MOTOROLA

External Chip Errata
DSP56654 Digital Signal Processor
Masks: 2H95G

SILICON ERRATA

Errata Number	Errata Description	Applies to Mask
ES1	Description (added 2/25/99): In the MCU core: Misaligned JMPI with pending interrupt generates an incorrect EPC result. The EPC should point at the JMPI, but instead points to the destination of the jump. (MARS1191) Workaround: Do not attempt misaligned JMPI accesses. This bug causes the misaligned access exception to be ignored if a misaligned JMPI were to occur in the code stream with an interrupt pending.	2H95G
ES2	Description (added 2/25/99): In the MCU core: When the FDB bit is set in the OnCE CTL register (this is the case when working with the Single Step debugger), if an interrupt, fast interrupt, or trace exception occurs simultaneously with an instruction access breakpoint, the interrupt, fast interrupt, or trace exception will be taken instead of the breakpoint which would put the processor in debug mode. (MARS 1212) Workaround: This errata should not cause a problem since the breakpoint will be taken after returning from the interrupt or exception processing.	2H95G
ES3	Description (added 3/2/99) In the MCU core: If a load byte (LD.B) or load half-word (LD.H) is followed by a JMP with a data dependency, it will not zero extend the jump address correctly. (MARS 1215) Workaround: Put another instruction between the LD.(B,H) and the JMP, or don't allow the compiler to output LD.(B,H) to a JMP. Current DIAB compiler does not generate this case.	2H95G

Errata Number	Errata Description	Applies to Mask
ES4	Description (added 3/9/99): In the MCU core: If debug mode is entered near the end of a multiply or divide instruction and there is still an outstanding instruction fetch, the PC may be inadvertently incremented. (MARS 1216) Workaround: Do not allow the assertion of debug request (DBGRQ) or breakpoints on the instruction following a multiply or divide instruction.	2H95G
ES5	Description (added 7/7/99) In the MCU core: In wait-stated memory, if a change-of-flow instruction (JMP/BR) follows a multiply or divide instruction, a debug request may cause the PC to point to destination-2 of the change-of-flow instruction upon exit from debug mode. (MARS 1226) Workaround: Insert a SYNC instruction between the multiply or divide instruction and the change-of-flow instruction.	2H95G
ES6	Description (added 2/25/99): In the MCU core: When the processor status register's (PSR's) exception enable (EE) bit and fast interrupt enable (FE) bit are both set and an execute exception occurs (refer to sections 2-4 of Table 4-3 in the M*CORE Reference Manual) in conjunction with both a breakpoint on an instruction access and a fast interrupt, the processor status register's (PSR's) fast interrupt enable (FE) bit may not be cleared when the fast interrupt is taken. (MARS 1869) Workaround: None	2H95G
ES7	Description (added 3/5/99): In the MCU core: If the MCU enters STOP mode with the CKIH clock selected (MCS=1 in CKCTL), an attempt to switch the MCU_CLK to CKIL after exiting the STOP mode may cause the MCU_CLK to stop. In this case, the only recovery mechanism is to reset the device. Workaround: Switch the MCU_CLK to CKIL prior to entering STOP mode.	2H95G

Errata Number	Errata Description	Applies to Mask
ES8	Description (added 7/7/99) In the MCU core: On a data breakpoint followed by a multicycle instruction and the instruction buffer is full (which will occur when the user is running out of zero wait-state instruction memory), the breakpoint will not be taken. In this case, the PC will be corrupted. (MARS 1351) Workaround: Don't allow data side breakpoints around multicycle instructions when executing out of wait-stated memories.	2H95G
ES9	Description (added 7/7/99) In the MCU core: If there is a data breakpoint on the table access of a JSRI or JMPI instruction, and an access error ($\overline{\text{TEA}}$) occurs on the destination fetch, an access error exception will be taken instead of the breakpoint being recognized. (MARS 1701) Workaround: If possible, fix the software so that JMPI/JSRI instructions do not cause an access error.Otherwise, don't allow data breakpoints on JSRI/JMPI table accesses.	2H95G
ES10	Description (added 7/7/99) In the MCU core: The combination of a MTCR instruction followed by a low-power instruction is not treated as an atomic operation if they are executed out of wait-stated memory. In this case, if the MTCR instruction enables interrupts, an interrupt may be taken before the low power instruction can be executed. (MARS 1839) Workaround: Run the MTCR/low-power mode sequence in zero wait-state memory to make this an atomic operation.	2H95G
ES11	Description (added 7/7/99) In the MCU core: Three OnCE™ registers are not writable during reset; OTC, MBCA, MBCB-trace, breakpoint a and b counter. Workaround: Enter debug mode out of reset and then write these registers. (Single Step debugger takes care of this.)	2H95G

Errata Number	Errata Description	Applies to Mask
ES12	Description (added 7/7/99) In the MCU core: In the OnCE™ Status register (OSR), the SWO bit, which should be set only as a result of BKPT instruction, may be inadvertently set when entering debug mode. This bit may also not be cleared as it should be when exiting debug mode. (MARS 1352) Workaround: Do not rely solely on the SWO bit to determine what caused entry to debug mode. (The Single Step debugger takes care of this.)	2H95G
ES13	Description (added 7/7/99) In the MCU core: For cases when the FDB bit is set in the OnCE CTL register (This is the case when working with the Single Step debugger), if there is a data breakpoint on the first fetch of a load multiple (LDM/LDQ) and the second fetch of the load multiple has an access error ($\overline{\text{TEA}}$) an access error exception will be taken instead of entering debug mode. (MARS 1516) Workaround: If possible, fix the software so that load multiple accesses do not cause access errors. Otherwise, don't set data breakpoints on load multiple data accesses.	2H95G
ES14	Description (added 7/7/99) In the MCU core: For cases when the FDB bit is set in the OnCE CTL register (this is the case when working with the Single Step debugger), if a data breakpoint occurs in conjunction with an access error ($\overline{\text{TEA}}$) on a load/store type (LDQ/STQ/LDM/STM/LD/ST) instruction to/from wait-stated memory, the PC will not be incremented correctly. When debug mode is exited, the PC will incorrectly point to the instruction prior to the load/store instruction. (MARS 1527) Workaround: If possible, fix the software so that load/store type instructions do not cause access errors. Otherwise, avoid setting data breakpoints on wait-stated memories.	2H95G

Errata Number	Errata Description	Applies to Mask
ES15	Description (added 7/7/99) In the MCU core: In instruction trace mode (PSR TM bits set to 01), if debug mode is entered immediately after an RTE or RFI instruction, the RTE or RFI will be traced after exiting debug mode. (MARS 1593) Workaround: Do not use instruction trace mode when debug mode may be entered.	2H95G
ES16	Description (added 7/7/99) In the MCU core: In a OnCE single-step of a LD.W instruction, the OnCE write-back bus register (WBBR) may get corrupted. (MARS 1730) Workaround: Follow OnCE single-step of a LD.W to the Rx register with “MOV Rx, Rx” to correct the value in WBBR. (The Single Step debugger takes care of this.)	2H95G
ES17	Description (added 7/7/99) In the MCU core: Sequential breakpoints with the Sequential control Field SQC[1:0] in the OnCE Control register (OCR) set to 10 or 11, do not function as specified. In addition, the sequential breakpoint SQB and SQA bits into the OnCE status register (OSR) are not updating properly in all sequential breakpoint modes. (MARS 1852, 1855) Workaround: None.	2H95G

Errata Number	Errata Description	Applies to Mask
ES18	Description (added 7/7/99) In the MCU core: For cases when the FDB bit is set in the OnCE CTL register (this is the case when working with the Single Step debugger), if an access error ($\overline{\text{TEA}}$) occurs in conjunction with an interrupt request and a data breakpoint, the breakpoint may not be recognized. This condition is sensitized when an access error ($\overline{\text{TEA}}$) occurs during the abort cycle after exception recognition. In this case, the core takes the access error exception instead of the data breakpoint. The EPC points to the data transaction. (MARS 1150) Workaround: Allocate system memory such that when the data transaction is rerun upon returning from the access error exception, it will not cause an access error ($\overline{\text{TEA}}$).	2H95G
ES19	Description (added 7/7/99) In the MCU core: When the processor status registers (PSR's) fast interrupt enable (FE) bit is set and the exception enable (EE) bit is cleared, if an execute exception occurs (refer to sections 2-4 of Table 4-3 in the M*CORE reference manual) in conjunction with a fast interrupt, an unrecoverable exception will be taken, which is the correct behavior, but will clear the PSR's FE bit. (MARS 1870) One exception to this is, on a data breakpoint on a load/store multiple (LDM/STM/LDQ/STQ) with the PSR's interrupt control (IC) bit set, a fast interrupt asserted with the PSR's FE bit set and the PSR's EE bit cleared, will cause the breakpoint to be recognized instead of an unrecoverable. (MARS 1621) Workaround: To limit exposure to this situation, save the EPC/EPSR and set the PSR's EE bit as soon as possible after EE is cleared.	2H95G
ES20	Description (added 7/7/99) In the MCU core: Processor status pins (PSTAT) are for debug purposes only and may not truly reflect the state of the machine. (MARS 1312, 1313, 1318, 1323) Workaround: None.	2H95G

Errata Number	Errata Description	Applies to Mask
ES21	Description (added 7/7/99) In the MCU core: If the processor status register's (PSR's) supervisor/user (SU) bit is cleared, which means the processor is in user mode, and the processor is running out of 2 or more wait-stated memory, and there is an exception, the first instruction fetch of the handler will occur while still in user mode. This will cause an unrecoverable error if the handler resides in memory which does not allow accesses while in user mode. (MARS 1895) Workaround: In this situation, place the exception handler at a location in memory in which user mode accesses are allowed.	2H95G
ES22	Description (added 7/7/99) In the MCU core: For cases when the FDB bit is set in the OnCE CTL register (this is the case when working with the Single Step debugger), on a data breakpoint with the buffer full, (which will occur with zero wait-state instruction memory), a subsequent load or store instruction will result in inadvertent data fetches before the processor enters debug mode. The address of the inadvertent data fetches will be equal to the data from the previous load. (MARS 1881) Workaround: These extraneous data fetches should not cause a problem to the user. If that is not the case, avoid data breakpoints on instructions followed by multicycle instructions.	2H95G
ES23	Description (added 7/19/99) In the MCU core: If the processor is running out of wait-stated memory and the trace mode (TM) bit is set in the processor status register (PSR) which means trace mode is enabled, unnecessary trace exceptions will occur around RTE, RFI, MTCR, and MFCR instructions which could potentially lock up the machine. (MARS 1917) Workaround: If the user is trying to debug code with the trace mode bit set, it needs to be run out of zero-wait state memory.	2H95G

Errata Number	Errata Description	Applies to Mask
ES24	Description (added 3/9/99): In the SIM peripheral: While in initial character mode, if the first received character has a parity error, the parity error will continue to be reported for subsequent characters. This is only in effect while in initial character mode. The parity error logic behaves correctly in the normal mode. Workaround: Reset the port by setting the SISR bit in SIMCR for 28us for CKIH of 16.8 MHz.	2H95G
ES25	Description (added 3/9/99): In the UARTA and UARTB peripheral: Byte write access to UCR1 does not execute properly. UCR1[3] (time-out mode) was connected to the upper byte enable write strobe instead of the lower one and the following occurs: • When attempting a byte write access to the LSB, bit number 3 won't be written. • When attempting a byte write access to the MSB, bit number 3 will be written although it wasn't supposed to be written. Workaround: Only write this register with halfword access.	2H95G

Errata Number	Errata Description	Applies to Mask

ES26	Description (added 2/24/99):
	
	In the DSP: When Stack Extension mode is enabled, use of the instructions BRKcc or ENDDO inside DO loops may cause improper operation. If the loop is not nested and does not have a nested loop inside of it, the errata is relevant only if LA or LC values are being used outside the loop.
	
	Workaround:
	
	If Stack Extension is used, emulate the BRKcc or ENDDO. This is applicable to both finite DO loops and DO forever loops. The following example is for Finite DO loops:
	
	BRKcc
	--- 	
	Original code:
	`do #N,label1`
	`.....`
	`.....`
	`do #M,label2`
	`.....`
	`.....`
	`BRKcc`
	`.....`
	`.....`
	`label2`
	`.....`
	`.....`
	`label1`
	
	Should be replaced by:
	
	`do #N, label1`
	`.....`
	`.....`
	`do #M, label2`
	`.....`
	`.....`
	`Jcc fix_brk_routine`
	`.....`
	`.....`
	`nop_before_label2`
	`nop ; This instruction must be NOP.`
	`label2`
	`.....`
	`.....`
	`label1`
	`....`
	`....`
	
	`fix_brk_routine`
	`move #1,lc`
	`jmp nop_before_label2`
	
	`ENDDO`
	`------`
		2H95G

Errata Number	Errata Description	Applies to Mask

	Original code:
	
	`do #M,label1`
	`.....`
	`.....`
	`do #N,label2`
	`.....`
	`.....`
	`ENDDO`
	`.....`
	`.....`
	`label2`
	`.....`
	`.....`
	`label1`
	
	Should be replaced by:
	
	`do #M, label1`
	`.....`
	`.....`
	`do #N, label2`
	`.....`
	`.....`
	`JMP fix_enddo_routine`
	`nop_after_jmp`
	`NOP ; This instruction must be NOP.`
	`.....`
	`.....`
	`label2`
	`.....`
	`.....`
	`label1`
	`....`
	`....`
	
	`fix_enddo_routine`
	`move #1,lc`
	`move #nop_after_jmp,la`
	`jmp nop_after_jmp`	

Errata Number	Errata Description	Applies to Mask

The following examples of code are for DO forever loops:

BRKcc

Original code:

```
            do #M,label1
            .....
            .....
                    do forever,label2
                    .....
                    .....
                    BRKcc
                    .....
                    .....
    label2
            .....
            .....
    label1
```

Should be replaced by:

```
            do #M,label1
            .....
            .....
                    do forever,label2
                    .....
                    .....
                  JScc    fix_brk_forever_routine  ; <--- note: JScc and not
    Jcc
                    .....
                    .....
    nop_before_label2
                    nop     ; This instruction must be NOP.
    label2
            .....
            .....
    label1
    ....
    ....


    fix_brk_forever_routine
            move ssh,x:<..>  ; <..> is some reserved not used address (for
    temporary data)
            move #nop_before_label2,ssh
            bclr #16,ssl     ;
            move #1,lc
            rti              ; <---- note: "rti" and not "rts" !


    ENDDO
    ------
```

Errata Number	Errata Description	Applies to Mask

	Original code:
	`do #M,label1`
	`.....`
	`.....`
	`do forever,label2`
	`.....`
	`.....`
	`ENDDO`
	`.....`
	`.....`
	`label2`
	`.....`
	`.....`
	`label1`
	Should be replaced by:
	`do #M,label1`
	`.....`
	`.....`
	`do forever,label2`
	`.....`
	`.....`
	`JSR fix_enddo_routine ; <--- note: JSR and not JMP`
	`nop_after_jmp`
	`NOP ; This instruction must be NOP.`
	`.....`
	`.....`
	`label2`
	`.....`
	`.....`
	`label1`
	`....`
	`....`
	`fix_enddo_routine`
	`nop`
	`move #1,lc`
	`bclr #16,ssl`
	`move #nop_after_jmp,la`
	`rti ; <--- note: "rti" and not "rts"!`	

ES27	Description (added 7/19/99) In the SIM peripheral: The SIMDATA line does not meet the ISO7816 specification regarding the logic low voltage and current. The SIMDATA line has a totem pole (full CMOS) driver rather than an open-drain driver. Workaround: None	2H95G

Errata Number	Errata Description	Applies to Mask
ES28	Description (added 7/19/99) In the SIM peripheral: The peripheral may not function properly if the SIM card transmits characters using a guard time of only 2 ETUs. Workaround: None	2H95G
ES29	Description (added 9/29/99) In the MCU core: If the MCU has entered STOP mode while CKIH clock is disabled (CKIHD bit is set in the CKCTL register) then the MCU can enter a deadlock condition upon leaving the STOP mode. This condition occurs (after exiting STOP) when CKIH is re-enabled and then selected as the MCU clock source (by clearing CKIHD and setting MCS bits, respectively). The only recovery mechanism for the deadlock is to reset the device. Workaround: To prevent the deadlock condition, an interrupt to the MCU core should be generated after the CKIH clock is re-enabled (CKIHD bit is cleared). The recommended way to cause this interrupt is to use one of the software interrupts in the MCU interrupt controller. After the interrupt, the MCU clock can be switched correctly to the CKIH clock. For example, entering and exiting stop mode with CKIH disabled: clear MCS bit to source CKIL to MCU set CKIHD bit to disable CKIH buffer enter MCU STOP . [device is in low power stop mode] . exit MCU STOP clear CKIHD bit to re-enable CKIH buffer trigger software interrupt (set ES0 in NIER register) wait for interrupt service to complete (the interrupt service routine should clear ES0) set MCS bit to source CKIH to MCU continue....	2H95G

Motorola and Ⓜ are registered trademarks of Motorola, Inc.
OnCE is a trademark of Motorola, Inc.

NOTES

1. An over-bar (i.e., $\overline{xxxx}$) indicates an active-low signal.
2. The letters seen to the right of the errata tell which mask numbers apply.
3. The Motorola DSP website has additional documentation updates that can be accessed at the following URL:

   http://www.mot.com/SPS/DSP/documentation

-end-