

## Mask Set Errata for Mask 0N49M

This report applies to mask 0N49M for these products:

- WPR1516

Errata ID	Errata Title
6964	ADC: Flag RSTAR{EIF is set unexpectedly in corner cases
8188	ADC: High current in Stop Mode
7914	PIT: After enabling the Periodic Interrupt Timer (PIT) clock gate, an attempt to immediately enable the PIT module may not be successful.

### **e6964: ADC: Flag RSTAR{EIF is set unexpectedly in corner cases**

**Errata type:** Errata

**Description:** In case of repeated Restart Request overruns (Restart overrun occurs again before previous Restart overrun could be handled) the RSTAR{EIF is erroneously set if:

- The first Restart Request overrun occurs correctly with a simultaneous Sequence Abort Request AND
- The second Restart Request overrun occurs without Sequence Abort Request, which is a flow control failure but for overrun situation the issue should not be flagged.

Misbehavior:

The ADC flags the flow control failure (RSTAR{EIF bit is set) in the second overrun which should not happen. Overruns are tracked quietly and executed as soon as Sequence Abort turnaround time allows.

**Workaround:** If possible, avoid multiple Restart Request overrun scenarios.

### **e8188: ADC: High current in Stop Mode**

**Errata type:** Errata

**Description:** The ADC can take a higher current in Stop Mode (500µA increasing over time to about 1mA per ADC instance) if the ADC is enabled and conversions have been done.

There is a software workaround available.

**Workaround:** This software workaround has been validated.

The subroutine ADC0\_stop\_current\_workaround takes about 105 bus clock cycles (ECLK) on S12ZVM128.

The subroutine must be executed for each ADC instance performing conversions before Stop Mode entry.

Before the workaround subroutine is executed, the ADC must be initialized (command list pointers and result list pointers have valid addresses).

Please note:

- All the conversion commands of List 0 must be valid (no illegal channel, no illegal SMP value)
- ADCFMT[SRES] value must not be illegal
- The command fetch must not cause an illegal access flag

```
void main(void) {
...
DisableInterrupts; // Shutdown sequence is not interruptible until STOP
...
ADC0_stop_current_workaround(); // Subroutine must be executed with interrupt protection
asm(andcc #0x6f); /* CCW settings: S = 0, I = 0 */
asm(stop); /* MCU enters Stop mode if S = 0 in CCW */
/* I-mask interrupts will be enabled because I = 0 in CCW */
...
}
//Workaround to avoid ADC high current during STOP mode
//ADC will be disabled in this function
//Therefore following register are cleared:
//Address | Register | Description
//-----+-----+-----
//0x02 | ADCSTS | CSL_SEL, RVL_SEL will be restored at the end of the function
//0x08 | ADCEIF | Error interrupt flags
//0x09 | ADCIF | Interrupt flags (SEQAD_IF, CONIF_OIF)
//0x0C-0x0D:| ADCCONIF | Conversion interrupt flags, EOL (end of list) interrupt flag
//0x0E-0x0F:| ADCIMDRI | Intermediate result information (must be stored by the customer
before,
// | | if this information is still needed after the function execution)
//0x10 | ADCEOLRI | EOL result information (must be stored by the customer before,
// | | if this information is still needed after the function execution)
//0x1C | ADCCIDX |
//0x20 | ADCRIDX |
```

```

void ADC0_stop_current_workaround (void) {
byte tmp_ADCxCTL_0 = ADC0CTL_0; // Save customer settings, these values will be restored
afterwards
byte tmp_ADCxTIM = ADC0TIM;
byte tmp_ADCxSTS = ADC0STS;
ADC0CTL_0 = 0x00; // Disable ADC
ADC0TIM = 0x00; // ADC is set to maximum frequency
// Device specification of allowed frequency is ignored,
// the ADC conversion is stopped when reaching error state
// There is no conversion result generated.
ADC0CTL_0 = 0x88; // ADC is enabled, single access mode data bus, restart mode
ADC0CTL_0 = 0x88; // Re-do in order to guarantee ADC is ready for requests before Restart
Event occurs
ADC0FLWCTL = 0x20; // ADC is restarted, RSTA bit is set: the first command of list 0 is
// loaded from memory. The command type does not matter, the conversion
// is immediately stopped
while (ADC0FLWCTL_RSTA == 1) {} // Wait for restart completion (within a few clock cycles)
ADC0FLWCTL = 0x40; // Start conversion (TRIG)
ADC0FLWCTL = 0x40; // The second TRIG immediately generates a TRIG_EIF, ERROR state
is entered
while (ADC0EIF_TRIG_EIF == 0) {} // Wait for trigger error interrupt flag (within a few clock
cycles)
ADC0CTL_0_ADC_SR = 1; // Execute ADC soft-reset (SR), ADC enters IDLE state
while (ADC0STS_READY == 0) {} // Wait for ADC soft-reset done (within a few clock cycles)
ADC0CTL_0 = 0x00; // ADC is disabled
ADC0TIM = tmp_ADCxTIM; //Restore previous customer settings
ADC0STS = tmp_ADCxSTS;
ADC0CTL_0 = tmp_ADCxCTL_0; // ADC0CTL_0 is the last one to be restored, in case ADC
was enabled before...
}

```

**e7914: PIT: After enabling the Periodic Interrupt Timer (PIT) clock gate, an attempt to immediately enable the PIT module may not be successful.**

**Errata type:** Errata

**Description:** If a write to the PIT module enable bit (PIT\_MCR[MDIS]) occurs within two bus clock cycles of enabling the PIT clock gate in the SIM\_CG register, the write will be ignored and the PIT will fail to enable.

**Workaround:** Insert a read of the PIT\_MCR register before writing to the PIT\_MCR register. This guarantees a minimum delay of two bus clocks to guarantee the write is not ignored.

**How to Reach Us:**

**Home Page:**  
[freescale.com](http://freescale.com)

**Web Support:**  
[freescale.com/support](http://freescale.com/support)

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

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale 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 does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: [freescale.com/SalesTermsandConditions](http://freescale.com/SalesTermsandConditions).

Freescale and the Freescale logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. All other product or service names are the property of their respective owners. ARM, the ARM Power logo, Cortex, and Cortex M0+ are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/ or elsewhere. All rights reserved.

© 2014 Freescale Semiconductor, Inc.

Document Number: WPR1516\_0N49M  
Rev. 29 OCT 2014

