

## MSP430FE423 Device Erratasheet

---



---



---

### 1 Functional Errata Revision History

Errata impacting device's operation, function or parametrics.

✓ The check mark indicates that the issue is present in the specified revision.

Errata Number	Rev K	Rev I	Rev H	Rev G	Rev E
ESP1	✓	✓	✓	✓	✓
ESP2	✓	✓	✓	✓	✓
ESP3	✓	✓	✓	✓	✓
ESP4	✓	✓	✓	✓	✓
ESP5	✓	✓	✓	✓	✓
FLL3	✓	✓	✓	✓	✓
MPY2			✓		✓
SD1	✓	✓	✓	✓	✓
SD2	✓	✓	✓	✓	✓
TA12	✓	✓	✓	✓	✓
TA16	✓	✓	✓	✓	✓
TA21	✓	✓	✓	✓	✓
TAB22	✓	✓	✓	✓	✓
US15	✓	✓	✓	✓	✓
WDG2	✓	✓	✓	✓	✓

### 2 Preprogrammed Software Errata Revision History

Errata impacting pre-programmed software into the silicon by Texas Instruments.

✓ The check mark indicates that the issue is present in the specified revision.

The device doesn't have Software in ROM errata.

### 3 Debug only Errata Revision History

Errata only impacting debug operation.

✓ The check mark indicates that the issue is present in the specified revision.

Errata Number	Rev K	Rev I	Rev H	Rev G	Rev E
EEM20	✓	✓	✓	✓	✓

## 4 Fixed by Compiler Errata Revision History

Errata completely resolved by compiler workaround. Refer to specific erratum for IDE and compiler versions with workaround.

✓ The check mark indicates that the issue is present in the specified revision.

Errata Number	Rev K	Rev I	Rev H	Rev G	Rev E
<a href="#">CPU4</a>	✓	✓	✓	✓	✓

Refer to the following MSP430 compiler documentation for more details about the CPU bugs workarounds.

### TI MSP430 Compiler Tools (Code Composer Studio IDE)

- [MSP430 Optimizing C/C++ Compiler](#): Check the --silicon\_errata option
- [MSP430 Assembly Language Tools](#)

### MSP430 GNU Compiler (MSP430-GCC)

- [MSP430 GCC Options](#): Check -msilicon-errata= and -msilicon-errata-warn= options
- [MSP430 GCC User's Guide](#)

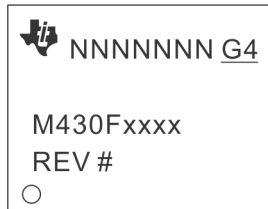
### IAR Embedded Workbench


- [IAR workarounds for msp430 hardware issues](#)

## 5 Package Markings

### PM64

### LQFP (PM), 64 Pin



# = Die revision  
 = Pin 1 location  
N = Lot trace code

## 6 Detailed Bug Description

### CPU4 *CPU Module*

---

**Category** Compiler-Fixed

**Function** PUSH #4, PUSH #8CPU4 - Bug

**Description** The single operand instruction PUSH cannot use the internal constants (CG) 4 and 8. The other internal constants (0, 1, 2, -1) can be used. The number of clock cycles is different:

PUSH #CG uses address mode 00, requiring 3 cycles, 1 word instruction

PUSH #4/#8 uses address mode 11, requiring 5 cycles, 2 word instruction

**Workaround** Refer to the table below for compiler-specific fix implementation information.

IDE/Compiler	Version Number	Notes
IAR Embedded Workbench	IAR EW430 v2.x until v6.20	User is required to add the compiler flag option below. --hw_workaround=CPU4
IAR Embedded Workbench	IAR EW430 v6.20 or later	Workaround is automatically enabled
TI MSP430 Compiler Tools (Code Composer Studio)	v1.1 or later	
MSP430 GNU Compiler (MSP430-GCC)	MSP430-GCC 4.9 build 167 or later	

### EEM20 *EEM Module*

---

**Category** Debug

**Function** Debugger might clear interrupt flags

**Description** During debugging read-sensitive interrupt flags might be cleared as soon as the debugger stops. This is valid in both single-stepping and free run modes.

**Workaround** None.

### ESP1 *ESP Module*

---

**Category** Functional

**Function** Suspending the ESP430CE1

**Description** Suspending the ESP430 may create an invalid interrupt which can lead to a reset-like behavior of the module.

**Workaround** Set the bit 0x08 together with the ESPSUSP bit:

```
bis.w #08h+ESPSUSP, &ESPCTL
```

This bit also must be cleared when the suspend mode is exited.

```
bic.w #08h+ESPSUSP, &ESPCTL
```

NOTE:

- After suspending the ESP430CE1 it can take up to 9 MCLK clock cycles before the

```

CPU can access the SD16 registers.
- An interrupt service routine for the SD16 is required.

// Shut down ESP (set Embedded Signal Processing into
// "Suspend" mode)
// ensure that it is not in measurement or calibration mode,
ESPCTL |= 0x08 + ESPSUSP;
// Set ESP into Suspend Mode
// incl. Bug Fix for Suspend Mode
// wait 9 clocks until proper access to the SD16 is possible
__delay_cycles(9);
MBCTL &= ~(IN0IFG + IN0IE);
// Clear any Pending MB interrupt and disable
// ESP interrupt
SD16CTL &= ~SD16REFON; // Switch Reference off

```

## ESP2

### *ESP Module*

#### Category

Functional

#### Function

Negative Energy Flag Operation.

#### Description

The NEGENFG negative energy flag treatment inside the ESP430 module is executed after each mains cycle and is set according to the energy accumulated during this period. Therefore the flag is set under two conditions:

- if during at least one mains cycle period over the last second the accumulated energy in the ACTENSPER register was negative
- if the accumulated energy in the ACTENERGY register is negative.

#### Workaround

If the indication of the negative energy status is required by the application, it can be done with the following sequence in CPU software.

1. Set negative energy bits: NE0 = 0, NE1 = 1 (negative energy is summed)
2. Perform the required steps manually software after new energy samples are available from the ESP430CE1:

- Check if the energy is negative and set the negative energy flag
- Correct the energy to positive values if required.

```

if ((msg_data & ENRDYFG))
{
if ((ACTENERGY1_HI & 0x8000) > 0 ) { // Negative Energy measured?
negenfg = 1; // set global neg. Energy Flag
}else{
negenfg = 0; // clear global neg. Energy Flag
}

if (negenfg == 1){ // Negative Energy measured?

```



```
ESPCTL |= 0x08 + ESPSUSP; // Set ESP into Suspend Mode
// incl. Bug Fix for Suspend Mode
}
```

### ESP5 *ESP Module*

**Category** Functional

**Function** In two current mode, CAPIND detection works on current channel 1 only

**Description** When using the ESP430 module in the two current mode, CAPIND detection works on current channel 1 only and does not work when using current channel 2. This could deliver wrong values if the current in channel 1 is near or equal 0.

**Workaround** None

### FLL3 *FLL+ Module*

**Category** Functional

**Function** FLLDx = 11 for /8 may generate an unstable MCLK frequency

**Description** When setting the FLL to higher frequencies using FLLDx = 11 (/8) the output frequency of the FLL may have a larger frequency variation (e.g. averaged over 2sec) as well as a lower average output frequency than expected when compared to the other FLLDx bit settings.

**Workaround** None

### MPY2 *MPY Module*

**Category** Functional

**Function** Multiplier Result register corruption

**Description** Depending on the address of the write instruction, writing to the multiplier result registers (RESHI, RESLO, or SUMEXT) may corrupt the result registers. The address dependency varies between a 2-word and a 3-word instructions.

**Workaround** Ensure that a write instruction to an MPY result register (for example, mov.w #200, &RESHI) is not located at an address with the four least significant bits shown in Table 1:

Table 1. Sensitive Addresses for Write Access to MPY Result Registers MAB[3:0]

RESLOW 013Ah		RESHI 013Ch		SUMEXT 013Eh	
3 Word	2 Word	3 Word	2 Word	3 Word	2 Word
2	4	2	4	2	4
6	8	4	6	6	8
A	C	A	C	A	C
E	0	C	E	-	-

<b>SD1</b>	<b><i>SD16 Module</i></b>
<b>Category</b>	Functional
<b>Function</b>	Reduced SINAD performance if SD16 clock source is greater than 6 MHz
<b>Description</b>	If the frequency of the SD16 input clock source is greater than 6 MHz, the performance of the SD16 may be degraded due to noise influencing the analog measurements under reduced SINAD.
<b>Workaround</b>	<p>Writing 0x48 to memory location 0xBF configures the SD16 for optimized performance at input clock frequencies greater than 6 MHz.</p> <p>Include the following code:</p> <pre>*(unsigned char*) 0xBF=0x48; // Write value 0x48 to memory address 0xBF</pre>
<b>SD2</b>	<b><i>SD16 Module</i></b>
<b>Category</b>	Functional
<b>Function</b>	Internal short measurement influenced by external Ax.0 analog voltages
<b>Description</b>	Applying a common mode voltage other than VSS or a differential voltage to the analog inputs of the SD16 may influence the measurement accuracy when converting the internal short channel (A7). The error under these conditions is proportional to the common-mode or differential voltage and is typically 150+ LSBs.
<b>Workaround</b>	Avoid applying common-mode voltages other than VSS, or a differential input voltage during the measurement of the internal short channel.
<b>TA12</b>	<b><i>TIMER_A Module</i></b>
<b>Category</b>	Functional
<b>Function</b>	Interrupt is lost (slow ACLK)
<b>Description</b>	Timer_A counter is running with slow clock (external TACLK or ACLK) compared to MCLK. The compare mode is selected for the capture/compare channel and the CCRx register is incremented by one with the occurring compare interrupt (if TAR = CCRx). Due to the fast MCLK the CCRx register increment (CCRx = CCRx+1) happens before the Timer_A counter has incremented again. Therefore the next compare interrupt should happen at once with the next Timer_A counter increment (if TAR = CCRx + 1). This interrupt gets lost.
<b>Workaround</b>	Switch capture/compare mode to capture mode before the CCRx register increment. Switch back to compare mode afterwards.
<b>TA16</b>	<b><i>TIMER_A Module</i></b>
<b>Category</b>	Functional
<b>Function</b>	First increment of TAR erroneous when IDx > 00
<b>Description</b>	The first increment of TAR after any timer clear event (POR/TACLK) happens immediately following the first positive edge of the selected clock source (INCLK, SMCLK, ACLK or TACLK). This is independent of the clock input divider settings (ID0,



ID1). All following TAR increments are performed correctly with the selected IDx settings.

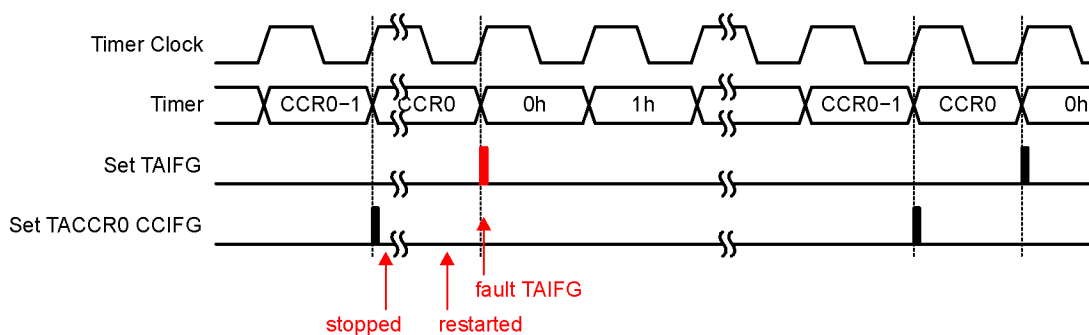
**Workaround** None

**TA21** *TIMER\_A Module*

**Category** Functional

**Function** TAIFG Flag is erroneously set after Timer A restarts in Up Mode

**Description** In Up Mode, the TAIFG flag should only be set when the timer counts from TACCR0 to zero. However, if the Timer A is stopped at TAR = TACCR0, then cleared (TAR=0) by setting the TACLK bit, and finally restarted in Up Mode, the next rising edge of the TACLK will erroneously set the TAIFG flag.



**Workaround** None.

**TAB22** *TIMER\_A/TIMER\_B Module*

**Category** Functional

**Function** Timer\_A/Timer\_B register modification after Watchdog Timer PUC

**Description** Unwanted modification of the Timer\_A/Timer\_B registers TACTL/TBCTL and TAIV/TBIV can occur when a PUC is generated by the Watchdog Timer(WDT) in Watchdog mode and any Timer\_A/Timer\_B counter register TACCRx/TBCCRx is incremented/decremented (Timer\_A/Timer\_B does not need to be running).

**Workaround** Initialize TACTL/TBCTL register after the reset occurs using a MOV instruction (BIS/BIC may not fully initialize the register). TAIV/TBIV is automatically cleared following this initialization.

Example code:

```
MOV.W #VAL, &TACTL
```

or

```
MOV.W #VAL, &TBCTL
```

Where, VAL=0, if Timer is not used in application otherwise, user defined per desired function.

**US15** *USART Module*

---

<b>Category</b>	Functional
<b>Function</b>	UART receive with two stop bits
<b>Description</b>	<p>USART hardware does not detect a missing second stop bit when SPB = 1.</p> <p>The Framing Error Flag (FE) will not be set under this condition and erroneous data reception may occur.</p>
<b>Workaround</b>	None (Configure USART for a single stop bit, SPB = 0)

**WDG2**


---

***WDT Module***

<b>Category</b>	Functional
<b>Function</b>	Incorrectly accessing a flash control register
<b>Description</b>	If a key violation is caused by incorrectly accessing a flash control register, the watchdog interrupt flag is set in addition to the expected PUC.
<b>Workaround</b>	None

## 7 Document Revision History

Changes from family erratasheet to device specific erratasheet.

1. Revision H was removed
2. Revision G was added
3. Errata TA22 was renamed to TAB22
4. Description for TAB22 was updated

Changes from device specific erratasheet to document Revision A.

1. Errata EEM20 was added to the errata documentation.

Changes from document Revision A to Revision B.

1. Errata TA21 was added to the errata documentation.

Changes from document Revision B to Revision C.

1. Silicon Revision I was added to the errata documentation.

Changes from document Revision C to Revision D.

1. Silicon Revision H was added to the errata documentation.

Changes from document Revision D to Revision E.

1. Package Markings section was updated.

Changes from document Revision E to Revision F.

1. TA21 Description was updated.

Changes from document Revision F to Revision G.

1. Silicon Revision K was added to the errata documentation.

Changes from document Revision G to Revision H.

1. Function for CPU4 was updated.
2. Workaround for CPU4 was updated.

Changes from document Revision H to Revision I.

1. Erratasheet format update.
2. Added errata category field to "Detailed bug description" section

## IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ("TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products (<http://www.ti.com/sc/docs/stdterms.htm>), [evaluation modules](#), and [samples](http://www.ti.com/sc/docs/sampterm.htm) (<http://www.ti.com/sc/docs/sampterm.htm>).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265  
Copyright © 2018, Texas Instruments Incorporated