Order this document by
MCF5206UMAD/AD

MCF5206

Chip Errata to
MCF5206
Integrated Microprocessor

February 10, 2000

This document and other information on this product is maintained on the World Wide Web at http://www.mot.com/coldfire.

Errata Mask Set: 0G10J and 0J43S

Instruction Processing

IP1- The MOVEC opcode to load the Vector Base Register does not function correctly. Address register 7 must be used as the source operand. A software workaround is shown below:

```
VBASE    EQU <vbr_value>  # VBASE holds the desired value for VBR

MOVE.W   #$2700,SR        # disable all but level 7 interrupts
MOVE.L   A7,<mem>         # save current stack pointer
MOVE.L   VBASE,A7         # load desired value of VBR into A7
MOVEC.L  A7,VBR           # load the Vector Base Register.  Most compilers
                          # recognize the VBR name syntax as the vector
                          # base register.
MOVEA.L  <mem>,A7         # restore the stack pointer
MOVE.W   #$2000,SR        # enable interrupts
```

IP2- Loading of the Module Base Register (MBAR) fails unless register D0 is used as the source operand. The workaround is straightforward:

```
MOVE.L   MBASE+VALID,D0  # mbar contents + set valid bit
MOVEC    D0,MBAR         # load mbar
```

Instruction Cache

I1- If a stream of instruction fetches does not hit in theK-Bus RAM (KRAM) and switches from cacheable mode to noncacheable mode, an incorrect value can be returned for an instruction fetch.

This situation can occur by enabling the instruction cache and using one or more access control registers (ACRs) and/or the default cache mode control in the CACR (cache control register) to create two instruction address ranges—one cacheable and one noncacheable—that both map to instruction cache space (i.e., not to the KRAM).

NOTE: This situation cannot occur by just enabling the instruction cache after reset. That is, if the instruction fetch stream changes from noncacheable to cacheable but never back to noncacheable, there is no problem.

Workarounds include:

- Do not use the ACRs for instruction address space.

or

- Have a consistent cache mode for the default cache mode and any enabled ACR (either all cacheable
  or all noncacheable) used for instruction space.

I2- Incorrect code fetches can occur if both of the following circumstances apply:

- cacheable code resides in the uppermost 512 bytes of the 4 GByte address space, i.e., addresses $FFFFFE00 - $FFFFFFFF,
- The external memory access profile allows for 0 or 1 wait states on the initial transfer of a line fetch, i.e., wait states <= 1-x-y-z, where x,y,z can be any wait state value.

NOTE: The uppermost 512 bytes of address space can be used for KRAM-resident instructions without the possibility of erroneous code fetches. Under any circumstances, there is no possibility for data corruption.

Workarounds include:

- Do not allow execution of cacheable instructions at addresses mapped to the top 512 bytes of the 4 GByte address space. This memory region can be used for KRAM-resident instructions,

or

- If instruction execution at the top of the address space is required, then mark that region as non-cacheable, or mark it as cacheable but require the External memory response to be at least 2-x-y-z.

MBUS Module

M1- The SCL and SDA input buffers of the MBUS module do not currently use Schmitt trigger input buffers, which would allow for slow input rise and fall time characteristics. The 1ms rise and fall times listed in the MBUS input specification section (specifications M3 and M5) of the *MCF5206 User's Manual* are too slow to guarantee proper operation with non-Schmitt trigger input buffers.

Workaround:

External hardware driving the SCL and SDA inputs must tighten up the input rise and fall times (specifications M3 and M5) to be no more than1 CLK period in length to ensure proper operation of the MBUS interface.

Background Debug Mode

BDM1 - Misaligned WDDATA Operands are Not Captured

When executing the "WDDATA" instruction, certain K-Bus sequences may cause a misaligned operand associated with a WDDATA instruction to not be captured. This restriction applies to word or long-word operands which are misaligned.

Workaround:

Guarantee that all operands referenced by WDDATA instructions are aligned, i.e., 16-bit references on 0-modulo-2 addresses and 32-bit references on 0-modulo-4 addresses.

BDM2- Possible Unexpected Data Breakpoint Trigger

If a data breakpoint monitoring multiple word and byte values is configured, there is a possibility that the breakpoint may falsely trigger. As an example, if the breakpoint hardware is configured to monitor for a certain byte value within a range of addresses, a byte reference is made and the data breakpoint value is present on the data bus, but in a different byte lane, a false trigger could be

generated. The logic to fully qualify the reference size and the appropriate bytes of the data bus is incorrect.

Note, this bug does NOT affect the ability to configure a breakpoint of a certain data value at a certain address. The false trigger can only occur when multiple bits in the following fields of the Trigger Definition Register are asserted:

- if multiple bits of TDR[27:26] are set, and a level 2 trigger is enabled, or
- if multiple bits of TDR[25:22] are set, and a level 2 trigger is enabled, or
- if multiple bits of TDR[11:10] are set, and a level 1 trigger is enabled, or
- if multiple bits of TDR[9: 6] are set, and a level 1 trigger is enabled,

then a false trigger may occur

Workaround:

Restrict the use of the data breakpoint to only a single operand, and enable only a single bit in the TDR appropriate for the operand size andlocation on the data bus.

Timing

T1- For proper operation at 33 MHz, the input clock duty cycle (spec C4) must be between a minimum of 45% and a maximum of 55 %. Accordingly, for 33 MHZ operation, the minimum and maximum clock widths (Specs C4a and C4b) must be 13.5 nsecs and 16.5 nsecs, respectively.

Permanent Specification Changes

SBDM1-When a BDM command is serially shifted into a ColdFire™ microprocessor, the debug module requests the use of the KBus to perform the required operation. Under certain conditions, the processor may never grant the KBus to the debug module causing the BDM command to never be executed

Specifically, the KBus grant may be withheld from the debug module if the processor is executing a tight loop where the entire loop is contained within one aligned longword.

Examples include:

```
          align 4
label1:   nop
          bra.b label1
          align 4
label2:   bra.w label2
```

Workaround:
The workaround is to force the loop to be aligned across two longwords. Given this alignment, the processor will correctly grant the KBUS to the debug module.

MCF5206 Errata Change History:

• Rev 1.0: Initial Errata with I1,M1, SBDM1 1/27/97

• Rev 1.1: Added IP1, IP2. 7/26/97

• Rev 1.2: Added BDM1, BDM2. 10/1/97

• Rev 1.3: Added I2, T1. 11/11/97

• Rev 1.4: Rewrote program code in IP1, IP2. 12/03/97

• Rev 1.5: Added mask set 0J43S

Literature Distribution Centers:
USA/EUROPE: Motorola Literature Distribution; P.O. Box 20912, Arizona 85036.
JAPAN: Nippon Motorola Ltd.; 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141 Japan.
ASIA-PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Center, No. 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong.