

---

## Mask Set Errata for Mask 0N22A

### Introduction

This report applies to mask 0N22A for these products:

- MC9S08GB60A
- MC9S08GT60A
- MC9S08GB32A
- MC9S08GT32A

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 0J27F. All standard devices are marked with a mask set number and a date code.

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 "0301" indicates the first week of the year 2003.

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.

### SE194-STOP3DCO: Stop3 Mode DCO Erratum

**Errata type:** Silicon

**Affects:** ICG, Stop3

**Description:** On some devices, during stop3 recovery in FEI, FEE, and SCM modes, the DCO clock can generate high frequency pulses that can cause the MCU to enter a code runaway condition due to excessive bus speed. This condition can occur when the Reduced Frequency Divider value (R) in the ICGC2 register is 1, 2, or 4. The Multiplication Factor value (N) in the ICGC2 register does not matter. Stop recovery by reset is not affected. Supply voltage is not a factor.

- Workaround:**
1. Select a value of R that is greater than or equal to 8 when choosing a system bus frequency. This divides the high frequency DCO clocks to within system specification.
  2. Increase the R value to 8 by writing the ICGC2 register just before the stop instruction. Restore the R value by writing the ICGC2 register to its original value in the service routine of the interrupt that wakes the MCU from stop3 mode.

### **SE160-KEYACC: Attempting to Break Security via Backdoor Key, while running in Flash, sends part into unknown operation**

**Description:** When KEYACC bit is set, as in the case of trying to start a procedure to unlock security, subsequent attempts to access flash will send the part into an unknown state.

This includes trying to read Flash locations that may contain the backdoor key, or trying to execute instructions out of Flash that may be trying to pull the backdoor key from RAM.

**Workaround:** Move routine and key to break security into RAM and execute from RAM, prior to setting KEYACC bit.

### **SE110-TPM: PWM Boundary Case Issues in HCS08 Timer PWM Module (TPM)**

**Errata type:** Silicon

**Affects:** PWM

**Description:** This errata describes boundary case issues that primarily affect the center-aligned PWM mode of operation. While investigating these issues, additional, less significant, issues were discovered. These will be explained, although they should not cause any significant problems in normal applications.

In center-aligned PWM mode, the timer counter counts up until it reaches the modulo value in TPMMODH:TPMMODL, reverses direction, and then counts down until it reaches zero, where it reverses and counts up again. A period of the PWM output is centered around the leading edge of the zero count and the period is considered to start when the count changes from TPMMODH:TPMMODL-1 to TPMMODH:TPMMODL (the same point where the counter changes from up-counting to down-counting). The zero value and the maximum modulo value occur for only one timer count cycle each, while all other values occur twice (once during the down-counting phase and again during the up-counting phase). Therefore, the total period of the PWM signal is two times the value in TPMMODH:TPMMODL.

The value on each TPM timer output pin is controlled by an internal flip-flop that is cleared at reset but is not readable by software. These internal flip-flops change state when timer output compare events or PWM duty cycle compare events occur (when the channel value registers match the timer count registers). This leads to these outputs remaining in a previous state until a compare event occurs after changing the configuration of the timer system. When the timer is initialized the first time after a reset, the state of these output flip-flops is known to be reset (logic low). If the configuration is changed after the channel has been running in another configuration for some period of time, you sometimes do not know the state of these internal flip-flops (and therefore the state of the timer output pins) until a new channel value register compare event occurs. There is nothing improper about these periods before the first event occurs, however some users might be surprised the first time they notice this behavior.

When the MCU is reset, the count (TPMCNTH:TPMCNTL) is reset to 0x0000. If the timer is configured for center-aligned pulse-width modulation (PWM) and then the clock is started, this corresponds to the middle of a PWM period. If the internal flip-flop corresponding to the output was at the inactive level when the PWM started, this would appear as if there was an extra half

period of delay before the first full PWM cycle started. If the internal flip-flop corresponding to the output happened to be at the active level when this PWM was started, a pulse equivalent to half of a normal duty cycle pulse could be produced at the PWM output pin.

There are eight cases discussed in this errata:

- Cases 1 and 2 — These are two error cases near the 100% duty cycle boundary. The first is when the channel value registers are set equal to the modulo value. The second is when the channel value registers are set to one less than the modulo value.
- Cases 3, 4, and 5 — These cases are related to changing the channel value to or from 0x0000. The errors depend upon whether this is done during the first or second half of the center-aligned PWM period. In all of these cases, the workaround strategy is to produce 0% duty cycle with a negative channel value instead of using the 0x0000 value. This can be done by checking any value that is about to be written to the channel value registers, and then decrement the 16-bit value or the high-order byte of the value before writing it to the channel value registers. This produces the desired 0% duty cycle and avoids the problems related to a zero in the channel value registers.
- Case 6 — Although this behavior wasn't discussed in the data sheet, the operation is different than some users might expect. In edge-aligned PWM mode, when the channel value is changed from zero to a non-zero value, the new PWM settings can take an extra half PWM period to take effect. It is unlikely that this would cause any problems in any practical application system.
- Case 7 — This case is more of a clarification of an unusual situation rather than a design problem. This case happens when the prescale factor is changed during operation and only affects center-aligned PWM. It would be very unusual to change the prescale setting after it is set during reset initialization. The prescale flip-flops are not reset when the prescale setting is changed, so the first prescaled clock period after a change may be shorter or longer than expected.
- Case 8 — This case would only arise when a series of unlikely events happened to occur. It affects only center-aligned PWM mode if the timer counter is stopped, reset, and restarted when the count value happened to be equal to the TPMxMODH:TPMxMODL value. Because the timer counter would not normally be stopped during operation in center-aligned PWM mode, this case should never arise in a practical application.

**Workaround:** Case 1: Center-Aligned PWMChannel Value (TPMxCnVH:TPMxCnVL) = Modulo Value (TPMxMODH:TPMxMODL)

This should produce 100% duty cycle where the TPM output pin remains at the active level continuously. Instead, the output remains at the inactive level, which corresponds to 0% duty cycle.

Check any value that is about to be written to the channel value registers. If the value is the same as the modulo value, increment the value before writing it to the channel value register. This workaround will work for any modulo value that is greater than zero and less than 0x7FFF. Setting the channel value to any 2's complement negative value (0x8000 through 0xFFFF) results in 0% duty cycle as expected and described in the original TPM documentation.

Another workaround would be to choose not to use 100% duty cycle in the application. Not all applications require the range to include the 100% duty cycle case.

Case 2: Center-Aligned PWMChannel Value (TPMxCnVH:TPMxCnVL) = Modulo Value Minus 1 (TPMxMODH:TPMxMODL – 1)

This should produce almost 100% duty cycle where the TPM output pin remains at the active level for  $[(TPMxCnVH:TPMxCnVL \times 100)/(TPMxMODH:TPMxMODL)]\%$  of the period. Instead, the output remains at the inactive level which corresponds to 0% duty cycle.

Reduce the prescale factor by a factor of two and then multiply the modulo and channel value settings by a factor of two. In this way, the frequency and resolution of the PWM output remain the same but channel values are always even numbers and are never equal to the modulo setting minus one.

Consider the case of a 20-MHz bus frequency, 25-kHz PWM frequency, and 0.25% resolution on the duty cycle. Before making the adjustments suggested in this workaround, you could have the following setup: Set the modulo to 400 and the prescale factor in PS2:PS1:PS0 to divide by 2 (0:0:1). Each step of the channel value from 0–1–2...398–399–400 would increase the duty cycle by 0.25%.

Increasing the modulo value to 800 and reducing the prescale factor to divide-by one, would still produce the same period or PWM frequency. If the original channel values were multiplied by two (shift left one bit position) before writing them to the channel value register, the resolution would still be 0.25% per step of the channel value, but the channel values would step by 2 each time as in 0-2-4-6...796-798-800. With this workaround, the channel value would never be equal to the modulo value minus one, and the error condition would not arise.

With common HCS08 bus frequencies, practical PWM frequencies, and reasonable resolution requirements, there is enough speed and flexibility in the TPM system so this workaround should work well with all except the most unusual application systems.

Another workaround would be to limit the range of allowed values in the channel value register so it does not include the TPMxMODH:TPMxMODL or (TPMxMODH:TPMxMODL – 1) values. Not all applications require the range to include these values.

#### Case 3: Center-Aligned PWMTPMxCnVH:TPMxCnVL Changed from 0x0000 to a Non-Zero Value

This case occurs only while the counter is counting down (first half of the center-aligned PWM period). The PWM output changes to the active level at the middle of the current PWM period as the count reaches 0x0000 instead of waiting for the start of a new PWM period to begin using the new duty cycle setting.

Use a negative channel value instead of 0x0000 to produce 0% duty cycle. This can be done by checking any value that is about to be written to the channel value registers, and then decrementing the 16-bit value or the high-order byte of this value before writing it to the channel value registers. This produces the desired 0% duty cycle and it avoids the problems related to a zero in the channel value registers.

#### Case 4: Center-Aligned PWMTPMxCnVH:TPMxCnVL Changed from a Non-Zero Value to 0x0000

This case occurs only while the counter is counting up (second half of the center-aligned PWM period) but before the count reaches the channel value setting in TPMxCnVH:TPMxCnVL. The PWM output remains at the active level until the end of the current PWM period instead of finishing the current PWM period using the old channel value setting.

Use a negative channel value instead of 0x0000 to produce 0% duty cycle. This can be done by checking any value that is about to be written to the channel value registers, and then decrement the 16-bit value or the high-order byte of this value before writing it to the channel value registers. This produces the desired 0% duty cycle and it avoids the problems related to a zero in the channel value registers.

#### Case 5: Center-Aligned PWMTPMxCnVH:TPMxCnVL Changed from 0x0000 to a Non-Zero Value

This case occurs only while the counter is counting down (first half of the center-aligned PWM period) and then TPMxCnVH:TPMxCnVL is changed back to 0x0000 during the first half of the next PWM period (while the counter is counting down). This is a very unlikely case in any practical application. The PWM output changes to the active level at the middle of the first

PWM period as the count reaches 0x0000 instead of waiting for the start of a new PWM period to begin using the new duty cycle setting, and then the output remains active until the end of the second PWM period. In this very unusual case, the PWM output remains active for one and one-half PWM periods rather than remaining inactive for the first PWM period and then active for  $2 \times \text{TPMxCnVH}:\text{TPMxCnVL}$  during the next PWM period.

Use a negative channel value instead of 0x0000 to produce 0% duty cycle. This can be done by checking any value that is about to be written to the channel value registers, and then decrementing the 16-bit value or the high-order byte of this value before writing it to the channel value registers. This produces the desired 0% duty cycle and it avoids the problems related to a zero in the channel value registers.

#### Case 6: Edge-Aligned PWM TPMxCnVH:TPMxCnVL Changed from 0x0000 to a Non-Zero Value

This is a minor issue related to edge-aligned PWM when duty cycle is changed from 0x0000 to a non-zero value. This issue is a specification clarification rather than a design error.

In this case, the channel value update occurs at the same time as the new PWM period begins, but due to circuit delays, the update occurs slightly too late for the new duty cycle to take effect for that PWM period and an extra period of 0% duty cycle is produced. This causes the new PWM duty cycle to take effect one PWM period later than expected. This should not cause any application problems so the data book functional description will be changed to clarify this situation.

#### Case 7: Changing the Counter Prescaler while the TPM Counter Is Disabled

This case would not arise in most applications because it would be unusual to change the prescaler at any time other than initial timer setup after reset.

1. TPM counter was previously running
2. Counting is stopped by writing 0:0 to CLKS[1:0]
3. Change prescale value PS[2:1:0] to a different value while keeping clocks off (CLKS[1:0] = 0:0)
4. Clear the counter by writing any value to TPMxCNTH:TPMxCNTL
5. Turn clocks back on by writing to CLKS[1:0]

Unexpected Operation: The prescaler divider flip-flops begin counting from the prior value rather than starting from zero. This can result in the counter detecting the first clock edge after restarting, either earlier or later than expected.

#### Case 8: Center-Aligned PWM, Counter is Stopped, Reset, and Restarted when Counting Up and Count Equals the Modulo Value

This case is extremely unlikely to occur in any practical application because it would be very unusual to stop or reset the TPM counter while using center-aligned PWM mode.

1. TPM counter is counting up in center-aligned PWM mode (second half of a PWM period)
2. Counter is stopped (write CLKS[1:0] = 0:0) when count equals modulo value (the direction would normally change from up counting to down counting at the next clock edge)
3. Counter is reset to 0x0000 by writing any value to TPMxCNTH:TPMxCNTL
4. Counter is turned on again by writing to CLKS[1:0]

Unexpected Operation: Because the internal up/down indicator was not cleared when the counter was reset, the counter begins counting down from 0x0000 to 0xFFFF-0xFFFE... This causes the timing of the first PWM period after the counter reset to be longer than expected.

## SE105-IRQ: IRQ Pin Pull-down Resistor When IRQ Configured for Rising Edge Detection

**Description:** When the IRQ pin is configured for rising edge detection (IRQEDG=1) rather than the usual falling edge/low level detection, and the internal pull-down device is enabled (IRQPE=1), the pull-down device can fail to pull the IRQ pin to a valid low level. This problem is worse at high  $V_{DD}$  and room temperature to cold temperature and an external pull-down resistor does not guarantee correct operation.

**Workaround:** There is no known workaround for this problem using a rising edge IRQ on this mask set. The IRQ pin should be configured for falling edges and low levels (IRQEDG=0) to avoid this problem. If a rising edge interrupt is required in the application, consider using any of the KBI pins that can be configured to detect rising edges or a TPM input capture function instead of the IRQ pin.

## SE97-IRQ\_KBI: Active Edge on IRQ or KBI May Not Be Detected

**Errata type:** Silicon

**Affects:** KBI

**Description:** When the IRQ or KBI modules are enabled, an active (falling or rising, depending on configuration settings) edge on the IRQ or KBI pin may not be detected if it transitions exactly two bus clock edges before the bus clock ceases upon stop mode entry. The IRQ/KBI interrupt will not occur and the MCU will remain in stop mode. Subsequent active edges on the IRQ or KBI pin will generate interrupts and wake the MCU from stop mode.

**Workaround:** NA

## SE96-IRQ\_KBI: Possible High Current in Stop Mode If KBI/IRQ Enabled

**Description:** In stop mode, with the IRQ or KBI pin functions enabled but the IRQ/KBI interrupt disabled, a toggle on the IRQ/KBI pin will turn on the voltage regulator and ICG, causing the clocks to run. This results in higher stop  $I_{DD}$ . In this condition, the CPU is halted (as if in wait mode) and the chip level interrupt is not generated. This higher current condition can also happen if the flag is set at time of stop entry.

**Workaround:** This problem applies to the non-interrupt functions that are shared with IRQ and KBI functions. Because interrupts from IRQ or KBI are disabled, these pins are not used to wake up the MCU from stop. To prevent signals on IRQ or KBI from causing an unexpected partial wakeup from stop in these applications, disable the IRQ and KBI modules just before entering stop mode. Also verify that any previous interrupt flags associated with IRQ and KBI have been cleared.

## SE90B-Stop3OSCSTEN-WA: Clocks Active in Stop3 Mode

**Description:** When keeping an external clock active during stop3 (OSCSTEN = 1), the ICG may indicate a loss of clock when there is really no problem. A false loss of clock will wake the MCU up from stop mode. The false loss of clock has been observed on the bench as high as 2.1 V, but it is possible for it to occur at higher voltages as well — up to about 2.2 V, depending on wafer fab processing. This issue is worst case with the low range crystal oscillator (RANGE = 0, REFS = 1) and at lower temperatures.

**Workaround:** To prevent a false loss of clock signal from waking the MCU from stop mode, set the LOCD bit in the ICGC1 register to disable the loss of clock detection. Disabling the loss of clock detection has an added benefit of reducing the stop  $I_{DD}$  from 14  $\mu\text{A}$  to 5  $\mu\text{A}$  (typical).



## **How to Reach Us:**

### **Home Page:**

[www.freescale.com](http://www.freescale.com)

### **Web Support:**

<http://www.freescale.com/support>

### **USA/Europe or Locations Not Listed:**

Freescale Semiconductor  
 Technical Information Center, EL516  
 2100 East Elliot Road  
 Tempe, Arizona 85284  
 +1-800-521-6274 or +1-480-768-2130  
[www.freescale.com/support](http://www.freescale.com/support)

### **Europe, Middle East, and Africa:**

Freescale Halbleiter Deutschland GmbH  
 Technical Information Center  
 Schatzbogen 7  
 81829 Muenchen, Germany  
 +44 1296 380 456 (English)  
 +46 8 52200080 (English)  
 +49 89 92103 559 (German)  
 +33 1 69 35 48 48 (French)  
[www.freescale.com/support](http://www.freescale.com/support)

### **Japan:**

Freescale Semiconductor Japan Ltd.  
 Headquarters  
 ARCO Tower 15F  
 1-8-1, Shimo-Meguro, Meguro-ku,  
 Tokyo 153-0064  
 Japan  
 0120 191014 or +81 3 5437 9125  
[support.japan@freescale.com](mailto:support.japan@freescale.com)

### **Asia/Pacific:**

Freescale Semiconductor China Ltd.  
 Exchange Building 23F  
 No. 118 Jianguo Road  
 Chaoyang District  
 Beijing 100022  
 China  
 +86 10 5879 8000  
[support.asia@freescale.com](mailto:support.asia@freescale.com)

### **For Literature Requests Only:**

Freescale Semiconductor Literature Distribution Center  
 1-800-441-2447 or +1-303-675-2140  
 Fax: +1-303-675-2150  
[LDCForFreescaleSemiconductor@hibbertgroup.com](mailto:LDCForFreescaleSemiconductor@hibbertgroup.com)

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductors products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claims alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

RoHS-compliant and/or Pb-free versions of Freescale products have the functionality and electrical characteristics as their non-RoHS-complaint and/or non-Pb-free counterparts. For further information, see <http://www.freescale.com> or contact your Freescale sales representative.

For information on Freescale's Environmental Products program, go to <http://www.freescale.com/epp>.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© 2011 Freescale Semiconductor, Inc.