**Freescale Semiconductor**
Engineering Bulletin

EB636
Rev. 1, 8/2004

# PMF Module Workaround for MC9S12E128 2L15P Mask Set

By **Brice Scharmann and Jim Williams**
**Austin, Texas**

## Introduction

The purpose of this engineering bulletin is to document one possible workaround associated with the pulse width modulator with fault protection (PMF) module interrupt errata (MUCts01254) in the MC9S12E128 2L15P mask set.

## Symptom

When the PMF is set up with synchronized generators (MTG = 0), the interrupts for generators B and C can not be cleared by clearing the reload flag bit. Interrupts for generators B and C can not be cleared because when the PMF is in single time base mode, the read and write enables for generators B and C are always zero and are not synchronized with generator A's read and write enables.



freescale™ semiconductor

## Detailed Description of Issue

When the PMF is set to single time base mode (MTG = 0), reload interrupts from generators B and C can not be cleared simply by clearing the reload flag in the PMFFQCA register.

In single time base mode (MTG = 0), there is no way to clear reload interrupts B and C. These interrupts will always be pending when using reload interrupts in single time base mode only.

## Workaround

The only known workaround is to use the PMF in multi-time base mode and synchronize the three generators manually via software. The software required is detailed here.

```
void PMF_Init(void)
{
// Start of pre-call setup
// The register names below are defined by Processor Expert. They are
// automatically included in any Metrowerks project which includes Processor Expert.
        PMFCFG0_EDGEA = 0;                      // Set = edge, clear = centered
        PMFCFG0_INDEPA = 0;                     // Set = independent
        PMFCFG0_INDEPB = 0;                     // Set = independant
        PMFCFG0_INDEPC = 0;                     // Set = independent
        PMFCFG1 = 0x00;                         // Top/bot polarity (0-3F)
// If modifying the previous line, be careful that motors are not started during initialization.
// The following value is not final. It will be modified by the program to the correct value.
        PMFDTMA = 0;                            // Set deadtime (0-7FFF)
        PMFVAL0 = 50;                           // The pulse width (0-7FFF).
        PMFMODA = 100;                          // The PWM period (0-7FFF).
        PMFFQCA_PRSCA = 2;                      // Selects the PWM clock frequency (0-3)
        PMFFQCA_LDFQA = 0;                      // Load Frequency A (0-15)
        PMFFQCA_HALFA = 0;                      // Set = enable half-cycle reloads
        (void)PMF_MUCts01254_emulate();         // For E128
//      PMFCFG0_MTG = 0;                        // For E256
        PMFDTMA = 150;                          // Set deadtime (0-7FFF)
        PMFVAL0 = 750;                          // The pulse width (0-7FFF).
        PMFVAL2 = 750;                          // The pulse width (0-7FFF).
        PMFVAL4 = 750;                          // The pulse width (0-7FFF).
        PMFMODA = 1502;                         // The PWM period (0-7FFF).
        PMFCFG0_WP = 1;                         // Set = enable write protection

        PMFFQCA_PWMRFA = 1;             /* Reset interrupt request flag */
        PMFENCA_PWMRIEA = 1;            /* Enable interrupt */
}

//******************
//   Start
//******************
int PMF_MUCts01254_emulate(void)
{
        __asm{

        ; This will emulate the effect of having MTG = 0.
        ; Generators A, B and C will be synched up.
```

```
START_OF_SYNCH_UP:

; Turn off output of each generator
                MOVB    #$00, PMFOUTB
;               MOVB    #$3F, PMFOUTC

; Set MTG to 1, and preserve alignment
                LDAB    PMFCFG0          ; get config reg.
                ANDB    #$08             ; mask edgeA bit
                BNE     EDGE1
                BSET    PMFCFG0,#$40     ; MTG = 1     For Center aligned
                BRA     NEXT1
EDGE1:          BSET    PMFCFG0,#$78     ; MTG = 1 EDGx = 1For Edge aligned

; Set dead time for each generator from value in PMFDTMA
NEXT1:          LDX     PMFDTMA
                STX     PMFDTMB
                STX     PMFDTMC

; Setup duty cycle for each generator from value in PMFVAL0
                LDX     PMFVAL0
                STX     PMFVAL1
                STX     PMFVAL2
                STX     PMFVAL3
                STX     PMFVAL4
                STX     PMFVAL5

; Set up period of output waveform from value in PMFMODA
                LDX     PMFMODA
                STX     PMFMODB
                STX     PMFMODC

; Setup the frequency control registers from value in PMFFQCA
                LDAA    PMFFQCA
                STAA    PMFFQCB
                STAA    PMFFQCC

;Set the LDOKx bit for each of the Generators A,B,C

                LDAA    PMFENCA          ;Must read bit first
                MOVB    #$02,PMFENCA     ;This sets LDOKA

                LDAA    PMFENCB          ;Must read bit first
                MOVB    #$02,PMFENCB     ;This sets LDOKB

                LDAA    PMFENCC          ;Must read bit first
                MOVB    #$02,PMFENCC     ;This sets LDOKC

;*****************************************************
; The start of the next period for generator A is defined
; by the following equation:
; DELAY_TIME = ((PMFMODA*ALIGNMENT*PRESCALE_VALUE)/3) - 3
; Below are the steps to implement the equation
;*****************************************************

; Need to subtract the amount of cycles that each of the
; following instructions take to execute.
;
;       TFR     X, D                    -------> Takes 1 cycle.
;                                                Need the /3 because
;                               the DBNE instr take 3 cycles.
;       DBNE    A, DELAY                        -------> Takes 3 cycles (REL).
;
```

```
; Calculate delay to start generator B and C

; First bring current alignment and pre-scalar into double accumulator
                  LDAB   PMFCFG0          ; get config reg.
                  ANDB   #$08             ; mask edgeA bit
                  BNE    EDGE2
                  LDAB   #$02             ; set mult by 2 for center.
                  BRA    NEXT2
EDGE2:            LDAB   #$01             ; set mult by 1 for edge
NEXT2:            LDAA   PMFFQCA          ; get pre-scalar
                  ANDA   #$06             ; mask for Pre-scalar bits
                  CMPA   #$00             ;
                  BEQ    NO_MULT
                  CMPA   #$06             ;
                  BEQ    MULT_8
                  BRA    MULT
MULT_8:           LDAA   #$08
MULT:             MUL

NO_MULT:
; Second multiple alignment by modulus
                  LDY    PMFMODA          ; Get modulus
                  EMUL

; Now D holds the result.
; NOTE:  The EMUL opcode stores the result in both Y and D.
;       Since the modulus is only 15 bits and the variable alignment
;       is only 1 or 2 there will never be any data in index register Y.

; Third divide D by 3 and see if there is a remainder
                  LDX    #$0003           ;Initlize index register X
                  IDIV

; Delay is in index register X
                  DEX                     ; Subtract two from delay value
                  DEX
; Need to check the user supplied modulus value before sync of generators
; If modulus value is divisible by 3
; Test the remainder to see which delay sequence to use
                  CMPB   #$02             ; remainder = .6666667
                  BEQ    GEN_START_3_DELAY
                  CMPB   #$01             ; remainder = .3333337
                  BEQ    GEN_START_2_DELAY

; delay amount when remainder = 0
GEN_START_1_DELAY:

; Start Generator A
                  MOVB   #$82, PMFENCA    ; This sets PWMENA and LDOKA

; Delay for sync
                  TFR    X, D             ; get delay value back
                  DBNE   D,*              ; Loop for Delay
                  NOP

; Start generator B
                  MOVB   #$82, PMFENCB    ; This sets PWMENB and LDOKB

; Delay for sync
                  TFR    X, D             ; get delay value back
                  DBNE   D,*              ; Loop for Delay
                  NOP
```

```
        ; Now start generator C
                        MOVB    #$82, PMFENCC     ; This sets PWMENC and LDOKC
                        BRA                       RUN

        ; delay amount when remainder = 0.3333337
        GEN_START_2_DELAY:

        ; Start Generator A
                        MOVB    #$82, PMFENCA     ; This sets PWMENA and LDOKA

        ; Delay for sync
                        TFR     X, D              ; get delay value back
                        DBNE    D,*               ; Loop for Delay
                        NOP
                        NOP
        ; Start generator B
                        MOVB    #$82, PMFENCB     ; This sets PWMENB and LDOKB

        ; Delay for sync
                        TFR     X, D              ; get delay value back
                        DBNE    D,*               ; Loop for Delay
                        NOP
                        NOP
        ; Now start generator C
                        MOVB    #$82, PMFENCC     ; This sets PWMENC and LDOKC
                        BRA                       RUN

        ; delay amount when remainder = 0.6666667
        GEN_START_3_DELAY:

        ; Start Generator A
                        MOVB    #$82, PMFENCA     ; This sets PWMENA and LDOKA

        ; Delay for sync
                        TFR     X, D              ; get delay value back
                        DBNE    D,*               ; Loop for Delay
                        NOP
                        NOP
                        NOP
        ; Start generator B
                        MOVB    #$82, PMFENCB     ; This sets PWMENB and LDOKB

        ; Delay for sync
                        TFR     X, D              ; get delay value back
                        DBNE    D,*               ; Loop for Delay
                        NOP
                        NOP
                        NOP
        ; Now start generator C
                        MOVB    #$82, PMFENCC     ; This sets PWMENC and LDOKC

        ; Almost done!
        ; Turn on output of all generators
        RUN:            TFR     X, D              ; get delay value back
                        LDX                       #$0002; divide by 2
                        IDIV
                        DBNE    X,*               ; Loop for Delay
                        MOVB    #$00, PMFOUTC

}
return(1);
}
```

## Conclusion

The software detailed here implements the only known workaround for this issue. While the initialization of the PMF is quite intensive the overall application functionally remains very much the same as the hardware implementation.

*This page intentionally left blank.*

***How to Reach Us:***

**Home Page:**
www.freescale.com

**USA/Europe or Locations Not Listed:**
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
1-800-521-6274 or 480-768-2130

**Europe, Middle East, and Africa:**
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)

**Japan:**
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-0047, Japan
0120-191014 or +81-3-3440-3569

**Asia/Pacific:**
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
852-26668334

***For Literature Requests Only:***
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150



freescale™
semiconductor



EB636
Rev. 1
8/2004