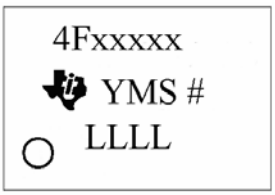
MSP430F11x2/12x2 Device Erratasheet
Current Version

Devices	Rev:	BCL5	CPU4	PORT3	RES4	TA12	TA13	TA16	US13	US15	WDG2
MSP430F1122	E	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MSP430F1132	E	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MSP430F1222	D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
MSP430F1232	D	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Note: See Appendix for prior revisions

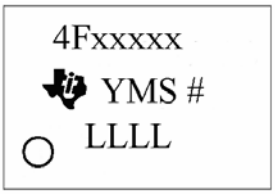
Package Markings

PW20: TSSOP(PW) 20-pin



YM = Year and Month Date Code
LLLL = LOT Trace Code
S = Assembly Site Code
# = DIE Revision
o = PIN 1

PW28: TSSOP(PW) 28-pin



YM = Year and Month Date Code
LLLL = LOT Trace Code
S = Assembly Site Code
# = DIE Revision
o = PIN 1

DW20: SOP(DW) 20-pin

YM = Year and Month Date Code
LLLL = LOT Trace Code
S = Assembly Site Code
# = DIE Revision
o = PIN 1

DW28: SOP(DW) 28-pin

YM = Year and Month Date Code
LLLL = LOT Trace Code
S = Assembly Site Code
# = DIE Revision
o = PIN 1

Package Markings (continued)

RGE32: QFN(RHB) 32-pin

YM = Year and Month Date Code
LLLL = LOT Trace Code
S = Assembly Site Code
# = DIE Revision
o = PIN 1

Detailed Bug Description

BCL5 BCL5 – Bug description

Module: Basic Clock, Function: RSELx bit modifications can generate high frequency spikes on MCLK

When DIVMx = 00 or 01 and the RSELx bits of the Basic Clock Module are incremented or decremented in steps of 2 or greater, the DCO output may momentarily generate high frequency spikes at MCLK, which may corrupt CPU operation. This is not an issue when DIVMx = 10 or 11.

Workaround:
Set DIVMx = 10 or 11 to divide the MCLK input prior to modifying RSELx. Once the RSELx bits are configured as desired, the DIVMx setting can be returned to the original selection.

CPU4 CPU4 - Bug description:

Module: CPU, Function: PUSH #4, PUSH #8

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 Assembler version 1.08 and higher produces correct code. 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 implemented in assembler.
No fix planned.

PORT3 Port3 - Bug description:

Module: PORT1/2, Function: Port interrupts can get lost

Port interrupts can get lost if they occur during CPU
access of the P1IFG and P2IFG registers.

Workaround:
None

RES4 RES4 - Bug description:

Module: General, Reset, No reset if external resistor exceeds certain value

No reset of the device is performed if the external pull down
resistor on RST/NMI pin is above a certain limit. The limits are:
VCC=1.8V: maximum pull down resistor = 12kohm
VCC=3.0V: maximum pull down resistor = 5kohm
VCC=3.6V: maximum pull down resistor = 2.5kohm
In addition, a higher current consumption occurs during high/low
RST/NMI signal transition when using improper resistors.

Workaround:
Use external resistors below the mentioned values.

Detailed Bug Description (continued)

TA12 TA12 - Bug description:

Module: TimerA, Function: Interrupt is lost (slow ACLK)

TimerA 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 1 with the occurring compare interrupt (if TAR = CCRx).
Due to the fast MCLK the CCRx register increment (CCRx = CCRx+1) happens before the TimerA counter has incremented again. Therefore the next compare interrupt should happen at once with the next TimerA counter increment (if TAR = CCRx + 1). This interrupt gets lost.

Workaround:
Switch capture/compare mode to capture mode before the CCRx register increment. Switch back to compare mode afterwards.

TA13 TA13 - Bug description:

Module: TimerA, Function: Unintended modification of OUTx signal

If TimerA is used with an asynchronous clock to MCLK, the PWM output can be disturbed during a modification of the CCTLx register bit OMOD[2] of the corresponding CCRx.

Workaround:
Disable TimerA clock or use the same clock source for TimerA and MCLK.

TA16 TA16 - Bug description:

Module: TimerA, Function: First increment of TAR erroneous when IDx > 00

The first increment of TAR after any timer clear event (POR/TACLR) 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

US13 US13 - Bug description:

Module: USART0, USART1, Function: Unpredictable program execution

USART interrupts requested by URXS can result in unpredictable program execution if this request is not served within two bit times of the received data.

Workaround:
Ensure that the interrupt service routine is entered within two bit times of the received data.

Detailed Bug Description (continued)

US15 US15 - Bug description:

Module: USART0, USART1, Function: UART receive with two stop bits

USART hardware does not detect a missing second stop bit when SPB = 1.
The Framing Error Flag (FE) will not be set under this condition and erroneous data reception may occur.

Workaround:
None (Configure USART for a single stop bit, SPB = 0)

WDG2 WDG2 - Bug description:

If a key violation is caused by incorrectly accessing a flash control register, the watchdog interrupt flag is set in addition to a correctly generated PUC.

Workaround:
None

Appendix: Prior Versions

None