test
0> <1f> Psycho ScacheB RAM Addr Test
0> <1f> Psycho ScacheB Error Status NTA Test
0> <1f> PBMA PCI Config Space Regs Test
0> <1f> PBMA Control/Status Reg Test
0> <1f> PBMA Diag Reg Test
0> <1f> PBMB PCI Config Space Regs Test
```

**CODE EXAMPLE 3-2** `diag-level` Variable Set to `min` *(Continued)*

```
0> <1f> PBMB Control/Status Reg Test
0> <1f> PBMB Diag Reg Test
0> <00> UltraSPARC-2 Prefetch Instructions Test
0> <00>Test 0: prefetch_mr
0> <00>Test 1: prefetch to non-cacheable page
0> <00>Test 2: prefetch to page with dmmu misss
0> <00>Test 3: prefetch miss does not check alignment
0> <00>Test 4: prefetcha with asi 0x4c is noped
0> <00>Test 5: prefetcha with asi 0x54 is noped
0> <00>Test 6: prefetcha with asi 0x6e is noped
0> <00>Test 7: prefetcha with asi 0x76 is noped
0> <00>Test 8: prefetch with fcn 5
0> <00>Test 9: prefetch with fcn 2
0> <00>Test 10: prefetch with fcn 12
0> <00>Test 11: prefetch with fcn 16 is noped
0> <00>Test 12: prefetch with fcn 29 is noped
0> <00>Test 13: prefetcha with asi 0x15 is noped
0> <00>Test 14: prefetch with fcn 3
0> <00>Test 15: prefetcha14 with fcn 2
0> <00>Test 16: prefetcha80_mr
0> <00>Test 17: prefetcha81_1r
0> <00>Test 18: prefetcha10_mw
0> <00>Test 19: prefetcha80_17 is noped
0> <00>Test 20: prefetcha10_6: illegal instruction trap
0> <00>Test 21: prefetcha11_1w
0> <00>Test 22: prefetcha81_31
0> <00>Test 23: prefetcha11_15: illegal instruction trap
2> <00> UltraSPARC-2 Prefetch Instructions Test
2> <00>Test 0: prefetch_mr
2> <00>Test 1: prefetch to non-cacheable page
2> <00>Test 2: prefetch to page with dmmu misss
2> <00>Test 3: prefetch miss does not check alignment
2> <00>Test 4: prefetcha with asi 0x4c is noped
2> <00>Test 5: prefetcha with asi 0x54 is noped
2> <00>Test 6: prefetcha with asi 0x6e is noped
2> <00>Test 7: prefetcha with asi 0x76 is noped
2> <00>Test 8: prefetch with fcn 5
2> <00>Test 9: prefetch with fcn 2
2> <00>Test 10: prefetch with fcn 12
2> <00>Test 11: prefetch with fcn 16 is noped
2> <00>Test 12: prefetch with fcn 29 is noped
2> <00>Test 13: prefetcha with asi 0x15 is noped
2> <00>Test 14: prefetch with fcn 3
2> <00>Test 15: prefetcha14 with fcn 2
2> <00>Test 16: prefetcha80_mr
2> <00>Test 17: prefetcha81_1r
2> <00>Test 18: prefetcha10_mw
```

**CODE EXAMPLE 3-2** `diag-level` Variable Set to `min` *(Continued)*

```
2> <00>Test 19: prefetcha80_17 is noped
2> <00>Test 20: prefetcha10_6: illegal instruction trap
2> <00>Test 21: prefetcha11_1w
2> <00>Test 22: prefetcha81_31
2> <00>Test 23: prefetcha11_15: illegal instruction trap
0>STATUS =PASSED


Power On Selftest Completed
```

# OBDiag Screen Changes

This section of the product note contains changes to the OpenBoot™ diagnostic (OBDiag) screens that were made after release of the *Sun Ultra 60 Service Manual* (805-1709-10).

This section of the product note contains updated section 4.7 (and subsections) from the *Sun Ultra 60 Service Manual.*

## 4.7 OpenBoot Diagnostics

The OpenBoot diagnostic (OBDiag) is a menu-driven set of diagnostics that verifies:

- PCI/Cheerio
- Ebus DMA/TCR registers
- Ethernet
- Keyboard
- Mouse
- Floppy
- Serial port A
- Serial port B
- NVRAM
- Audio
- SCSI
- All above

OBDiag performs root-cause failure analysis on the referenced devices by testing internal registers, confirming subsystem integrity, and verifying device functionality.

To initilize the OBDiag menu:

1. **At the** `ok` **prompt, type**: `obdiag`.

2. **Verify that the OBDiag screen is displayed (**CODE EXAMPLE 4-7**).**

**CODE EXAMPLE 4-7** OBDiag Screen

```
{0} ok obdiag
stdin: fffe1eb0
stdout: fffe1eb8
loading code into: /pci@1f,4000/ebus@1
loading code into: /pci@1f,4000/ebus@1/eeprom
loading code into: /pci@1f,4000/ebus@1/ecpp@14,3043bc
loading code into: /pci@1f,4000/ebus@1/su@14,3062f8
loading code into: /pci@1f,4000/ebus@1/se@14,400000
loading code into: /pci@1f,4000/network@1,1
loading code into: /pci@1f,4000/ebus@1/fdthree@14,3023f0
loading code into: /pci@1f,4000/ebus@1
SUNW,CS4231 Debugging enabled
```

3. **At the** `ok` **prompt, type:** `obtest`.

4. **Verify that the OBDiag menu is displayed (**CODE EXAMPLE 4-8**).**

**CODE EXAMPLE 4-8** OBDiag Menu

```
OBDiag Menu

  0 ..... PCI/Cheerio
  1 ..... EBUS DMA/TCR Registers
  2 ..... Ethernet
  3 ..... Keyboard
  4 ..... Mouse
  5 ..... Floppy
  6 ..... Parallel Port
  7 ..... Serial Port A
  8 ..... Serial Port B
  9 ..... NVRAM
 10 ..... Audio
 11 ..... SCSI
 12 ..... All Above
 13 ..... Quit
 14 ..... Display this Menu
 15 ..... Toggle script-debug
 16 ..... Enable External Loopback Tests
 17 ..... Disable External Loopback Tests

Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

5. **At the OBDiag menu prompt, type** `15` **to enable toggle script-debug messages.**

## 4.7.1 PCI/Cheerio

The PCI/Cheerio diagnostic performs the following:

1. `vendor_ID_test` – Verifies the Cheerio ASIC vendor ID is 108e.
2. `device_ID_test` – Verifies the Cheerio ASIC device ID is 1000.
3. `mixmode_read` – Verifies the PCI configuration space is accessible as half-word bytes by reading the EBus2 vendor ID address.
4. `e2_class_test` – Verifies the address class code. Address class codes include bridge device (0 x B, 0 x 6), other bridge device (0 x A and 0 x 80), and programmable interface (0 x 9 and 0 x 0).
5. `status_reg_walk1` – Performs walk-one test on status register with mask 0 x 280 (Cheerio ASIC is accepting fast back-to-back transactions, DEVSEL timing is 0 x 1).
6. `line_size_walk1` – Performs tests 1 through 5.
7. `latency_walk1` – Performs walk one test on latency timer.
8. `line_walk1` – Performs walk one test on interrupt line.
9. `pin_test` – Verifies interrupt pin is logic-level high (1) after reset.

CODE EXAMPLE 4-9 identifies the PCI/Cheerio output message.

**CODE EXAMPLE 4-9** PCI/Cheerio Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 0

TEST='all_pci/cheerio_test'
SUBTEST='vendor_id_test'
SUBTEST='device_id_test'
SUBTEST='mixmode_read'
SUBTEST='e2_class_test'
SUBTEST='status_reg_walk1'
SUBTEST='line_size_walk1'
SUBTEST='latency_walk1'
SUBTEST='line_walk1'
SUBTEST='pin_test'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.2 EBus DMA/TCR Registers

The EBus DMA/TCR registers diagnostic performs the following:

1. `DMA_reg_test` – Performs a walking ones bit test for control status register, address register, and byte count register of each channel. Verifies that the control status register is set properly.
2. `DMA_func_test` – Validates the DMA capabilities and FIFOs. Test is executed in a DMA diagnostic loopback mode. Initializes the data of transmitting memory with its address, performs a DMA read and write, and verifies that the data received is correct. Repeats for four channels.

CODE EXAMPLE 4-10 identifies the EBus DMA/TCR registers output message.

**CODE EXAMPLE 4-10** EBus DMA/TCR Registers Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 1

TEST='all_dma/ebus_test'
SUBTEST='dma_reg_test'
SUBTEST='dma_func_test'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.3 Ethernet

The Ethernet diagnostic performs the following:

1. `my_channel_reset` – Resets the Ethernet channel.
2. `hme_reg_test` – Performs walk1 on the following registers set: global register 1, global register 2, bmac xif register, bmac tx register, and the mif register.
3. `MAC_internal_loopback_test` – Performs Ethernet channel engine internal loopback.
4. `10_mb_xcvr_loopback_test` – Enables the 10Base-T data present at the transmit MII data inputs to be routed back to the receive MII data outputs.
5. `100_mb_phy_loopback_test` – Enables MII transmit data to be routed to the MII receive data path.
6. `100_mb_twister_loopback_test` – Forces the twisted-pair transceiver into loopback mode.

CODE EXAMPLE 4-11 identifies the Ethernet output message.

**CODE EXAMPLE 4-11** Ethernet Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 2

TEST='ethernet_test'
SUBTEST='my_channel_reset'
SUBTEST='hme_reg_test'
SUBTEST='global_reg1_test'
SUBTEST='global_reg2_test'
SUBTEST='bmac_xif_reg_test'
SUBTEST='bmac_tx_reg_test'
SUBTEST='mif_reg_test'
SUBTEST='mac_internal_loopback_test'
SUBTEST='10mb_xcvr_loopback_test'
SUBTEST='100mb_phy_loopback_test'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.4 Keyboard

The keyboard diagnostic consists of an external and internal loopback. The external loopback requires a passive loopback connector. The internal loopback verifies the keyboard port by transmitting and receiving 128 characters.

CODE EXAMPLE 4-12 identifies the keyboard output message.

**CODE EXAMPLE 4-12** Keyboard Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 3

TEST='keyboard_test'
SUBTEST='internal_loopback'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

### 4.7.5 Mouse

The mouse diagnostic performs a keyboard-to-mouse loopback.

CODE EXAMPLE 4-13 identifies the mouse output message.

**CODE EXAMPLE 4-13** Mouse Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 4

TEST='mouse_test'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

### 4.7.6 Floppy

The floppy diagnostic verifies the diskette drive controller initialization. It also validates the status of a selected disk drive and reads the diskette drive header.

CODE EXAMPLE 4-14 identifies the floppy output message.

**CODE EXAMPLE 4-14** Floppy Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 5

TEST='floppy_test'
SUBTEST='floppy_id0_read_test'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

### 4.7.7 Parallel Port

The parallel port diagnostic performs the following:

1. `sio_passive_lb` – Sets up the SuperIO configuration register to enable extended/compatible parallel port select, then does a write 0, walk one, write 0 x ff to the data register. It verifies the results by reading the status register.
2. `dma_read` – Enables ECP mode and ECP DMA configuration, and FIFO test mode. Transfers 16 bytes of data from memory to the parallel port device and then verifies the data is in FIFO device.

CODE EXAMPLE 4-15 identifies the parallel port output message.

**CODE EXAMPLE 4-15** Parallel Port Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 6

TEST='parallel_port_test'
SUBTEST='dma_read'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.8 Serial Port A

The serial port A diagnostic invokes the `uart_loopback test`. The `uart_loopback test` transmits and receives 128 characters and checks the transaction validity. The following baud rates are tested in asynchronous mode: 460800, 307200, 230400, 153600, 76800, 57600, 38400, 19200, 9600, 4800, 2400, and 800.

CODE EXAMPLE 4-16 identifies the serial port A output message.

**CODE EXAMPLE 4-16** Serial Port A Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 7

TEST='uarta_test'
BAUDRATE='1200'
SUBTEST='internal_loopback'
BAUDRATE='1800'
SUBTEST='internal_loopback'
BAUDRATE='2400'
SUBTEST='internal_loopback'
BAUDRATE='4800'
SUBTEST='internal_loopback'
BAUDRATE='9600'
SUBTEST='internal_loopback'
BAUDRATE='19200'
SUBTEST='internal_loopback'
BAUDRATE='38400'
SUBTEST='internal_loopback'
BAUDRATE='57600'
SUBTEST='internal_loopback'
BAUDRATE='76800'
SUBTEST='internal_loopback'
BAUDRATE='115200'
SUBTEST='internal_loopback'
BAUDRATE='153600'
```

**CODE EXAMPLE 4-16** Serial Port A Output Message *(Continued)*

```
SUBTEST='internal_loopback'
BAUDRATE='230400'
SUBTEST='internal_loopback'
BAUDRATE='307200'
SUBTEST='internal_loopback'
BAUDRATE='460800'
SUBTEST='internal_loopback'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

---

**Note –** The serial port A diagnostic will stall if the TIP line is installed on serial port A. CODE EXAMPLE 4-17 identifies the serial port A output message when the TIP line is installed on serial port A.

---

**CODE EXAMPLE 4-17** Serial Port A Output Message With TIP Line Installed

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 7

TEST='uarta_test'
'UART A in use as console - Test not run.'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.9 Serial Port B

The serial port B diagnostic is identical to the serial port A diagnostic.

CODE EXAMPLE 4-18 identifies the serial port B output message.

---

**Note –** The serial port B diagnostic will stall if the tip line is installed on serial port B.

---

**CODE EXAMPLE 4-18** Serial Port B Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 8

TEST='uartb_test'
BAUDRATE='1200'
SUBTEST='internal_loopback'
BAUDRATE='1800'
SUBTEST='internal_loopback'
BAUDRATE='2400'
SUBTEST='internal_loopback'
BAUDRATE='4800'
SUBTEST='internal_loopback'
BAUDRATE='9600'
SUBTEST='internal_loopback'
BAUDRATE='19200'
SUBTEST='internal_loopback'
BAUDRATE='38400'
SUBTEST='internal_loopback'
BAUDRATE='57600'
SUBTEST='internal_loopback'
BAUDRATE='76800'
SUBTEST='internal_loopback'
BAUDRATE='115200'
SUBTEST='internal_loopback'
BAUDRATE='153600'
SUBTEST='internal_loopback'
BAUDRATE='230400'
SUBTEST='internal_loopback'
BAUDRATE='307200'
SUBTEST='internal_loopback'
BAUDRATE='460800'
SUBTEST='internal_loopback'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.10 NVRAM

The NVRAM diagnostic verifies the NVRAM operation by performing a write and read to the NVRAM.

CODE EXAMPLE 4-19 identifies the NVRAM output message.

**CODE EXAMPLE 4-19** NVRAM Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 9

TEST='nvram_test'
SUBTEST='write/read_patterns'
SUBTEST='write/read_inverted_patterns'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.11 Audio

The audio diagnostic performs the following:

1. cs4231_test – Verifies the cs4231 internal registers.
2. Line-in to line-out external loopback.
3. Microphone to headphone external loopback.

CODE EXAMPLE 4-20 identifies the audio output message.

**CODE EXAMPLE 4-20** Audio Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 10

TEST='audio_test'
SUBTEST='cs4231_test'
Codec_ID='8a'
Version_ID='a0'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.12 SCSI

The SCSI diagnostic validates both the SCSI chip and the SCSI bus subsystem.

CODE EXAMPLE 4-21 identifies the SCSI output message.

**CODE EXAMPLE 4-21** SCSI Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 11

TEST='selftest'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

## 4.7.13 All Above

The all above diagnostic validates the system unit.

CODE EXAMPLE 4-22 identifies the all above output message.

---

**Note –** The all above diagnostic will stall if the tip line is installed on serial port A or serial port B.

---

**CODE EXAMPLE 4-22** All Above Output Message

```
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===> 12

TEST='all_pci/cheerio_test'
SUBTEST='vendor_id_test'
SUBTEST='device_id_test'
SUBTEST='mixmode_read'
SUBTEST='e2_class_test'
SUBTEST='status_reg_walk1'
SUBTEST='line_size_walk1'
SUBTEST='latency_walk1'
SUBTEST='line_walk1'
SUBTEST='pin_test'

TEST='all_dma/ebus_test'
SUBTEST='dma_reg_test'
SUBTEST='dma_func_test'

TEST='ethernet_test'
SUBTEST='my_channel_reset'
```

**CODE EXAMPLE 4-22** All Above Output Message *(Continued)*

```
SUBTEST='hme_reg_test'
SUBTEST='global_reg1_test'
SUBTEST='global_reg2_test'
SUBTEST='bmac_xif_reg_test'
SUBTEST='bmac_tx_reg_test'
SUBTEST='mif_reg_test'
SUBTEST='mac_internal_loopback_test'
SUBTEST='10mb_xcvr_loopback_test'
SUBTEST='100mb_phy_loopback_test'

TEST='keyboard_test'
SUBTEST='internal_loopback'

TEST='mouse_test'

TEST='floppy_test'
SUBTEST='floppy_id0_read_test'

TEST='parallel_port_test'
SUBTEST='dma_read'

TEST='uarta_test'
'UART A in use as console - Test not run.'

TEST='uartb_test'
BAUDRATE='1200'
SUBTEST='internal_loopback'
BAUDRATE='1800'
SUBTEST='internal_loopback'
BAUDRATE='2400'
SUBTEST='internal_loopback'
BAUDRATE='4800'
SUBTEST='internal_loopback'
BAUDRATE='9600'
SUBTEST='internal_loopback'
BAUDRATE='19200'
SUBTEST='internal_loopback'
BAUDRATE='38400'
SUBTEST='internal_loopback'
BAUDRATE='57600'
SUBTEST='internal_loopback'
BAUDRATE='76800'
SUBTEST='internal_loopback'
BAUDRATE='115200'
SUBTEST='internal_loopback'
BAUDRATE='153600'
SUBTEST='internal_loopback'
```

**CODE EXAMPLE 4-22** All Above Output Message *(Continued)*

```
BAUDRATE='230400'
SUBTEST='internal_loopback'
BAUDRATE='307200'
SUBTEST='internal_loopback'
BAUDRATE='460800'
SUBTEST='internal_loopback'

TEST='nvram_test'
SUBTEST='write/read_patterns'
SUBTEST='write/read_inverted_patterns'

TEST='audio_test'
SUBTEST='cs4231_test'
Codec_ID='8a'
Version_ID='a0'

TEST='selftest'
Enter (0-12 tests, 13 -Quit, 14 -Menu) ===>
```

# Updates to Service Manual Illustrations

This section contains illustrations that have been updated after the release of the *Sun Ultra 60 Service Manual*.

## Motherboard Serial Port Jumpers

Below is an update to Figure 10-9, which shows the position of the serial port jumpers on the motherboard.

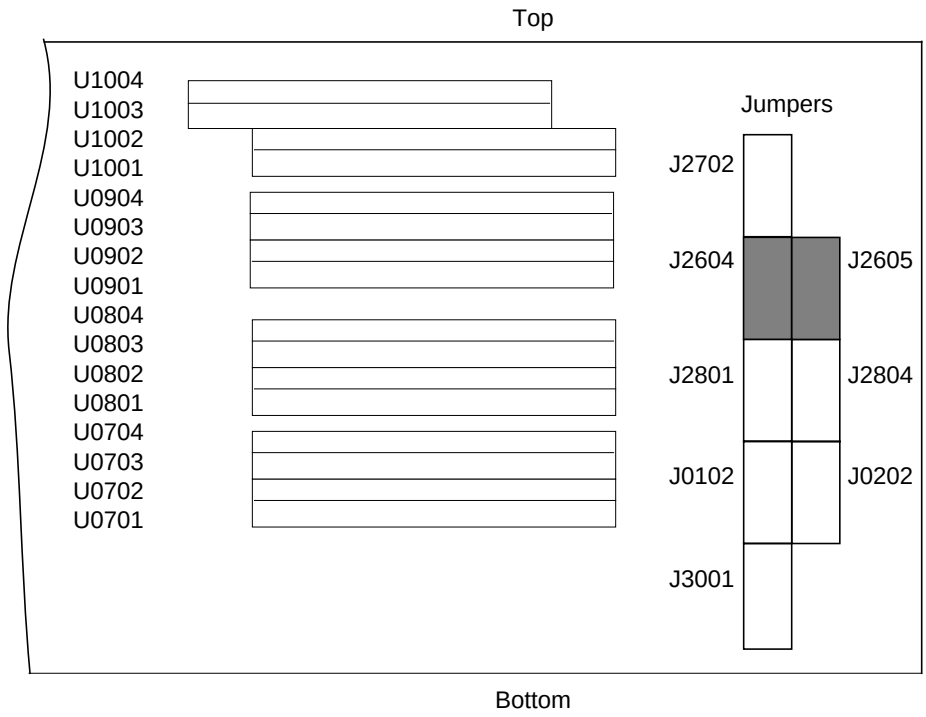


**FIGURE 10-9** Location of the Motherboard Serial Port Jumpers (J2604 and J2605)

# DIMM Banks and Slots

Below is an update to Figure C-6, which shows the position of the DIMM banks and slots on the motherboard.

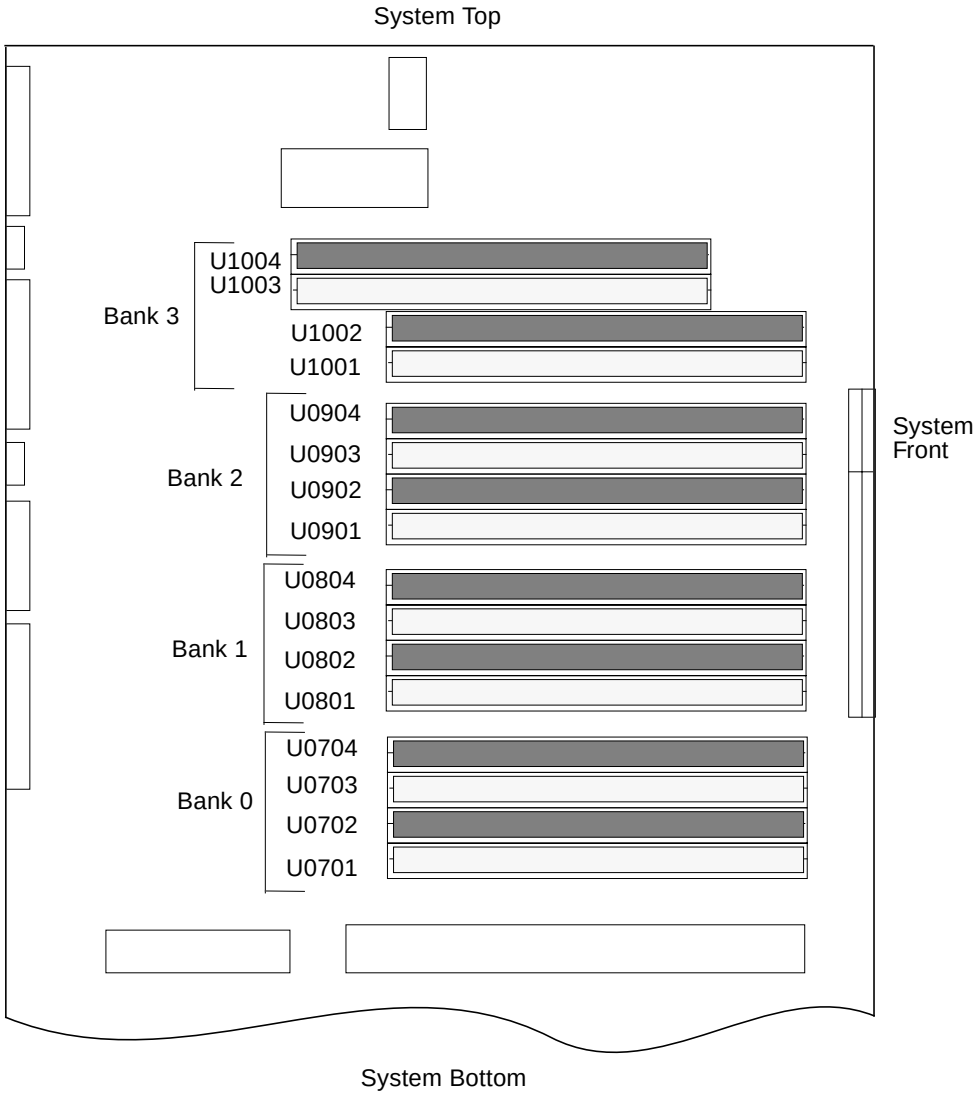


**FIGURE C-6** DIMM Banks

# Motherboard Functional Block Diagram

Below is an update to Figure C-13, which shows a functional block diagram of the system unit motherboard.

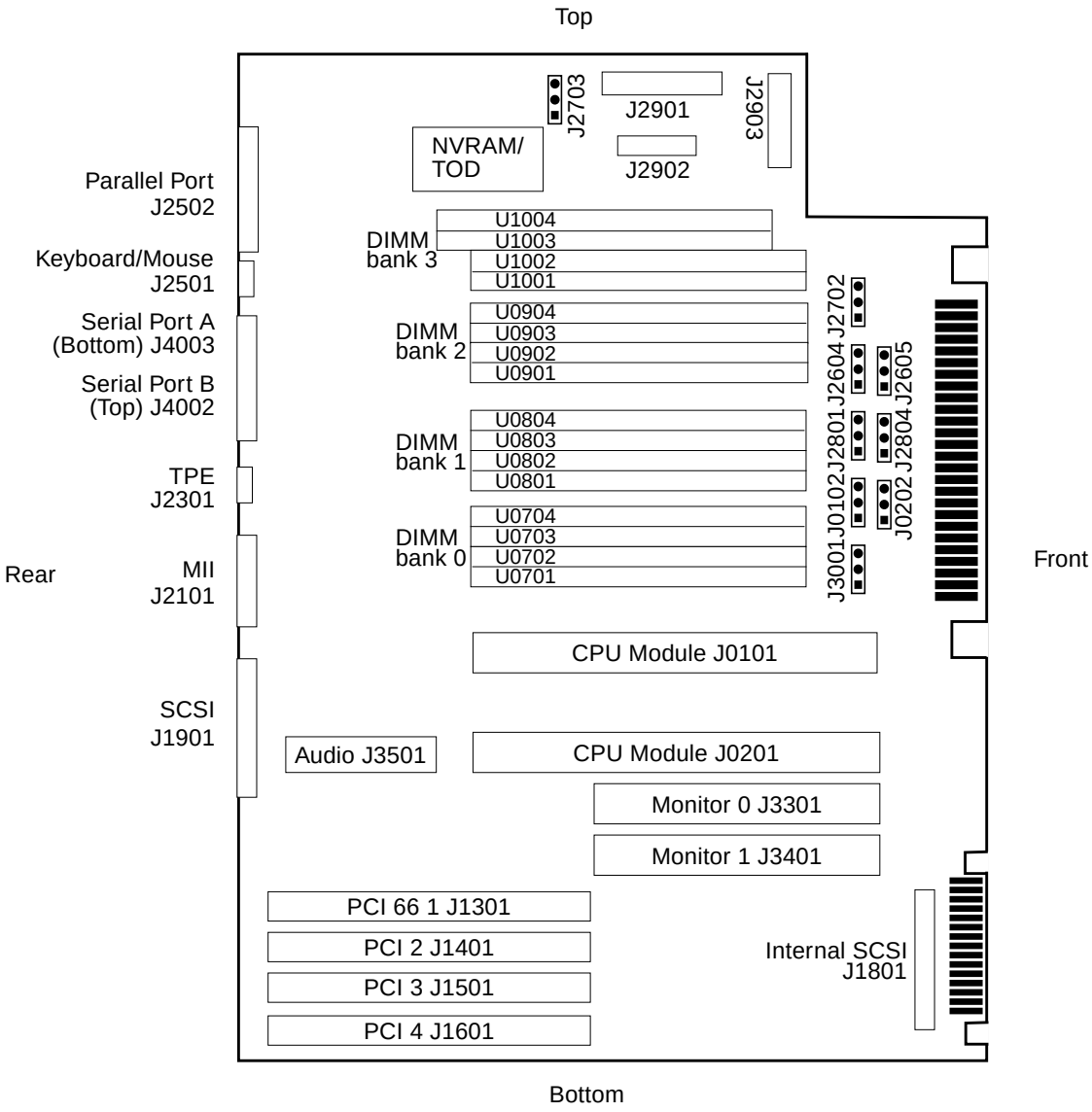


**FIGURE C-13** System Unit Motherboard Functional Block Diagram