

# MC9S12DP512, Mask 4L00M

---

## Introduction

This errata sheet applies to the following devices:

- MC9S12DP512, MC9S12DT512, MC9S12DJ512, MC9S12A512

---

## MCU Device Mask Set Identification

The mask set is identified by a 5-character code consisting of a version number, a letter, two numerical digits, and a letter, for example 1K79X. All standard devices are marked with a mask set number and a date code.

---

## MCU Device Date Codes

Device markings indicate the week of manufacture and the mask set used. The date is coded as four numerical digits where the first two digits indicate the year and the last two digits indicate the work week. For instance, the date code "0201" indicates the first week of the year 2002.

---

## MCU Device Part Number Prefixes

Some MCU samples and devices are marked with an SC, PC, or XC prefix. An SC prefix denotes special/custom device. A PC prefix indicates a prototype device which has undergone basic testing only. An XC prefix denotes that the device is tested but is not fully characterized or qualified over the full range of normal manufacturing process variations. After full characterization and qualification, devices will be marked with the MC or SC prefix.

---

## Errata System Tracking Numbers

MUCtsXXXXX is the tracking number for device errata. It can be used with the mask set and date code to identify a specific erratum.

## Errata Summary

Errata Number	Module affected	Brief Description	Work-around
<a href="#">MUCts00735</a>	atd_10b8c	Flags in ATDSTAT0 do not clear by writing '1', ETORF erroneously set	YES
<a href="#">MUCts00784</a>	atd_10b8c	write to ATDCTL5 may not clear SCF, CCF and ASCIF flags	YES
<a href="#">MUCts00807</a>	fts512k4	Additional write protection exists via mirroring.	NO
<a href="#">MUCts00821</a>	crg	PLL: If osc_clock is 2 to 3 times pll_clock, STOP can cause SCM or reset	YES
<a href="#">MUCts00865</a>	fts512k4	ACCERR is not set for a Byte Access	YES
<a href="#">MUCts00904</a>	fts512k4	STOP instruction may set flash ACCERR flag.	YES
<a href="#">MUCts00991</a>	eets4k	STOP instruction may set EEPROM ACCERR flag.	YES
<a href="#">MUCts01011</a>	ect_16b8c	ECT: Input pulse shorter than delay counter period recognised as a valid	YES
<a href="#">MUCts01029</a>	atd_10b8c	CCF flags in ATDSTAT1 register might fail to set	NO
<a href="#">MUCts01039</a>	atd_10b8c	ATD: Clearing of CCF flags in ATDSTAT1 by write of ATDCTL5 might not work	YES
<a href="#">MUCts01104</a>	mscan	MSCAN: Time stamp corrupted in receive buffer	YES
<a href="#">MUCts01489</a>	mcu_9dp512	Reduced FLASH programming temperature range and increased programming time for all product marked before date code 0514 (2005 WW14).	NO
<a href="#">MUCts01966</a>	S12_bdm	Possible manipulation of return address when exiting BDM active mode	YES
<a href="#">MUCts02382</a>	eets4k	EEPROM Program Failure during Sector-Modify	YES
<a href="#">MUCts02415</a>	S12_mebi	MEBI: Missing ECLK edge on first external access after mode switching	YES
<a href="#">MUCts03403</a>	spi	SPI: Disabling slave SPI together with clearing CPHA while SS low locks transmit shift register for the next transmission	YES

## Flags in ATDSTAT0 do not clear by writing '1', ETORF erroneously set

MUCts00735

### Description

For the flags SCF, ETORF and FIFOR in ATDSTAT0 it is specified that writing a '1' to the respective flag clears it. This does not work. Writing '1' to the respective flag has no effect.

The ETORF flag is also set by a non-active edge, e.g. falling edge trigger (ETRILE=0, ETRIGP=0). ETORF is set on both falling edges and rising edges while conversion is in progress.

### Workaround

#### SCF

1. Use the alternative flag clearing mechanisms:
  - a. Write to ATDCTL5 (a new conversion sequence is started)
  - b. If AFFC=1 a result register is read

#### ETORF

1. Use the alternative flag clearing mechanisms:
  - a. Write to ATDCTL2, ATDCTL3 or ATDCTL4 (a conversion sequence is aborted)
  - b. Write to ATDCTL5 (a new conversion sequence is started)
2. Avoid external trigger edges during conversion process by using short pulses
3. Ignore ETROF flag

#### FIFOR

1. Use the alternative flag clearing mechanism:
  - a. Start a new conversion sequence  
(write to ATDCTL5 or external trigger)

---

## write to ATDCTL5 may not clear SCF, CCF and ASCIF flags

MUCts00784

### Description

If a write to ATDCTL5 happens at exactly the bus cycle when an ongoing conversion sequence ends, the SCF, CCF and (if ASCIE=1) ASCIF flags remain set and are NOT cleared by a write to ATDCTL5

## Workaround

1. Make sure the device is protected from interrupts (temporarily disable interrupts with the I mask bit).
2. Write to ATDCTL5 twice.

---

## Additional write protection exists via mirroring.

**MUCts00807**

## Description

Flash protection is mirrored once (hence appears twice) in every flash block. The mirrored protection will occur four pages lower than the intended protection. For example, if flash page \$3F is protected, the mirrored protection, specified by the FPROT register setting for the respective block, will occur in flash page \$3B. If flash page \$3E is protected, the mirrored protection will occur in page \$3A. This issue exists in the same format in all flash blocks.

## Workaround

None.

---

## PLL: If osc\_clock is 2 to 3 times pll\_clock, STOP can cause SCM or reset

**MUCts00821**

## Description

This Erratum applies only to systems where PLL is used to divide down the osc\_clock by a ratio between 2 and 3.

If

- 1) pll\_clock (PLLON=1) is running  
and
- 2)  $2 < \text{osc\_clock}/\text{pll\_clock} < 3$   
and
- 3) full stop mode is entered (STOP instruction with PSTP Bit =0)

there is a small possibility that when entering full stop mode the chip reacts as follows:

- 1) if self clock mode is disabled (SCME=0) monitor reset is asserted.  
The system does NOT enter stop mode.  
or
- 2) if self clode mode and SCM interrupt are enabled (SCME=1 and SCMIE=1) a self clock mode interrupt is generated. The SCMIF flag is set.  
The system does NOT enter stop mode.  
or
- 3) if SCME=1 and SCMIE=0 the system will enter full stop mode.  
But after wakeup self clock mode is entered without doing the specified clock quality check. The SCMIF flag is set.

## Workaround

- 1) Avoid osc\_clock/pll\_clock ratios between 2 and 3.  
or
- 2) if you really require osc\_clock/pll\_clock ratio between 2 and 3 do the following before going into stop.
  - a) deselect PLL (PLLSEL=0)
  - b) turn off PLL (PLLON=0)
  - c) enter stop
  - d) exiting stop: turn on PLL again (PLLON=1)

---

## ACCERR is not set for a Byte Access    MUCts00865

### Description

Starting a command sequence with a MOVB array write instruction (Byte Write) will not generate an access error. The command is processed normally programming the array according to the content (word) of the data register while only the high byte in the FDATA register holds valid information.

## Workaround

Avoid the use of MOVB instruction for array program operations.

---

## STOP instruction may set flash ACCERR flag. MUCts00904

### Description

If the FCLKDIV flash clock divider register has been loaded, and the flash is not executing a command (flash CCIF command complete flag is set), the execution of a STOP instruction will erroneously set the ACCERR access error bit in the FSTAT flash status register.

## Workaround

The ACCERR bit in the FSTAT register must be cleared after the execution of a STOP instruction if the FCLKDIV register has been loaded.

---

## STOP instruction may set EEPROM ACCERR flag. MUCts00991

### Description

If the ECLKDIV EEPROM clock divider register has been loaded, and the EEPROM is not executing a command (EEPROM CCIF command complete flag is set), the execution of a STOP instruction will erroneously set the ACCERR access error bit in the ESTAT EEPROM status register.

## Workaround

The ACCERR bit in the ESTAT register must be cleared after the execution of a STOP instruction if the ECLKDIV register has been loaded.

---

## ECT: Input pulse shorter than delay counter period recognised as a valid

MUCts01011

### Description

According to the observation, input pulse (high/low) whose pulse width is shorter than delay counter window is mistakenly recognized as a valid pulse. Hence the ic flags will be set and may result in an IRQ if IRQ is enabled.

## Workaround

A software workaround is available.

User software should check the logic level of the input capture pin within the interrupt service routine and compare this with the logic level when the input is not asserted. This can be performed using the appropriate registers in the port integration module.

If the pin reads the logic level of the inactive state, the pulse is shorter than the time defined in the delay counter control register plus the interrupt latency. In this case, the pulse triggering the input capture is not valid (too short), hence the interrupt can be acknowledged and exited without further action taking place. If the pin reads the logic level of the active state, the input pulse is valid

and

the interrupt should be acknowledged and the correct input capture service routine executed.

The effectiveness of this workaround must be evaluated by identifying the worst case latency involved in the call of the ISR. To maximise the effectiveness of pulse rejection, users must consider checking the value in the capture register against the free-running timer on every new capture.

---

## **CCF flags in ATDSTAT1 register might fail to set**

**MUCts01029**

### **Description**

The setting of the CCF7-0 flags in ATDSTAT1 register is not independent of the clearing.

A clear on CCFx (e.g. Bit AFFC=1 and read of ATDDRx)

which occurs in exactly the same bus cycle as the setting of any other flag CCFy (x,y = 0,1,...,7; x!=y) masks the setting of CCFy.

CCFy will not set in this special case although the corresponding conversion has completed and the result (ATDDRy) is valid.

### **Workaround**

None.

---

## **ATD: Clearing of CCF flags in ATDSTAT1 by write of ATDCTL5 might not work**

**MUCts01039**

### **Description**



Starting a new conversion by writing to the ATDCTL5 register should clear all CCF flags in the ATDSTAT1 register.  
This does not always work if the write to ATDCTL5 register occurs near the end of an ongoing conversion.  
Although all CCF flags are cleared one CCF flag might be set again within the 1st ATD clock period of the new conversion.

## Workaround

If the unexpected setting of one CCF flag can not be accepted by the application one of the following workarounds can be taken:

- 1) Abort conversion (e.g. by write to ATDCTL3)  
    Pause for 2 ATD clock periods  
    Start new conversion
- 2) Ignore first conversion sequence and clear CCF flags

---

## MSCAN: Time stamp corrupted in receive buffer

MUCts01104

### Description

When the foreground receive buffer (RxFG) is read, with the Receiver Full Flag (RXF) set, the value of the Time Stamp Register may be incorrect due to corruption. The Time Stamp Register is written correctly when the message is received, but may be overwritten by the timer value at the end of a subsequent reception. The corruption can only occur close to a data overrun, when the receive buffer FIFO is full.

The problem occurs whenever the following two conditions are met:

1. Receive buffer system is full

All five receive buffers contain valid messages waiting to be read by the application.

2. Another valid message is seen on the bus. This message must be sent from another node, i.e. it must not be transmitted from the respective mSCAN module itself.

At the end of the message in 2. the Time Stamp Register of the oldest message in the receive FIFO is overwritten.

Note: if the message in 2. passes the message filter system the Overrun Interrupt Flag (OVRIF) is also set.

**Workaround**

The application software has to ensure to read the receive messages in due time to avoid data overrun in any case. This will automatically minimize the risk of a Time Stamp Register overwrite event.

**Reduced FLASH programming temperature range and increased programming time for all product marked before date code 0514 (2005 WW14).**

**MUCts01489**

**Description**

The FLASH programming temperature range specification was reduced for product marked before date code 0514 (i.e. up to and including date code 0513). The erratum to the specification stipulates that, for product marked before date code 0514, FLASH programming operations must take place between temperatures of -20°C and 125°C ambient.

Any product marked on or after date code 0514 is guaranteed for FLASH programming operations over the full temp specification of -40°C to 125°C ambient.

The FLASH programming times for product marked before date code 0514 are as follows:

	Programming Time	
	Min.	Max.
Single Word Programming Time (tswpgm)	66.0µs	99.2µs
Flash Burst Programming consecutive word (tbwpgm)	40.4µs	57.8µs
Flash Burst Programming Time for 64 Word row (tbrpgm)	2608.7µs	3742.7µs

The FLASH programming times for product marked on or after date code

0514 are as follows:

	Programming Time	
	Min.	Max.
Single Word Programming Time (tswpgm)	46.0µs	74.5µs
Flash Burst Programming consecutive word (tbwpgm)	20.4µs	31.0µs
Flash Burst Programming Time for 64 Word row (tbrpgm)	1331.2µs	2027.5µs

Actual programming time will vary between the above bounds, depending on FLASH clock and bus clock frequencies.

Important Notes:

- 1. Programming time is internally controlled by the FLASH state machine.  
No action needs to be taken by users in connection with this erratum.
- 2. Both FLASH erase and FLASH read temperature specifications and operation times are unaffected by this erratum.
- 3. EEPROM is unaffected by this erratum.
- 4. This erratum DOES NOT apply to product marked on or after 0514.

Workaround

None.

Description

Upon leaving BDM active mode, the CPU return address is stored temporarily for a few cycles in the BDM shift register. If a BDM command transmission is detected during this time, the return address will be manipulated in the BDM shift register. This situation is likely to occur when a CPU BGND instruction is executed in user code during debugging under the following conditions:

- (i) The BDM module is not enabled AND
- (ii) BDM commands are sent from the host

If this situation occurs, the CPU will execute BDM firmware and will check the status of the ENBDM bit in the BDMSTS register. If the BDM is disabled, the ENBDM bit will be clear, and hence the BDM firmware will be exited and the shift register manipulation described above will occur.

**Workaround**

Avoid using the BGND instruction when the ENBDM bit in the BDMSTS register is cleared.

---

EEPROM Program Failure during Sector-Modify

MUCts02382

**Description**

At oscillator frequencies above 4MHz the Program step of the EEPROM Sector-Modify command can fail depending on the bus frequency. As a result, no programming of the EEPROM occurs. There is no impact to the Erase step of the Sector-Modify command. Since a partial programming of the word cannot occur, there is not a reliability issue caused by the Sector-Modify command if the programmed word is verified.

Oscillator Frequency	Bus Frequency
-----	-----
4MHz	No Issue
8MHz	Fbus <20MHz : No issue
16MHz	Fbus <16MHz : No issue

## Workaround

Use separate Erase and Program commands in place of the Sector-Modify command. If the Sector-Modify command is used and fails the program step as confirmed by a user verification step, a Program command alone can be used to effectively complete the operation since the erase step does successfully erase the sector.

---

## MEBI: Missing ECLK edge on first external access after mode switching

MUCts02415

### Description

If the ECLK is used as an external bus control signal (ESTR=1) the first external access is lost after switching from a single chip mode with enabled ECLK output to an expanded mode. The ECLK is erroneously held in the high phase thus the first external bus access does not generate a rising ECLK edge for the external logic to latch the address. The ECLK stretches low after the lost access resulting in all following external accesses to be valid.

## Workaround

Enter expanded mode with ECLK output disabled (NECLK=1). Enable the ECLK after switching the mode before executing the first external access.

---

## SPI: Disabling slave SPI together with clearing CPHA while SS low locks transmit shift register for the next transmission

MUCts03403

### Description

With the SPI configured as a slave, clearing the SPE bit (to disable the SPI) together with clearing the CPHA bit while the SS pin is low causes the transmit shift register to be locked for the next transmission following the SPI being re-enabled as a slave with SS still being low.

This means new transmit data is not accepted for the first transmission after re-enabling the SPI (indicated by SPTEF staying low after storing transmit data into SPIDR), but for the next following transmission.

## **Workaround**

When disabling the slave SPI, CPHA should not be cleared at the same time.