PROJECT

32-BIT MICROPROCESSOR AND COMPUTER SYSTEM DEVELOPMENT

TITLE

DEM32 V.2 USER'S MANUAL

	Name	Function	Date	Signature
Prepared :	Khalid Ismail	Project Engineer		
Checked :	Joakim Ohlsson	Design Engineer		
	Bo Törnberg	Design Engineer		
Authorized :	Ritva Svenningsson	Project Manager		

Distribution
Complete : ALSYS/2, ESTEC/2, B2S, T/CM-BT, T/CK-JO, T/CM-TS, T/CP-KI
Summary :

		Saab Ericsson Space AB		Telephone	Telefax	Visiting address
From	Head office :	S-405 15 GÖTEBORG	SWEDEN	+46 31 35 00 00	+46 31 35 95 20	Delsjömotet
	Branch office :	S-581 88 LINKÖPING	SWEDEN	+46 13 28 64 00	+46 13 13 16 28	Bröderna Ugglas gata

Class : Industrial
Contract No :
Host System : Word 2.0c for Windows, SE Macro Rev 1.0
Host File : \Arkiv1\mcd\tnt\0021_01.doc

SUMMARY

This document is the User's Manual for the DEM32 Version 2 evaluation board developed within the 32 Bit Microprocessor project. DEM32 is a computer board with the purpose to demonstrate the radiation tolerant SPARC V7 compatible chip set of IU-RT, FPU-RT and MEC (Rev. CBA).

DOCUMENT CHANGE RECORD

Changes between issues are marked with a left-bar.

Issue	Date	Paragraphs affected	Change information
1	13 April 1996	All	New document

TABLE OF CONTENTS Page

1. INTRODUCTION 5
1.1. Abbreviations and Acronyms 5

2. DOCUMENTS 6

3. DEM32 FUNCTIONAL DESCRIPTION 7
3.1. ERC32 Core 8
3.1.1. Integer Unit (IU-RT) 9
3.1.1.1. Traps and Exceptions 9
3.1.1.2. Multitasking 9
3.1.1.3. Multiprocessing 9
3.1.1.4. Concurrent Error Detection 9
3.1.2. Floating Point Unit (FPU-RT) 10
3.1.3. Memory Controller (MEC) 10
3.1.3.1. External Bus Interface 10
3.1.3.2. Trap Handling 11
3.1.3.3. Timers 11
3.1.3.4. Serial Channels 11
3.1.3.5. Watch-Dog Timer 12
3.1.3.6. Power Down Mode 12
3.1.3.7. I/O Ports 12
3.2. Clock Oscillators 12
3.3. Ada Tasking Accelerator Coprocessor (ATAC) 12
3.4. Memory 12
3.4.1. Nominal RAM 12
3.4.2. Memory Expansion 13
3.4.3. Start-up PROM 13

4. HARDWARE INSTALLATION INSTRUCTIONS 14
4.1. Transport and handling 14
4.2. Storing and operating 14
4.3. Unpacking 14
4.4. Installing 14

5. OPERATING INSTRUCTIONS 15
5.1. Front Panel Functions 15
5.1.1. Serial Interface 15
5.1.2. Reset Button 15
5.1.3. Halt Switch 15
5.1.4. Indicators 15
5.2. DEM32 Board Layout 16
5.3. DEM32 Connectors 18
5.3.1. Power Connector 18
5.3.2. Serial Interface Connectors 19
5.3.3. Debug and Manufacturing Test Connectors 20
5.4. Hardware Configuration 23

6. HARDWARE SOFTWARE INTERFACE 24
6.1. DEM32 Memory Map 24
6.2. MEC Registers 25
6.2.1. Register Address Map 25
6.2.2. Register Configuration and Bit Allocation 26
6.3. Interrupt Allocation 41

7. RESIDENT SOFTWARE IN START-UP PROM 42
7.1. Boot Strap 42
7.2. System Self-Test 42
7.3. SPARCmon Adaptation 43
7.3.1. General 43
7.3.2. Serial Communication 43
7.3.3. Memory Organisation 43
7.3.4. SPARCmon Function Commands 44
7.3.5. Board Specific Registers in SPARCmon for DEM32 50
7.3.6. SPARCmon Enhancements for DEM32 51

1. INTRODUCTION

This document is the User's Manual for the DEM32 Version 2 evaluation board developed within the 32 Bit Microprocessor project. DEM32 is a computer board with the purpose to demonstrate the radiation tolerant SPARC V7 compatible chip set of IU-RT, FPU-RT and MEC (Rev. CBA). The User's Manual is divided into the following chapters:

- FUNCTIONAL DESCRIPTION

- HARDWARE INSTALLATION INSTRUCTIONS

- OPERATING INSTRUCTIONS

- HARDWARE SOFTWARE INTERFACE

- RESIDENT SOFTWARE IN START-UP PROM
  - Hardware Initialisation
  - System Self-test
  - SPARCMon Adaptation

NOTE: Please read the Hardware Installation Instructions carefully before unpacking the board.

1.1. Abbreviations and Acronyms

AD	Applicable Documents
ATAC	Ada TAsking Coprocessor
ERC32	32 bit Embedded Real-time Computing Core
FAR	Failing Address Register
FPU	Floating Point Unit
DEM32	DEMonstration bread board for the erc32 core
IU	Integer Unit
MEC	MEmory Controller
MEXC	Memory EXCeption
PROM	Programmable Read Only Memory
RAM	Random Access Memory
RD	Reference Document
RTC	Real Time Clock
SFSR	System Fault Status Register
UART	Universal Asynchronous Receiver Transmitter

2. DOCUMENTS

[STDV7] SPARC Standard Version 7

[MECSPC] MEC Device Specification, MCD/SPC/0009/SE, issue 1, 23 April 1996.

[IUSPC] IU-RT Preliminary Device Specification, AMS/IURT/0017/CLG, issue 6, August 28th 1995.

[FPUSPC] FPU-RT Preliminary Device Specification, AMS/FPURT/0018/VS, issue 6, August 28th 1995.

[ATAC1] ATAC 2.0 Ada Tasking Accelerating Coprocessor Data Sheet, draft issue 0.99, March 1993.

[ATAC2] ATAC 2.0 Ada Mechanism Performance, draft issue 1.0, March 1993.

[ATAC3] ATAC 2.0 Compiler Adaptation Guide, draft, October 3, 1992.

[VME] ANSI/IEEE, American National Standard, IEEE Standard for a Versatile Back Plane VME Bus, ANSI/IEEE Std 1014-1987, 28 March 1988.

[RS232] EIA Standard for RS-232C

[MON1] SPARCmon Installation Guide, Rev. A, Sun Microsystems, Inc. April 1990

[MON2] SPARCmon User's Guide, Rev. A, Sun Microsystems, Inc. April 1990

3. DEM32 FUNCTIONAL DESCRIPTION

The DEM32 is a computer board based on the ERC32 processor core. The purpose of the board is to demonstrate the ERC32 core capabilities and it is also intended to be used to develop application software. The DEM32 is an industry-standard double Europe size (160mm x 233,4 mm) printed circuit board. The board may be inserted into a standard VME crate for power supply, but power may also be supplied with a custom connector.

The DEM32 has the following characteristics:

- ERC32 computer core: IU, FPU and MEC
- 2 Mbyte RAM
- 128 kbyte start-up EEPROM (also called boot PROM)
- Ada Tasking Accelerating Coprocessor (ATAC)
- Two RS232C standard serial ports

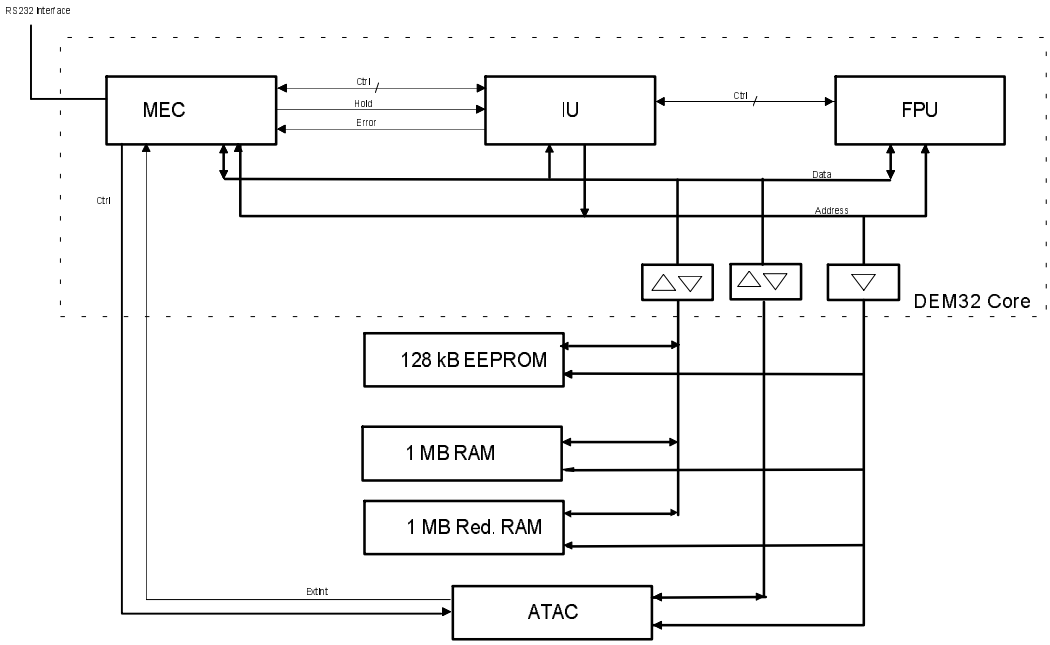

Figure 1 - DEM32 Block Diagram

3.1. ERC32 Core

The block diagram of the ERC32 core architecture is shown in Figure 2.

- Integer Unit (IU)
- Floating Point Unit (FPU)
- Address latches and data buffers
- MEmory Controller (MEC)

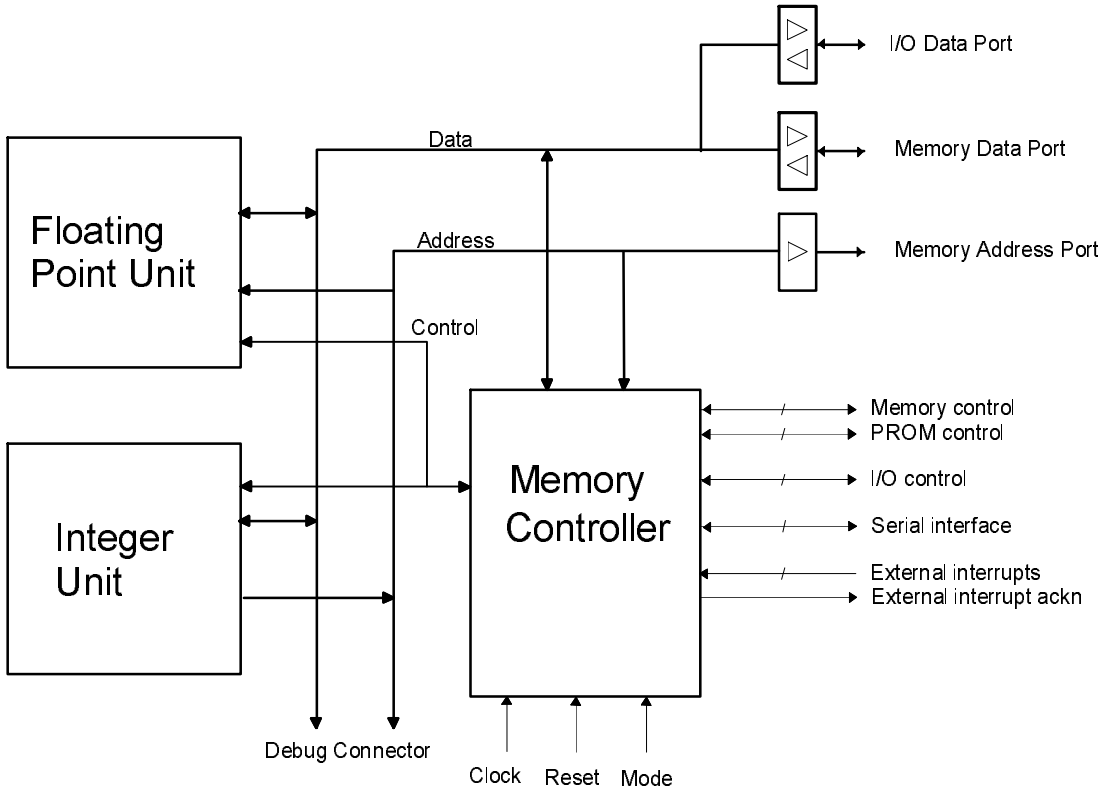

Figure 2 - ERC32 core

3.1.1. Integer Unit (IU-RT)

The IU is the primary processing engine in the SPARC architecture, executing all instructions except for specific floating-point operations. The FPU performs floating-point calculations concurrently with the IU. The architecture also allows for concurrent operation through the use of an optional second coprocessor (not implemented in the DEM32). For a detailed description of the IU, see [IUSPC].

Significant features of the IU include:

- Full binary compatibility with entire SPARC V7 application software base
- Architectural efficiency that sustains 1.25 to 1.5 clocks per instruction
- Large windowed register file
- Tightly coupled floating-point interface
- User/supervisor modes for multitasking
- Semaphore instructions and alternate address spaces for multiprocessing
- Tagged arithmetic instructions to support artificial intelligence software

3.1.1.1. Traps and Exceptions

The IU supports a full set of traps and exceptions. A table-based set of trap vectors supports 128 hardware and 128 software trap types, both synchronous (error conditions and instructions) and asynchronous (interrupts and reset). The IU supports a very fast interrupt time of 4 to 10 clocks, depending upon the contents of the instruction pipeline.

3.1.1.2. Multitasking

Multitasking is supported with user and supervisor modes. Certain privileged instructions can only be executed while the IU is in supervisor mode, ensuring that user programs cannot accidentally alter the state of the machine. Supervisor mode is only accessible by using a hardware interrupt or by executing a trap instruction.

3.1.1.3. Multiprocessing

The IU supports multiprocessing with two instructions for implementing semaphores in memory. Atomic Load/Store Unsigned Byte loads a byte from memory, then sets the memory location to all ones. The SWAP instruction exchanges the contents of a register and a memory location. Both of these instructions are "atomic", meaning not possible to interrupt.

3.1.1.4. Concurrent Error Detection

The IU supports concurrent error detection by including parity generation and checking, program flow control, and possibility to work in a master/slave configuration.

3.1.2. Floating Point Unit (FPU-RT)

The FPU provides high-performance, IEEE STD-754-1985 compatible single- and double-precision floating-point calculations for ERC32 systems, and is designed to operate concurrently with the IU. All address and control signals for memory accesses by the FPU are supplied by the IU. Floating-point instructions are addressed by the IU, and are simultaneously latched from the data bus by both the IU and FPU. Floating-point instructions are concurrently decoded by the IU and the FPU, but do not begin execution in the FPU until after the instruction is enabled by a signal from the IU. Pending and currently executing FP instructions are placed in an on-chip queue while the IU continues to execute non-floating-point instructions.

The FPU has a 32 x 32-bit data register file for floating-point operations. The contents of these registers are transferred to and from external memory under control of the floating-point load/store instructions. Addresses and control signals for data accesses during a floating-point load or store are supplied by the IU, while the FPU supplies or receives data. Although the FPU operates concurrently with the IU, a program containing floating-point computations generates results as if the instructions were being executed sequentially.

The FPU supports concurrent error detection by including parity generation and checking, and possibility to work in a master/slave configuration.

For more details on the FPU refer to the FPU device specification [FPUSPC].

3.1.3. Memory Controller (MEC)

The Memory Controller (MEC) interfaces directly with the address, data, and control buses of the IU and the FPU. It generates the system clock, reset and control signals for the processor elements, and all strobes for the memory and the Input/Output ports. Also, it recognises external interrupt inputs and includes two serial ports, UARTs. For a detailed description of the MEC refer to the MEC device specification [MECSPC].

3.1.3.1. External Bus Interface

The on-chip bus controller generates all necessary strobes required by external memory, IU and FPU, and external I/O units.

3.1.3.2. Trap Handling

The main objective for the ERC32 trap handling is to resolve internal hardware errors and memory access errors. Memory access errors are signalled by assertion of the MEXC* signal which force the IU to vector a data access exception. For the memory access errors, the MEC FAR (Failing Address Register) is the address of the last synchronous data error detected for IU or DMA.

The MEC is capable of handling nine different fault events which all result in a synchronous trap (assertion of MEXC*).

1. Parity error on control bus
2. Parity error on data bus
3. Parity error on address bus
4. Access to protected area
5. Access to unimplemented area
6. MEC register access violation
7. Uncorrectable and detected error in memory
8. Bus time-out
9. System bus error

The MEC handles fifteen different events which all represent asynchronous trap (assertion of the Interrupt Request Level IRL inputs of the IU).

1. Watch Dog time-out
2. DMA time-out
3. DMA access error
4. UART error
5. UART A data ready
6. UART B data ready
7. RTC interrupt
8. General Purpose Timer interrupt
9. Correctable error in memory
10. Masked hardware error
11. Five external interrupts

3.1.3.3. Timers

The MEC includes a timer clock divider, that supplies the timer functions with a timer clock derived from the system clock. The timers have configurable scalers and counters.

The MEC has two timers:

- One real-time clock timer.
- One general purpose timers.

3.1.3.4. Serial Channels

The MEC supplies the board with two asynchronous RS232 communication channels. These channels implement full duplex serial communication. The serial communication channel baud rate is programmable.

3.1.3.5. Watch-Dog Timer

The watch-dog function is realised as a retriggerable single shot timer with programmable time-out. A reset timer starts counting when the watch-dog timer has elapsed. The reset triggering can be avoided if the watch-dog interrupt is handled before the reset timer elapses. There is also a "trap door" function which disables the watchdog.

3.1.3.6. Power Down Mode

The MEC includes a software programmable power down mode that puts the system in a power saving mode by inactivating of the IU/FPU bus controls. The system wakes up from Power down mode if any enabled asynchronous interrupt is detected.

3.1.3.7. I/O Ports

Four general purpose I/O ports are supported by the MEC. Each port can be individually controlled by programming an internal MEC register. On the DEM32 only one port is used. This IO port is used for the ATAC interface.

3.2. Clock Oscillators

The MEC provides the IU and FPU with a system clock. This clock is derived from the external system oscillator. The oscillator operates at a frequency of 20 MHz, which means that the system clock frequency is 10 MHz.

The watchdog oscillator is connected to the WDCLK input of the MEC. The watchdog oscillator frequency is 1,8432 MHz.

3.3. Ada Tasking Accelerator Coprocessor (ATAC)

The ATAC is addressed as I/O unit 0 in the DEM32, i.e. command address 0 of the ATAC is accessed at the first memory address of I/O unit 0. The chip is operated at 10 MHz which is the system clock frequency. The RTC input of the ATAC is connected to the WDCLK signal. ATAC external memory is not supported on the DEM32. The EXTCLK input of the ATAC is not used, nor are the external interrupt inputs of the ATAC. For details about the ATAC see [ATAC1-3]. The ATTN and FAULT interrupts from the ATAC are connected to MEC external interrupt inputs, and INTACK is connected to the EXTINTACK output of the MEC.

3.4. Memory

3.4.1. Nominal RAM

The board is equipped with 2 Mbytes of nominal RAM (excluding checkbits). The RAM is implemented as one bank with four 512k x 8 utilising MEMCS*(0) signal of the MEC. Once the Memory Configuration Register of the MEC has been initialised it is seen as a continuous 2 MBytes segment by software. The RAM may be accessed at zero wait states (programmable in the MEC).

3.4.2. Memory Expansion

In addition to the 2 Mbytes of nominal RAM, an expanded RAM area is reserved in the board. The DEM32 is able to access totally 6 Mbytes, two separate nominal RAM banks of 2 Mbytes utilising MEMCS*(0) and MEMCS*(1) signals and one redundant bank of 2 Mbytes utilising MEMCS*(8) signal (excluding checkbits). As mentioned before, only one nominal RAM bank of 512 kwords and 512 kbytes check bits are implemented.

3.4.3. Start-up PROM

The board is equipped with 128 kBytes of start-up PROM, byte wide organised (8-bit option set in MEC). The PROM is realised with one 128k x 8 EEPROM allowing access with one wait state.

The start-up PROM of the DEM32 contains system start up tests and the SPARCMon monitor [MON1-2] which has been adapted to the DEM32 board (see paragraph 7.3).

4. HARDWARE INSTALLATION INSTRUCTIONS

4.1. Transport and handling

The following environmental conditions must be met when transporting and handling the DEM32:

Minimum temperature:	+10 °C
Maximum temperature:	+50 °C
Maximum relative humidity:	60 %

Great care shall be taken in order to reduce shock to a minimum.

4.2. Storing and operating

The following environmental conditions must be met during storage and during operation of the DEM32:

Minimum temperature:	+10 °C
Maximum temperature:	+30 °C
Maximum relative humidity:	60 %

No vibration and shock shall be applied.

4.3. Unpacking

When unpacking the DEM32 consider the following:

* Avoid touching areas of integrated circuitry and connector pins; static discharge can damage circuits.

* Handle the board carefully; the components on the board can be easily damaged, especially the large flat-pack components (IU-RT, FPU-RT and MEC).

4.4. Installing

The DEM32 shall be supplied with power from a back plan conforming to the VME standard [RD8]. Any standard VME crate is sufficient. The DEM32 occupies one slot in any slot position of the backplane.

Make sure no power is applied when inserting the board; inserting the DEM32 while power is applied can result in damage to the components.

5. OPERATING INSTRUCTIONS

5.1. Front Panel Functions

Figure 3 shows the front panel of the DEM32 board.

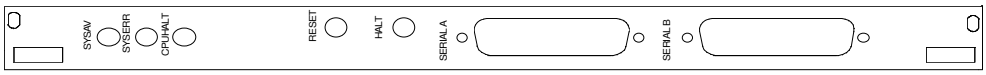

Figure 3 - DEM32 front panel

Two RS232 type serial interface connectors, used for host computer communication, are provided on the front panel of the DEM32. One halt switch is provided. The halt switch shall be set to off (left) position for normal operation. A reset push-button used for hardware reset of the DEM32. Three LED indicators show the status of the DEM32. After power-up, the SYSAV LED should be lit and the SYSERR and CPUHALT indicators should be unlit.

5.1.1. Serial Interface

The electrical characteristics for the serial interface pins comply the RS-232 standard, [RD9]. The Serial A connector corresponds to channel A of the MEC UART. Channel A is used by SPARCMon for host communication. The Serial B connector corresponds to channel B of the MEC UART. Channel B is used by SPARCMon for remote debugging. Note that no hardware handshake signals are provided. All handshaking must be implemented by software (e.g. XON/XOFF).

5.1.2. Reset Button

Pushing the reset button on the front panel will result in a hardware reset of the DEM32. A hardware reset is automatically performed when powering on the DEM32.

5.1.3. Halt Switch

There is a halt function in the DEM32. Setting the HALT switch in ON position (right) will halt execution in the DEM32 by asserting the SYSHALT signal of the MEC. During halt the IU, FPU, MEC including timers and UARTs are halted.

5.1.4. Indicators

There are three indicators on the front panel named SYSAV, SYSERR and CPUHALT. These indicators reflect the status of the corresponding pins of the MEC (see [RD7]).

SYSAV: System Availability. This LED is under software control. It is deasserted (unlit) by hardware reset. SPARCMon will assert SYSAV after successful completion of start-up tests. If SPARCMon detects any hardware error during start-up, SYSAV will remain unlit. SYSAV is automatically deasserted when CPUHALT is active.

SYSERR: System Error. This output is asserted whenever an unmasked error is detected by the MEC. This condition will normally lead to CPUHALT unless the MEC has been programmed to give reset. When running

SPARCMon, CPUHALT is default. The error can originate either from the IU, FPU or the MEC.

CPUHALT: Halt Indication. This LED reflects the status of the HALT* inputs of the IU and the FPU. The CPUHALT can be generated either from the HALT push button on the front panel, from software halt or from error halt (see SYSERR above).

5.2. DEM32 Board Layout

The DEM32 board layout is shown in Figure 4 on next page.

Figure 4 - DEM32 Board Layout

5.3. DEM32 Connectors

The DEM32 has two serial interface connectors on the front panel and one connector used to power the board from the VME backplane on the opposite side. The other connectors are used for manufacturing test and debug only. Table 1 lists all connectors of the DEM32 board.

Table 1 - DEM32 connector list

Designation	Type	Function
J101	EURO96/96	Power
J102	EURO96/96	Debug
J201	CANNON DB-25S	Serial Port A
J202	CANNON DB-25S	Serial Port B
J301	IDC 20	Test 1
J302	IDC 20	Test 2
J303	IDC 20	Test 3
J304	IDC 10	TAP I/F

Note: connectors J102, J301-304 are used for debug only.

5.3.1. Power Connector

Table 2 lists the signal names of the power connector (J101). Only the power pins of the VME-bus are connected to the DEM32.

Table 2 - Power connector (J101) pin-out

PIN NUMBER	A	B	C
1			
2			
3			
4			
5			
6			
7			
8			
9	GND		GND
10			
11	GND		
12			
13			
14			
15	GND		
16			
17	GND		
18			
19	GND		
20		GND	
21			
22			
23		GND	
24			
25			
26			
27			
28			
29			
30			
31			
32	VCC	VCC	VCC

5.3.2. Serial Interface Connectors

Table 3 provides the pin assignment for the serial interface connectors. The pin numbers are consistent with the EIA-232 standard. The DEM32 transmits data on the TxD pin and receives data on the RxD pin.

Table 3 - Serial interface connector A pinout

PIN NR.	J201
1	N.C.
2	TxD
3	RxD
4	N.C.
5	N.C.
6	N.C.
7	GND
8-25	N.C.

Table 4 - Serial interface connector B pinout

PIN NR.	J202
1	N.C.
2	TxD
3	RxD
4	N.C.
5	N.C.
6	N.C.
7	GND
8-25	N.C.

5.3.3. Debug and Manufacturing Test Connectors

Tables list the signal names of the debug and test connectors (J102, J301-J304). The debug and test connectors are only used for manufacturing test, debug and configuration of the board.

Table 5 - Debug connector (J102) pin-out

PIN NUMBER	A	B	C
1	D0	A0	GND
2	D1	A1	CLK
3	D2	A2	DMAGNT*
4	D3	A3	DPARIO
5	D4	A4	DRDY*
6	D5	A5	MDS*
7	D6	A6	MEXC*
8	D7	A7	ASI0
9	D8	A8	ASI1
10	D9	A9	ASI2
11	D10	A10	ASI3
12	D11	A11	APAR
13	D12	A12	ASPAR
14	D13	A13	DMAAS
15	D14	A14	DMAREQ*
16	D15	A15	DXFER
17	D16	A16	IMPAR
18	D17	A17	LDSTO
19	D18	A18	SIZE0
20	D19	A19	SIZE1
21	D20	A20	LOCK
22	D21	A21	RD
23	D22	A22	WRT
24	D23	A23	WE*
25	D24	A24	RESET*
26	D25	A25	MHOLD*
27	D26	A26	INST
28	D27	A27	INULL
29	D28	A28	EXMCS*
30	D29	A29	ALE*
31	D30	A30	BUSRDY*
32	D31	A31	GND

Table 6 - Test connector 1 (J301) pin-out

PIN NR.	J301
20	GND
19	MEMCS*(0)
18	MEMCS*(1)
17	MEMCS*(2)
16	MEMCS*(3)
15	MEMCS*(4)
14	MEMCS*(5)
13	MEMCS*(6)
12	MEMCS*(7)
11	MEMCS*(8)
10	MEMCS*(9)
9	IOSEL*(1)
8	IOSEL*(2)
7	IOSEL*(3)
6	N.C.
5	N.C.
4	MECHWERR*
3	N.C.
2	N.C.
1	N.C.

Table 7 - Test connector 2 (J302) pin-out

PIN NR.	J302
1	CMODE*
2	GND
3	601MODE*
4	GND
5	602MODE*
6	GND
7	NOPAR*
8	GND
9	ROMWRT*
10	GND
11	N.C.
12	GND
13	N.C.
14	GND
15	N.C.
16	GND
17	N.C.
18	GND
19	N.C.
20	GND

Table 6 - Test connector 3 (J303) pin-out

PIN NR.	J303
1	FLOW*
2	GND
3	EXTINTACK (MEC)
4	INTACK (ATAC)
5	N.C.
6	GND
7	IUHWERR*
8	GND
9	IUERR*
10	GND
11	N.C.
12	GND
13	FPUHWERR*
14	GND
15	EXTINT(2)
16	EXTINT(3)
17	EXTINT(4)
18	ATAC INT(0)
19	ATAC INT(8)
20	ATAC INT(15)

Table 7 - TAP I/F connector (J304) pin-out

PIN NR.	J304
1	TRST*
2	TDO
3	TDI
4	TMS
5	TCKF
6	TCK
7	GND
8	N.C.
9	N.C.
10	N.C.

5.4. Hardware Configuration

Specific configuration options of the IU/RT/MEC kernel in the DEM32 can be set by means of hardware straps on J302 and J303.

Option	Strap	Default Setting	Description
601MODE*	J302 : 3-4	Active	Selects 601 mode for IU, i.e. error detection etc. disabled
FLOW	J303 : 1-2	Inactive	Selects flow control mode for IU
602MODE*	J302 : 5-6	Active	Selects 602 mode for FPU, i.e. error detection etc. disabled
CMODE*	J302 : 1-2	Inactive	Selects comparison mode for IU/FPU/MEC
NOPAR*	J302 : 7-8	Active	Disables parity checking in the MEC when active
EXTINTACK-INTACK	J303 : 3-4	Active	Connects EXTINTACK output of MEC to INTACK input of ATAC.
ROMWRT*	J302 : 9-10	Inactive	Enables writing of EEPROM (boot-PROM)

Note that the default settings upon delivery of the DEM32 should not be altered, with the exception of ROMWRT* which may be set to active by installing a hardware strap. However if ROMWRT* is set, the user must take care not to corrupt the boot-PROM contents by writing to the boot-PROM area. EEPROM write must also be software enabled in the MEC in addition to ROMWRT* being active.

6. HARDWARE SOFTWARE INTERFACE

This chapter describes the DEM32 board from a software developers point of view. The detailed functionality of the main components of the DEM32 (IU-RT, FPU-RT, MEC and ATAC) is described in the corresponding specification documents, [IUSPC], [FPUSPC], [MECSPC] and [ATAC1-3].

6.1. DEM32 Memory Map

The memory map of the DEM32 is defined by the MEC. Only a small part of the possible MEC memory configuration is implemented in the DEM32. The memory map of the DEM32 is programmed on start-up as shown in Figure 5, see also paragraph 7.1.

Address (hexadecimal)	Memory contents	Size (Bytes)	Data size and parity options
0x00000000	Boot PROM	128k	8-bit mode 8 to 32 bit conversion No parity Only byte write
0x00020000	Non-implemented Boot PROM area	384k	
0x00080000	Non-implemented Extended PROM area	30,5M	
0x01F00000	Non-implemented Exchange memory	512k	
0x01F80000	MEC Registers	512k	Parity only Word write and Word/ Hword/ Byte read
0x02000000	RAM Memory	2M	Parity/EDAC options All data sizes allowed
0x02100000	Non-implemented RAM area	30M	
0x04000000	Non-implemented Extended RAM area	192M	
0x10000000	I/O area 0, ATAC	512	Parity option All data sizes allowed
0x11000200	Non-implemented I/O area 0	(16M-512)	
0x11000000	Non-implemented I/O area 1	16M	
0x12000000	Non-implemented I/O area 2	16M	
0x13000000	Non-implemented I/O area 3	16M	
0x14000000	Non-implemented Extended I/O area	1728M	
0x80000000	Non-implemented Extended general area	2G	

Figure 5 - DEM32 Memory Map

Note: The RAM area from address 0x2000000 to 0x201FFFC is used by SPARCMon (see paragraph 7.3.3.) and is not available for application/user software.

6.2. MEC Registers

A detailed description of the MEC registers can be found in [MECSPC]. Nevertheless the MEC registers are listed in this paragraph for reference. The writable registers of the MEC are only writable in the supervisor mode. All registers except the UART RX and TX are readable in every access mode. The UART RX and TX registers are only readable in supervisor mode.

6.2.1. Register Address Map

MEC Register	**Address (hex)**
MEC Control Register	01F8 0000
Software Reset Register	01F8 0004
Power Down Register	01F8 0008
Unimplemented area	01F8 000C
Memory Configuration Register	01F8 0010
I/O Configuration Register	01F8 0014
Waitstate Configuration Register	01F8 0018
Unimplemented area	01F8 001C
Access Protection Segment 1 Base Register	01F8 0020
Access Protection Segment 1 End Register	01F8 0024
Access Protection Segment 2 Base Register	01F8 0028
Access Protection Segment 2 End Register	01F8 002C
Unimplemented area	01F8 0030 - 01F8 0040
Interrupt Shape Register	01F8 0044
Interrupt Pending Register	01F8 0048
Interrupt Mask Register	01F8 004C
Interrupt Clear Register	01F8 0050
Interrupt Force Register	01F8 0054
Unimplemented area	01F8 0058 - 01F8 005C
Watchdog Program and Time-out Acknowledge Register	01F8 0060
Watchdog Trap Door Set	01F8 0064
Unimplemented area	01F8 0068 - 01F8 007C
Real Time Clock Timer <Counter>	01F8 0080
Real Time Clock Program Register <Counter>	01F8 0080
Real Time Clock <Scaler>	01F8 0084
Real Time Clock Program Register <Scaler>	01F8 0084
General Purpose Timer <Counter>	01F8 0088
General Purpose Timer Program Register <Counter>	01F8 0088
General Purpose Timer <Scaler>	01F8 008C
General Purpose Timer Program Register <Scaler>	01F8 008C
Unimplemented area	01F8 0090 - 01F8 0094
Timer Control Register	01F8 0098
Unimplemented area	01F8 009C
System Fault Status Register	01F8 00A0
Failing Address Register	01F8 00A4
Unimplemented area	01F8 00A8 - 01F8 00AC
Error and Reset Status Register	01F8 00B0
Unimplemented area	01F8 00B4 - 01F8 00CC
Test Control Register	01F8 00D0
Unimplemented area	01F800D4 - 01F800DC
UART Channel A RX and TX Register	01F8 00E0
UART Channel B RX and TX Register	01F8 00E4
UART Status Register	01F8 00E8
Unimplemented area	01F8 00EC - 01FF FFFF

6.2.2. Register Configuration and Bit Allocation

For each register the following information is given:

- **Address** of the register

- **Bits** is the bit allocation of the register. A bit can either have a unique function or a number of bits can be grouped together into a specific field.

- **Name** is the abbreviation for each bit or field.

- **Reset value** states the value for each bit or field to be obtained after the assertion of system reset.

- **Function** explains the function and use of each bit or field.

- **r/w** explains if the bit or field can be read and/or written.

Note that in some cases only part of the 32-bit register is used. The non usable bits are marked as **reserved** and will always have a fix value (generally zero) when being read and can thus not be altered by a write operation. However, all reserved bits have to be written with zeros in order to avoid parity error resulting in a MEC internal error.

Certain registers below are not true registers in the sense that they only correspond to a write operation with any data. Any data may be used when writing to these registers. Nevertheless, they are part of the register memory map and therefore included.

MEC Control Register 01F8 0000 H

Bits	Name	Reset value	Function	r/w
0	PRD	0	Power-down 1 : enabled (allowed) 0 : disabled	r/w
1	SWR	0	Software reset 1 : enabled (allowed) 0 : disabled	r/w
2	BTO	1	Bus timeout 1 : enabled 0 : disabled	r/w
3	BP	0	Block protection instead of normal access protection 1 : enabled 0 : disabled	r/w
4	WDCS	1	Watchdog clock supply 1 : external clock with prescaler (divide by 16) 0 : external clock, no prescaler	r/w
5	IUEMMSK	0	IU Error Mode Mask 1 : Error masked (= disabled) 0 : Error not masked	r/w
6	RHIUEM	0	Reset or Halt when IU error mode (ERROR*) 1 : Reset 0 : Halt	r/w
7	IUHEMSK	0	IU Hardware Error Mask 1 : Error masked (= disabled) 0 : Error not masked	r/w
8	RHIUHE	0	Reset or Halt when IU Hardware Error (HWERR*) 1 : Reset 0 : Halt	r/w
9	IUCMPMSK	0	IU Comparison Error Mask 1 : Error masked (= disabled) 0 : Error not masked	r/w
10	RHIUCMP	0	Reset or Halt when IU comparison error 1 : Reset 0 : Halt	r/w
11	FPUCMPMSK	0	FPU Comparison Error Mask 1 : Error masked (= disabled) 0 : Error not masked	r/w
12	RHFPUCMP	0	Reset or Halt when FPU comparison error 1 : Reset 0 : Halt	r/w
13	MECHEMSK	0	MEC HW Error Mask 1 : Error masked (= disabled) 0 : Error not masked	r/w
14	RHMECHE	0	Reset or Halt when MEC HW Error (MECHWERR) 1 : Reset 0 : Halt	r/w
15	reserved	0	Not used	r
16	DMAE	1	DMA 1 : enabled 0 : disabled	r/w
17	DPE	0	DMA Parity Enabled 1 : enabled 0 : disabled	r/w
18	DST	1	DMA session timeout 1 : enabled 0 : disabled	r/w
19	UBR	0	UART baud rate(1) 1 : No change of UART scaler baudrate 0 : Divide UART scaler baudrate by two	r/w
20	UPE	1	UART parity enable 1 : parity enabled 0 : no parity	r/w
21	UP	1	UART parity 1 : odd parity 0 : even parity	r/w
22	USB	0	UART stop bits 1 : two stop bits 0 : one stop bit	r/w
23	UCS	1	UART clock supply 1 : system clock 0 : external clock	r/w
31-24	Scaler	00000001	UART scaler, 1 - 255: Divide factor 0: stops the UART clock	r/w

1) The scaler shall be set to: **Scaler = Clock/(32*Baudrate*(2-UBR))-1**
Where **Clock** is either the frequency of the system clock (SYSCLK) or the watchdog clock (WDCLK) selected by UCS (bit 23 in MEC Control Register). **UBR** is the value of the UBR bit. **Baudrate** is the desired baudrate. Note that the resulting actual baudrate will probably not be exactly the desired one. This is due to the fact that **Scaler** is an integer number, and the above given equation may yield a non-integer result, depending on the **Clock** frequency.

Software Reset Register 01F8 0004 H

Write only with any data. A write to this register will cause the MEC to issue the RESET* signal if the software reset function is enabled in the MEC Control register. If the software reset function is not enabled, a write to this register will have no effect.

Power Down Register 01F8 0008 H

Write only with any data. A write to this register will cause the system to enter power down mode if the power down function is enabled in the MEC Control Register. If the power down function is not enabled, a write to this register will have no effect.

Memory Configuration Register 01F8 0010 H

Bits	Name	Reset value	Function	r/w
1-0	RBCS	00	Number of RAM blocks (chip selects used) 00 : MEMCS<0> active 01 : MEMCS<1:0> active 10 : MEMCS<3:0> active 11 : MEMCS<7:0> active	r/w
2	RBS0	0	Redundant RAM block0 selected. 0 : not selected; 1 : selected	r/w
5-3	RBR0	000	Redundant RAM block0 can replace any of the RAM blocks. The MEMCS corresponding to the replaced block will be inactive and MEMCS<8> will be active instead. 000 : MEMCS<0> inactive and replaced 001 : MEMCS<1> inactive and replaced 010 : MEMCS<2> inactive and repleced 011 : MEMCS<3> inactive and replaced 100 : MEMCS<4> inactive and replaced 101 : MEMCS<5> inactive and replaced 110 : MEMCS<6> inactive and replaced 111 : MEMCS<7> inactive and replaced	r/w
6	RBS1	0	Redundant RAM block1 selected. 0 : not selected; 1 : selected	r/w
9-7	RBR1 (1)	000	Redundant RAM block1 can replace any of the RAM blocks. The MEMCS corresponding to the replaced block will be inactive and MEMCS<9> will be active instead. 000 : MEMCS<0> inactive and replaced 001 : MEMCS<1> inactive and replaced 010 : MEMCS<2> inactive and repleced 011 : MEMCS<3> inactive and replaced 100 : MEMCS<4> inactive and replaced 101 : MEMCS<5> inactive and replaced 110 : MEMCS<6> inactive and replaced 111 : MEMCS<7> inactive and replaced	r/w
12-10	RSIZ	000	RAM size 000 : 256 kbyte 001 : 512 kbyte 010 : 1 Mbyte 011 : 2 Mbyte 100 : 4 Mbyte 101 : 8 Mbyte 110 : 16 Mbyte 111 : 32 Mbyte	r/w
13	RPA (3)	0	RAM memory parity protected 1 : enabled; 0 : disabled	r/w
14	REC	0	RAM EDAC protected 1 : enabled; 0 : disabled (Note: When EDAC is enabled parity is enabled independent of RPA)	r/w
15	Reserved	0	Not used	r
16	PWR	1	PROM write function 1 : enabled if external ROMWRT* present, 0 : disabled	r/w
17	P8 (2)	*	PROM 8-bit wide 1 : 40-bit wide, EDAC and Parity; 0 : 8-bit wide, Parity generation	r
20-18	PSIZ	000	PROM size 000 : 128 kbyte 001 : 256 kbyte 010 : 512 kbyte 011 : 1 Mbyte 100 : 2 Mbyte 101 : 4 Mbyte 110 : 8 Mbyte 111 : 16 Mbyte	r/w
23-21	Reserved	000	Not used	r
26-24	ESIZ	000	Exchange memory size 000 : 4 kbyte 001 : 8 kbyte 010 : 16 kbyte 011 : 32 kbyte 100 : 64 kbyte 101 :128 kbyte 110 :256 kbyte 111 :512 kbyte	r/w
27	EPA (3)	0	Exchange memory parity protected 1 : enabled; 0 : disabled	r/w
28	EEC	0	Exchange memory EDAC protected 1 : enabled; 0 : disabled (Note: When EDAC is enabled parity is enabled independent of EPA)	r/w
29	EEX	0	Exchange memory exists 1 : exists; 0 : exists not	r/w
31-30	Reserved	00	Not used	r

1) If RBR0 and RBR1 are set to the same value, RBR1 settings will have no effect.

2) Is at reset automatically written with same value as PROM8 input pin to MEC. A write operation to this bit has no effect (is don't care).

3) If the NOPAR* signal is asserted, data parity is not checked even if enabled in this register.

I/O Configuration Register 01F8 0014 H

Bits	Name	Reset value	Function	r/w
3-0	SIZ0	0000	I/O unit<0> size. 0000 : 512 bytes 0001 : 1 kbytes 0010 : 2 kbytes 0011 : 4 kbytes 0100 : 8 kbytes 0101 : 16 kbytes 0110 : 32 kbytes 0111 : 64 kbytes 1000 : 128 kbytes 1001 : 256 kbytes 1010 : 512 kbytes 1011 : 1 Mbytes 1100 : 2 Mbytes 1101 : 4 Mbytes 1110 : 8 Mbytes 1111 : 16 Mbytes	r/w
4	IO0	0	I/O unit<0> enabled 1 : enabled 0 : disabled	r/w
5	PA0	0	I/O unit<0> parity protected. 1 : enabled 0 : disabled	r/w
6	Reserved	0	Not used	r
7	Reserved	0	Not used	r
11-8	SIZ1	0000	I/O unit<1> size. Coded as for I/O unit<0>	r/w
12	IO1	0	I/O unit<1> enabled 1 : enabled 0 : disabled	r/w
13	PA1	0	I/O unit<1> parity protected. 1 : enabled 0 : disabled	r/w
14	Reserved	0	Not used	r
15	Reserved	0	Not used	r
19-16	SIZ2	0000	I/O unit<2> size. Coded as I/O unit<0>	r/w
20	IO2	0	I/O unit<2> enabled 1 : enabled 0 : disabled	r/w
21	PA2	0	I/O unit<2> parity protected. 1 : enabled 0 : disabled	r/w
22	Reserved	0	Not used	r
23	Reserved	0	Not used	r
27-24	SIZ3	0000	I/O unit<3> size. Coded as I/O unit<0>	r/w
28	IO3	0	I/O unit<3> enabled 1 : enabled 0 : disabled	r/w
29	PA3	0	I/O unit<3> parity protected. 1 : enabled 0 : disabled	r/w
30	Reserved	0	Not used	r
31	Reserved	0	Not used	r

Waitstate Configuration Register 01F8 0018 H

Bits	Name	Reset value	Function	r/w
1-0	RAR	11	RAM read, no. of waitstates. 00 : 0 WS 01 : 1 WS 10 : 2 WS 11 : 3 WS	w
3-2	RAW	11	RAM write, no of waitstates. 00 : 0 WS 01 : 1 WS 10 : 2 WS 11 : 3 WS	w
7-4	PRR	1111	PROM read, no. of waitstates. 0000 : 0 WS 0001 : 0 WS 0010 : 1 WS .. 1111 : 14 WS	w
11-8	PRW	1111	PROM write, no. of waitstates. 0000 : 0 WS 0001 : 1 WS .. 1111 : 15 WS	
15-12	EXRW	1111	Exchange memory read/write, no. of waitstates. 0000 : 0 WS 0001 : 1 WS .. 1111 : 15 WS	w
19-16	IO0RW	1111	IO 0 read/write, no. of waitstates. 0000 : "0 WS" See **Note 1)** below 0001 : 0 WS 0010 : 1 WS .. 1111 : 14 WS	w
23-20	IO1RW	1111	IO 1 read/write, no. of waitstates. 0000 : "0 WS" See **Note 1)** below 0001 : 0 WS 0010 : 1 WS .. 1111 : 14 WS	w
27-24	IO2RW	1111	IO 2 read/write, no. of waitstates. 0000 : "0 WS" See **Note 1)** below 0001 : 0 WS 0010 : 1 WS .. 1111 : 14 WS	w
31-28	IO3RW	1111	IO 3 read/write, no. of waitstates. 0000 : "0 WS" See **Note 1)** below 0001 : 0 WS 0010 : 1 WS .. 1111 : 14 WS	w

Note 1) In the I/O area the MEC will always insert one waitstate to wait for the BUSRDY* signal, even when "0 WS" is programmed in the above register.

Access Protection Segment 1 Base Register 01F8 0020 H
Access Protection Segment 2 Base Register 01F8 0028 H

Bits	Name	Reset value	Function	r/w
22-0	SEGBASE	0	Segment start address. The base address for the segment is calculated according to the formula BASEADDR = 0x02000000+SEGBASE*4	r/w
23	UE	0	0 = access protection disabled in user mode 1 = access protection enabled in user mode	r/w
24	SE	0	0 = access protection disabled in supervisor mode 1 = access protection enabled in supervisor mode	r/w
31-25		0	Not used	r/w

Note) The Access Protection is enable only for data write cycles.

Access Protection Segment 1 End Register 01F8 0024 H
Access Protection Segment 2 End Register 01F8 002C H

Bits	Name	Reset value	Function	r/w
22-0	SEGEND	0x3FFFFF	Segment end address denoting the first address outside the segment. The end address for the segment is calculated according to the formula ENDADDR = 0x02000000+SEGEND*4	r/w
31-23		0	Not used	r/w

Interrupt Shape Register 01F8 0044 H

Bits	Name	Reset value	Function	r/w
4-0	EDGE	00000	External interrupts edge or level triggered xxxx1 : interrupt 0 edge triggered xxxx0 : interrupt 0 level triggered xxx1x : interrupt 1 edge triggered xxx0x : interrupt 1 level triggered .. 1xxxx : interrupt 4 edge triggered 0xxxx : interrupt 4 level triggered	r/w
7-5	ACK	000	Acknowledge on external interrupt. 000 : No action 001 : interrupt 0 acknowledged 010 : interrupt 1 acknowledged 011 : interrupt 2 acknowledged 100 : interrupt 3 acknowledged 101 : interrupt 4 acknowledged 110 : No action 111 : No action	r/w
12-8	POL	00000	External interrupts polarity xxxx1 : interrupt 0 active high xxxx0 : interrupt 0 active low xxx1x : interrupt 1 active high xxx0x : interrupt 1 active low .. 1xxxx : interrupt 4 active high 0xxxx : interrupt 4 active low	r/w
31-13	Reserved	0h	Not used	r

Interrupt Pending Register 01F8 0048 H

Bits	Name	Reset value	Function	r/w
0	Reserved	0	Not used	r
15-1	IP	0h	Pending interrupts bit 1 = 1 : interrupt 1 pending bit 1 = 0 : interrupt 1 not pending .. bit 15 = 1 : interrupt 15 pending bit 15 = 0 : interrupt 15 not pending	r
31-16	Reserved	0h	Not used	r

Interrupt Mask Register 01F8 004C H

Bits	Name	Reset value	Function	r/w
0	Reserved	0	Not used	r
14-1	IM	3FFFh	Masked interrupts bit 1 = 1 : interrupt 1 masked bit 1 = 0 : interrupt 1 not masked .. bit 14 = 1 : interrupt 14 masked bit 14 = 0 : interrupt 14 not masked	r/w
31-15	Reserved	0h	Not used	r

Interrupt Clear Register 01F8 0050 H

Bits	Name	Reset value	Function	r/w
0	Reserved	0	Not used	-
15-1	IC	0h	Cleared interrupts bit 1 = 1 : interrupt 1 cleared bit 1 = 0 : interrupt 1 not cleared .. bit 15 = 1 : interrupt 15 cleared bit 15 = 0 : interrupt 15 not cleared	w
31-16	Reserved	0h	Not used	-

Interrupt Force Register 01F8 0054 H

Bits	Name	Reset value	Function	r/w
0	Reserved	0	Not used	r
15-1	IF	0h	Forced interrupts bit 1 = 1 : interrupt 1 forced bit 1 = 0 : interrupt 1 not forced .. bit 15 = 1 : interrupt 15 forced bit 15 = 0 : interrupt 15 not forced	r/w
31-16	Reserved	0h	Not used	r

Watchdog Program and Timeout Acknowledge Register 01F8 0060 H

Bits	Name	Reset value	Function	r/w
15-0	WDC	FFFFh	Preset 16-bit counter value	w
23-16	WDS	FFh	Preset 8-bit scaler value	w
31-24	WDR	FFh	Preset 8-bit reset counter value	w

The timeout for the watchdog before the watchdog interrupt occurs is calculated as:

Timeout =$(16^{WDCS\cdot}$(WDS+1)$\cdot$(WDC+1))/WDCLK

The timeout for the watchdog before warm reset occurs is calculated as:

Reset timeout= Timeout + $16^{WDCS\cdot}$(WDR+1)/WDCLK

Where WDCS is the Watchdog Clock Supply bit of the MEC Control Register and WDCLK is the frequency of the WDCLK input signal.

Watchdog Trap Door Set 01F8 0064 H

Write only with any data. A write to this register after reset but before the watchdog has elapsed will disable the watchdog. The watchdog will stay disabled until it is reprogrammed by writing to the Watchdog Program and Timeout Acknowledge Register.

Real Time Clock Timer <Counter> 01F8 0080 H

Bits	Name	Reset value	Function	r/w
31-0	RTCC	FFFFFFFFh	Downcounting 32-bit counter	r

Real Time Clock Program Register <Counter> 01F8 0080 H

Bits	Name	Reset value	Function	r/w
31-0	RTCC	FFFFFFFFh	Programmed 32-bit counter value	w

Real Time Clock <Scaler> 01F8 0084 H

Bits	Name	Reset value	Function	r/w
7-0	RTCS	FFh	Downcounting 8-bit scaler	r
31-8	Reserved	0h	Not used	-

Real Time Clock Program Register <Scaler> 01F8 0084 H

Bits	Name	Reset value	Function	r/w
7-0	RTCS	FFh	Programmed 8-bit scaler value	w
31-8	Reserved	0h	Not used	-

The timeout for the real time clock before the interrupt occurs is calculated as:

Timeout =((RTCC)·(RTCS+1))/SYSCLK

where SYSCLK is the system clock frequency.

General Purpose Timer <Counter> 01F8 0088 H

Bits	Name	Reset value	Function	r/w
31-0	GPTC	FFFFFFFFh	Downcounting 32-bit counter	r

General Purpose Timer Program Register <Counter> 01F8 0088 H

Bits	Name	Reset value	Function	r/w
31-0	GPTC	FFFFFFFFh	Programmed 32-bit counter value	w

General Purpose Timer <Scaler> 01F8 008C H

Bits	Name	Reset value	Function	r/w
15-0	GPTS	FFFFh	Downcounting 16-bit scaler	r
31-16	Reserved	0h	Not used	-

General Purpose Timer Program Register <Scaler> 01F8 008C H

Bits	Name	Reset value	Function	r/w
15-0	GPTS	FFFFh	Programmed 16-bit scaler value	w
31-16	Reserved	0h	Not used	-

The timeout for the general purpose timer before the interrupt occurs is calculated as:

Timeout =((GPTC)·(GPTS+1))/SYSCLK

where SYSCLK is the system clock frequency.

Timer Control Register 01F8 0098 H

Bits	Name	Reset value	Function	r/w
0	GCR	0	General Purpose Timer Counter Reload 1 : reload counter at zero and restart 0 : stop counter at zero	w
1	GCL	0	General Purpose Timer counter load 1 : load counter with preset value and start if enabled 0 : no function	w
2	GSE	0	General Purpose Timer enable 1 : enable counting 0 : hold scaler (and counter)	w
3	GSL	0	General Purpose Timer Scaler load 1 : load scaler with preset value and start if enabled 0 : no function	w
4	Reserved	0	Not used	r
5	Reserved	0	Not used	r
6	Reserved	0	Not used	r
7	Reserved	0	Not used	r
8	RTCCR	1	Real Time Clock Counter Reload 1 : reload counter at zero and restart 0 : stop counter at zero	w
9	RTCCL	0	Real Time Clock counter load 1 : load counter with preset value and start if enabled 0 : no function	w
10	RTCSE	0	Real Time Clock Scaler enable 1 : enable counting 0 : hold scaler (and counter)	w
11	RTCSL	0	Real Time Clock Scaler load 1 : load scaler with preset value and start if enabled 0 : no function	w
31-12	Reserved	0h	Not used	-

System Fault Status Register 01F8 00A0 H

Bits	Name	Reset value	Function	r/w
1-0	Reserved	00	Not used	r
2	SDFV	0	IU data fault valid 1 : valid 0 : not valid	r/w
6-3	SRFT	1111	Data fault type or DMA error type 0000 : Parity error on control bus 0001 : Parity error on data bus 0010 : Parity error on address bus 0011 : Access to protected area 0100 : Access to unimplemented area 0101 : MEC register parity error 0110 : MEC register access violation 0111 : Uncorrectable error in memory 1000 : Bus time-out 1001 : System bus error 1010 to 1110 : Not used 1111 : Reset value	r/w
7	Reserved	0	Not used	r
8	ASFV	0	Asynchronous fault valid 1 : valid 0 : not valid	r/w
10-9	ASFT	00	Asynchronous fault type 00 : Watch Dog time-out 01 : DMA time-out/access error 10 : UART error 11 : Correctable error in memory	r/w
11	Reserved	0	Not used	r
15-12	AT	0000	Access type see table below	r/w
31-16	Reserved	0h	Not used	r

Note 1) If this register is written the data is irrelevant. The register is set to the value 00000078 hex.
Note 2) DMA errors will not overwrite the System Fault Status Register if IU data fault valid bit is set.
Note 3) The information about ASFT can be extracted by SW from the IPR as well.

Access type

AT	Access Type	Access Mode	ASI
0000	Load	User Data RAM/ROM/Register	0xA
0001	Load	Supervisor Data RAM/ROM/Register	0xB
0010	-	Not used	-
0011	-	Not used	-
0100	Load	DMA RAM/ROM/Register	Do not care
0101	Load/execute	User I/O/Exchange	0xA, 0x8
0110	Load/execute	Supervisor I/O/Exchange	0xB, 0x9
0111	Load	DMA I/O/Exchange	Do not care
1000	Store	User Data RAM/ROM/Register	0xA
1001	Store	Supervisor Data RAM/ROM/Register	0xB
1010	-	Not used	-
1011	-	Not used	-
1100	Store	DMA RAM/ROM/Register	Do not care
1101	Store	User I/O/Exchange	0xA, 0x8
1110	Store	Supervisor I/O/Exchange	0xB, 0x9
1111	Store	DMA I/O/Exchange	Do not care

Note: Accesses in the "Extended general area" are treated as RAM Accesses from an Access Type point of view.

Failing Address Register 01F8 00A4 H

Bits	Name	Reset value	Function	r/w
31-0	FFA	0h	Failing address	r

Error and Reset Status Register 01F8 00B0 H

Bits	Name	Reset value	Function	r/w
0	IUEM	0	IU error mode 1 : error 0 : no error	r/w
1	IUHE	0	IU hardware error 1 : error 0 : no error	r/w
2	IUCMP	0	IU comparison error 1 : error 0 : no error	r/w
3	FPUHE	0	FPU hardware error (no MEC action) 1 : error 0 : no error	r/w
4	FPUCMP	0	FPU comparison error 1 : error 0 : no error	r/w
5	MECHE	0	MEC hardware error. 1 : Internal parity error 0 : no error	r/w
11 - 6		0	Not used	r/w
12	SYSAV	0	ERC32 System availability 1 : System available 0 : System not available	r/w
13	HLT	0	IU/FPU halted 1 : active 0 : not active	r
15-14	RSTC	00	Reset cause. 00 : system reset 01 : software reset 10 : error reset 11 : watchdog reset	r
31-16	Reserved	0h	Not used	r

Note: Bit 5-0 are only writable when the EWE bit in the Test Control Register is set

Test Control Register 01F8 00D0 H

Bits	Name	Reset value	Function	r/w
6-0	CB	0	Check bits	r/w
16-7		0	Not used	r
17	ET	0	EDAC test enable 0: Testing disabled 1: Memory test enabled	r/w
18	PT	0	Parity test 1 : test enabled 0 : test disabled	r/w
19	IT	0	Interrupt test 1 : test enabled 0 : test disabled	r/w
20	EWE	0	Error Write Enable 1: Write to Error and Reset Status Register enabled 0: Write to Error and Reset Status Register disabled	r/w
31-21		0	Not used	r

UART Channel A
RX and TX Register 01F8 00E0 H

Bits	Name	Reset value	Function	r/w
7-0	RTD	0h	Rx/Tx data	r/w
31-8	Reserved	0h	not used	r

UART Channel B
RX and TX Register 01F8 00E4 H

Bits	Name	Reset value	Function	r/w
7-0	RTD	0h	Rx/Tx data	r/w
31-8	Reserved	0h	not used	r

UART Status Register 01F8 00E8 H

Bits	Name	Reset value	Function	r/w
0	DRA	0	Data Ready in channel A	r
1	TSEA	1	Transmitter A Send Register Empty (no data to send)	r
2	THEA	1	Transmitter A Holding register Empty (ready to load data)	r
3	Reserved	0	Not used	r
4	FEA	0	Framing Error in receiver A	r
5	PEA	0	Parity Error in receiver A	r
6	OEA	0	Overrun Error in receiver A	r
7	CUA	0	Clear UART A (bit read as zero)	r/w
15-8	Reserved	0h	Not used	r
16	DRB	0	Data Ready in channel B	r
17	TSEB	1	Transmitter B Send Register Empty (no data to send)	r
18	THEB	1	Transmitter B Holding register Empty (ready to load data)	r
19	Reserved	0	Not used	r
20	FEB	0	Framing Error in receiver B	r
21	PEB	0	Parity Error in receiver B	r
22	OEB	0	Overrun Error in receiver B	r
23	CUB	0	Clear UART B (bit read as zero)	r/w
31-24	Reserved	0h	not used	r

6.3. Interrupt Allocation

The DEM32 interrupt allocation is shown in Table 8 below.

Table 8 - Interrupt Trap Type and default priority assignments

Trap	Priority	Trap Type	Interrupt level
Watch Dog time-out	13	0x1F	15
Ext. Interrupt 4 (not used)	14	0x1E	14
Real Time Clock	15	0x1D	13
General purpose timer	16	0x1C	12
Ext. Interrupt 3 (not used)	17	0x1B	11
Ext. Interrupt 2 (not used)	18	0x1A	10
DMA time-out	19	0x19	9
DMA access error	20	0x18	8
UART error	21	0x17	7
Correctable error in memory	22	0x16	6
UART B Data Ready or Transmitter Ready	23	0x15	5
UART A Data Ready or Transmitter Ready	24	0x14	4
Ext. Interrupt 1 - ATAC FAULT interrupt *)	25	0x13	3
Ext. Interrupt 0 - ATAC ATTN Interrupt *)	26	0x12	2
Masked Hardware errors	27	0x11	1

*) The ATAC ATTN and FAULT interrupts are active high, i.e. the MEC must be programmed to recognize high polarity for external interrupts 0 and 1. Also, these interrupts should be set to edge triggered and external interrupt acknowledge should be configured for interrupt 0 (ATTN) for normal use of the ATAC.

7. RESIDENT SOFTWARE IN START-UP PROM

Upon system reset of the DEM32, performed on power-on or by pushing the RESET button on the front panel, execution is started in the Start-Up Prom. The Start-Up PROM contains a small boot-strap routine, hardware initialisation, system self-test and SPARCmon debugger/monitor, see [MON1-2].

7.1. Boot Strap

A small boot strap routine is executed immediately after system reset. The boot strap routine initialises some MEC registers and copies the contents of the PROM to RAM, whereafter execution continues in RAM. The following parameters are changed in the MEC register reset values:

- PROM size is set to 128 kbyte.
- PROM wait state value is set to 1.
- RAM size is set to 2 MByte
- RAM wait state value is set to 0
- RAM EDAC and parity protection is enabled
- Number of nominal RAM blocks are set to 1
- I/O Area 0 is enabled, and size is set to 512 (minimum value)
- I/O Area 0 waitstate value is set to 0
- Watchdog counter is disabled

7.2. System Self-Test

After initialisation, a check sum test of the Start-Up PROM is always performed.
If the cause of reset was system reset, and a simple destructive RAM test of the entire RAM is performed including check bits and parity memory. In case the reset cause was software, error or watchdog reset (see [MECSPC]), no RAM test is performed which means that RAM contents are retained. This feature allows for post-mortem analysis of RAM contents.

If all system self tests were successful the SYSAV bit is set in the MEC, which means that the green SYSAV LED on the front panel is lit. In case any test failed, the SYSAV LED remains unlit.

7.3. SPARCmon Adaptation

7.3.1. General

SPARCmon is a PROM-based monitor program intended to reside and run on a target machine based on the SPARC microprocessor architecture. It carries out processor and system initialisation, provides trap handlers, and provides support for downloading and executing code on the target system. SPARCmon has a line-oriented, assembly-level command interface that provides basic non-symbolic debugging facilities, including step, trace, and run capabilities, instruction address breakpoints, the ability to display and modify memory and registers, disassembly, and communication facilities to permit downloading from a host. The user invokes a command by typing in the command keyword, or as many letters of the command keyword as are required to form a unique, recognisable command to the command interpreter. Some commands have optional arguments that qualify the effect of the command. These optional argument names are shown enclosed in square brackets. The SPARCmon operates with the concept of a current address. This is useful because most debugging operations take place around an address in the program. The current address is used by a number of SPARCmon commands as the default when no address is specified. It is also updated by a number of SPARCmon commands, to reflect the new current address.

The SPARCmon debugger monitor software has been adapted to the DEM32 board. Refer to [MON1-2] for a detailed description and users manual of SPARCmon.

Apart from adapting the monitor to the DEM32 memory organisation and MEC UART, a number of enhancements have been added to the general SPARCmon implementation, see paragraphs 7.3.5 and 7.3.6.

7.3.2. Serial Communication

Serial connector A (MEC UART channel A), also denoted *console* port, is used by SPARCmon for host computer communication. The following communication protocol is used: 9600 baud, no parity, one stop bit.

Serial connector B (MEC UART channel B), also denoted *auxiliary* port, is foreseen to be used by the remote debugger option of SPARCmon (not implemented in this version), or it can be used freely by the user program.

7.3.3. Memory Organisation

The SPARCmon code, residing in the Start-Up PROM copies itself to RAM after start-up. In addition to the instruction code the monitor uses some additional RAM for stack and local variables. This means that memory from address 0x2000000 to 0x201FFFF inclusive is reserved by SPARCmon and is not available to the user. Note that the reserved area is not write protected and it is the user's responsibility not to write to the area. Doing so will most likely corrupt the execution of the SPARCmon software with undefined results.

7.3.4. SPARCmon Function Commands

In this paragraph, the SPARCmon commands are briefly described. For a detailed description, see [MON2].

The commands listed in Table 9 below are available in SPARCmon:

Table 9 - SPARCmon commands

Function	Command(s)
block copy	copy
block write	fill
disassemble instructions	dis
display current address	current
display/modify baud rate	baud
examine/modify register values	reg
examine/modify memory	ex
flush bytes from serial port	flush
download a program	load
manipulate breakpoints	+bp, bp, -bp
run and trace program	cont, go, run, step, trace
reset board or monitor	reset
start program execution	go, run
miscellaneous commands	rdb, help, repeat

Some commands relinquish processor control to the target process. Control is returned to the command loop only as a result of trap processing.

baud [c|a] [-v value]

Set the baud rate for a specified port. Default and is 9600. The following values are valid: 1200, 2400, 4800, 9600 and 19200. Displaying the current baud rate is not supported.

bp

Displays all the currently active breakpoints set using the "+bp" command.

+bp

Set breakpoint at the specified address.

You can provide a series of breakpoint addresses, separated by space characters. Program execution will stop at the specified addresses. Breakpoints are implemented in SPARCmon by copying the instruction at the breakpoint address to a holding area, then overwriting it with a ITicc instruction. When the trap instruction is encountered in the instruction stream, a trap results and the trap handlers recognise the trap as a breakpoint by the trap type and the PC.

If a Ticc is encountered while the ET field of the PSR is zero, a reset trap occurs and program state may be lost. Therefore it is an error to set breakpoints or step through code where traps are disabled.

-bp [address]
Remove breakpoint at specified address, where address specifies the hexadecimal address at which to remove a breakpoint. If no address is specified, all breakpoints are removed.

cont [address]
Resume execution of a program stopped at a breakpoint at the current PC, or at the specified address, where address specifies the hexadecimal address at which to resume program execution. The default is the current PC.

copy <saddr> <daddr>[-n count] [-b|-s|-l]
Copies the specified count of bytes, shorts (16 bits) or longs (32 bits), from the source address to the destination address, where:

saddr Specifies the source address of the bytes to be copied.

daddr Specifies the destination address for the copied bytes.

-n count The number of bytes (hexadecimal value) to be copied from the source address to the destination address. The default count is 1.

-b Specifies the size as one byte (the default).

-s Specifies the size as short (16 bits), or two bytes.

-l Specifies the size as long (32 bits), or four bytes.

current [address|+offset|-offset]
Display or change current address, where:

address Specifies the hexadecimal address that is to be the new current address.

+offset Specifies that the new current address should be the old current address, plus the offset.

-offset Specifies that the new current address should be the old current address, minus the offset.

dis [address] [-n count]
Disassemble the number of instructions specified as count starting at the address specified as address, where:

address Specifies the hexadecimal address at which to begin disassembling instructions. The default is the current address.

-n count Specifies the number of instructions (hexadecimal value) to be disassembled. If count is not specified, then the default address instruction is disassembled.

ex [address] [-n count] [-b|-s|-l]

Displays the specified byte, short (16 bits) or long (32 bits), located at the specified address and allows values to be modified, where:

address Specifies the address of the byte to be displayed. The default address is the current address.

-n count Specifies the number of bytes (hexadecimal value) to be displayed, as a block of memory. There must be a space between the n and the value specified as count. The default count is 1.

-b Specifies the size as one byte (the default).

-s Specifies the size as short (16 bits), or two bytes.

-l Specifies the size as long (32 bits), or four bytes.

fill [address] [-v value] [-n count] [-b|-s|-l | -w]

Writes the specified value into each byte, short (16 bits) or long (32 bits), starting at the specified address for the number of times specified by count. The default address is the current address, the default value is 0x00, the default sizes is bytes, and the default count is one. where:

address Specifies the starting address for the fill operation. The default address is the current address.

-v value Specifies the value to write into each memory location. The default value is 0.

-n count Specifies the number (hexadecimal value) of bytes, shorts, or longs that are to have the specified value written to them. The default count is 1.

-b Specifies the size as one byte (the default).

-s Specifies the size as short (16 bits), or two bytes.

-l Specifies the size as long (32 bits), or four bytes.

-w No readback is performed after finished write access.

flush [c|a]

Flush bytes from a serial port in preparation for downloading, where:

c Selects the console serial port to be flushed.

a Selects the auxiliary serial port to be flushed.

If neither the c nor a argument is specified, operation defaults to a.

go [address]

Start program execution at the default address, or at the specified address, where:

address Specifies the hexadecimal address at which to start program execution. The default is the initial PC.

help [command]

Display the Help menu for the monitor, or the detailed help explanation for the specified command, where command specifies the name of the command for which you would like help information displayed. The Help menu is the first level of Help information available. When you invoke Help with the name of a particular command, Help displays more detailed information specific to that command.

Help Menu

```
baud      - set or display the baud rate for a port
bp        - display all breakpoints
+bp       - set one or more breakpoints
-bp       - remove one or more breakpoints
cont      - continue execution
copy      - block copy
current   - print or change current address
dis       - disassemble instructions
ex        - examine/modify memory
fill      - block fill
flush     - flush bytes from a serial port
go        - start executing
help      - display the help menu or individual command help
load      - load program into memory
rdb       - enter remote debugger mode
reg       - display/modify SPARC or board registers
repeat    - print the repeat command
reset     - cold/warm reset
run       - initialise and start program
step      - execute single instruction
trace     - display stack backtrace
```

Further help on a command, type 'help <command>'

load [c|a] [s|f|r] [address count]

Load from console or auxiliary port using the specified format, where:

c Selects the console serial port as the load source.

a Selects the auxiliary serial port as the load source. If neither the c nor a argument is specified, operation defaults to a.

s Selects the Motorola S-record format. This format does not require address or count arguments.

f Selects the monitor's fast download format. This format does not require address or count arguments.

r Selects the raw byte format. If no format specification is provided, the default is the raw byte format. This format requires at least one address and count argument pair.

address Specifies the address at which to begin storing bytes of information.

count Specifies the number of bytes (in hexadecimal) to store, starting at address.

rdb [c|a]

Start a remote debugging session over the specified serial port, where:

c Specifies the console port as the port for the remote debugger session.

a Specifies the auxiliary port as the port for the remote debugger session.

If you do not specify a port, operation defaults to the auxiliary serial port (a). This command returns control to the monitor's console only when a remote debugger process sends the appropriate message.
NOTE: remote debugging is not supported in this version of SPARCmon.

reg [board] [float] [name] [value]

Displays and/or modifies target SPARC and board register values, where:

reg Displays all SPARC integer unit (IU) registers

reg board Displays all readable registers in the MEC.

reg float Displays all the SPARC floating point unit (FPU) registers, as well as the contents of the FP queue, if any.

reg name Displays the contents of the specified register.

reg name value Store value in the register name.

There's a bug left behind by Sun here: to access register o0, type 'reg o'.

repeat

Prints the active repeat command, the command that will be echoed and executed if an "Esc" keystroke is typed.

reset [c|w]

Causes a reset, where:

c	Specifies a cold reset, which performs the same board initialisation and configuration as a power-on reset would.
w	Specifies a warm reset, which resets only the software state of the monitor. This is the default.

run [args]

Initialise and start program execution, where args specifies the arguments to be passed to the target process (in standard C argc/argv form).

The program must have been previously downloaded, and must have been linked together with the SPARCmon library programs.

step

Execute the next instruction, display current state, and then return to the command main loop.

Note: Setting a breakpoint in or step through code where traps are disabled will not work.

trace

Display a backtrace of the stack.

7.3.5. Board Specific Registers in SPARCmon for DEM32

Support for all MEC registers has been added to the monitor, i.e. all MEC registers can be referred to by name rather than memory address if desired by the user. Table 10 below lists the names used in SPARCmon for all board specific registers.

Table 10 - SPARCmon board specific registers

SPARCmon designation	Read/ write	Register name
mec_cr	rw	Control Register
mec_srr	w	Software Reset Register
mec_pdr	w	Power Down Register
mec_mcr	rw	Memory Configuration Memory
mec_iocr	rw	I/O Configuration Register
mec_wcr	w	Waitstate Configuration Register
mec_apsb1	rw	Access Protection Segment 1 Base Register
mec_apse1	rw	Access Protection Segment 1 End Register
mec_apsb2	rw	Access Protection Segment 2 Base Register
mec_apse2	rw	Access Protection Segment 2 End Register
mec_isr	rw	Interrupt Shape Register
mec_ipr	r	Interrupt Pending Register
mec_imr	rw	Interrupt Mask Register
mec_icr	w	Interrupt Clear Register
mec_ifr	rw	Interrupt Force Register
mec_wptar	w	Watchdog Program and Timeout Acknowledge Register
mec_wtds	w	Watchdog Trap Door Set
mec_rtcc	r	Real Time Clock Timer Counter
mec_rtcprc	w	Real Time Clock Program Timer Counter
mec_rtcs	r	Real Time Clock Timer Scaler
mec_rtcprs	w	Real Time Clock Program Register Scaler
mec_gptc	r	General Purpose Timer Counter
mec_gptprc	w	General Purpose Timer Program Register Counter
mec_gpts	r	General Purpose Timer Scaler
mec_gptprs	w	General Purpose Timer Program Register Scaler
mec_tcr	w	Timer Control Register
mec_sfsr	rw	System Fault Status Register
mec_far	r	Failing Address Register
mec_ersr	rw	Error and Reset Status Register
mec_tscr	rw	Test Control Register
mec_uartar	rw	UART A Rx/Tx Register
mec_uartbr	rw	UART B Rx/Tx Register
mec_uartsr	rw	UART Status Register

7.3.6. SPARCmon Enhancements for DEM32

An a addition has been made to the "fill" command in the DEM32 adaptation of SPARCmon. The "fill" command usually performs a read-back of data to the memory address which has been written. This is undesirable in certain situations, for example when accessing the ATAC or MEC write-only registers. To avoid read back the switch -w is added to the "fill" command. An example:

fill 01f800c4 -v 02020004 -l -w

The monitor will fill a MEC Register at address 1f800c4 (word format) with the value 02020004 with no read back.