


```

IN_TIMER1CH2 equ    0
IN_TIMER2CH0 equ    0
IN_TIMER2CH1 equ    0
IN_TIMER2CH2 equ    0
IN_TIMER2CH3 equ    0
IN_TIMER2CH4 equ    0

; Change the one of the following eight TIMERxCHx choices to 1 for the
; timer OUTPUT channel to be used by the program and the set rest to 0.
; Then save and reassemble the source.
OUT_TIMER1CH0 equ    0
OUT_TIMER1CH1 equ    1           ;User chooses timer 1 channel 1
OUT_TIMER1CH2 equ    0           ; as output in this case
OUT_TIMER2CH0 equ    0
OUT_TIMER2CH1 equ    0
OUT_TIMER2CH2 equ    0
OUT_TIMER2CH3 equ    0
OUT_TIMER2CH4 equ    0

; Input channel data:

        IF      IN_TIMER1CH0 ;if true, choose timer 1, channel 0
IN_TPMxSC equ    TPM1SC
IN_TPMxCnSC equ    TPM1C0SC
IN_TPMxCnVH equ    TPM1C0VH
        ENDIF

        IF      IN_TIMER1CH1 ;if true, choose timer 1, channel 1
IN_TPMxSC equ    TPM1SC
IN_TPMxCnSC equ    TPM1C1SC
IN_TPMxCnVH equ    TPM1C1VH
        ENDIF

        IF      IN_TIMER1CH2 ;if true, choose timer 1, channel 2
IN_TPMxSC equ    TPM1SC
IN_TPMxCnSC equ    TPM1C2SC
IN_TPMxCnVH equ    TPM1C2VH
        ENDIF

        IF      IN_TIMER2CH0 ;if true, choose timer 2, channel 0
IN_TPMxSC equ    TPM2SC
IN_TPMxCnSC equ    TPM2C0SC
IN_TPMxCnVH equ    TPM2C0VH
        ENDIF

        IF      IN_TIMER2CH1 ;if true, choose timer 2, channel 1
IN_TPMxSC equ    TPM2SC
IN_TPMxCnSC equ    TPM2C1SC
IN_TPMxCnVH equ    TPM2C1VH
        ENDIF

        IF      IN_TIMER2CH2 ;if true, choose timer 2, channel 2
IN_TPMxSC equ    TPM2SC
IN_TPMxCnSC equ    TPM2C2SC
IN_TPMxCnVH equ    TPM2C2VH
    
```

```

        ENDIF

        IF      IN_TIMER2CH3 ;if true, choose timer 2, channel 3
IN_TPMxSC    equ      TPM2SC
IN_TPMxCnSC  equ      TPM2C3SC
IN_TPMxCnVH  equ      TPM2C3VH
        ENDIF

        IF      IN_TIMER2CH4 ;if true, choose timer 2, channel 4
IN_TPMxSC    equ      TPM2SC
IN_TPMxCnSC  equ      TPM2C4SC
IN_TPMxCnVH  equ      TPM2C4VH
        ENDIF

; Output channel data:

        IF      OUT_TIMER1CH0 ;if true, choose timer 1, channel 0
OUT_TPMxSC   equ      TPM1SC
OUT_TPMxCnSC equ      TPM1C0SC
OUT_TPMxCnVH equ      TPM1C0VH
OUT_TPMxCnVL equ      TPM1C0VL
OUT_TPMxMODH equ      TPM1MODH
OUT_TPMxMODL equ      TPM1MODL
        ENDIF

        IF      OUT_TIMER1CH1 ;if true, choose timer 1, channel 1
OUT_TPMxSC   equ      TPM1SC
OUT_TPMxCnSC equ      TPM1C1SC
OUT_TPMxCnVH equ      TPM1C1VH
OUT_TPMxCnVL equ      TPM1C1VL
OUT_TPMxMODH equ      TPM1MODH
OUT_TPMxMODL equ      TPM1MODL
        ENDIF

        IF      OUT_TIMER1CH2 ;if true, choose timer 1, channel 2
OUT_TPMxSC   equ      TPM1SC
OUT_TPMxCnSC equ      TPM1C2SC
OUT_TPMxCnVH equ      TPM1C2VH
OUT_TPMxCnVL equ      TPM1C2VL
OUT_TPMxMODH equ      TPM1MODH
OUT_TPMxMODL equ      TPM1MODL
        ENDIF

        IF      OUT_TIMER2CH0 ;if true, choose timer 2, channel 0
OUT_TPMxSC   equ      TPM2SC
OUT_TPMxCnSC equ      TPM2C0SC
OUT_TPMxCnVH equ      TPM2C0VH
OUT_TPMxCnVL equ      TPM2C0VL
OUT_TPMxMODH equ      TPM2MODH
OUT_TPMxMODL equ      TPM2MODL
        ENDIF

        IF      OUT_TIMER2CH1 ;if true, choose timer 2, channel 1
OUT_TPMxSC   equ      TPM2SC
OUT_TPMxCnSC equ      TPM2C1SC

```

```

OUT_TPMxCnVH equ    TPM2C1VH
OUT_TPMxCnVL equ    TPM2C1VL
OUT_TPMxMODH equ    TPM2MODH
OUT_TPMxMODL equ    TPM2MODL
                ENDIF

                IF    OUT_TIMER2CH2 ;if true, choose timer 2, channel 2
OUT_TPMxSC    equ    TPM2SC
OUT_TPMxCnSC equ    TPM2C2SC
OUT_TPMxCnVH equ    TPM2C2VH
OUT_TPMxCnVL equ    TPM2C2VL
OUT_TPMxMODH equ    TPM2MODH
OUT_TPMxMODL equ    TPM2MODL
                ENDIF

                IF    OUT_TIMER2CH3 ;if true, choose timer 2, channel 3
OUT_TPMxSC    equ    TPM2SC
OUT_TPMxCnSC equ    TPM2C3SC
OUT_TPMxCnVH equ    TPM2C3VH
OUT_TPMxCnVL equ    TPM2C3VL
OUT_TPMxMODH equ    TPM2MODH
OUT_TPMxMODL equ    TPM2MODL
                ENDIF

                IF    OUT_TIMER2CH4 ;if true, choose timer 2, channel 4
OUT_TPMxSC    equ    TPM2SC
OUT_TPMxCnSC equ    TPM2C4SC
OUT_TPMxCnVH equ    TPM2C4VH
OUT_TPMxCnVL equ    TPM2C4VL
OUT_TPMxMODH equ    TPM2MODH
OUT_TPMxMODL equ    TPM2MODL
                ENDIF
    
```

Appendix B: HCS08 Assembly Code Example for RS-232

The following is the assembly code listing for trimming the internal oscillator to 243 kHz using a 9600-baud RS-232 terminal as the reference clock.

Metrowerks HC08-Assembler
(c) COPYRIGHT METROWERKS 1987-2003

Rel. Loc	Obj. code	Source line

1		*****
2		;* Filename: 9S08GB_GT_IRGTRIM_9600.asm Copyright (c) 2003
3		*****
4		*****
5		;* Oscillator Trim Routine for 9S08GB/GT Family Bill Lucas/Scott Pape
6		;*
7		;* Rev: 1.0 Date: 18July2002 Scott Pape

Calibrating the MC9S08GB/GT Internal Clock Generator (ICG)

**For More Information On This Product,
Go to: www.freescale.com**


```

114
115 00A0 C6 FFBE          lda    NVICGTRIM      ;Current stored Trim value.
116 00A3 43              coma                    ;$FF -> $00
117 00A4 26 6D          bne    Trim_error    ;If Trim <> $FF, Not erased skip cal
118
119 00A6 3F 87          clr    LoopCnt       ;Initialize the iteration counter
120 00A8 3F 84          TryAgain:  clr    ActualH
121 00AA 3F 85          clr    ActualL
122 00AC 3F 86          clr    Delta
123 00AE 3C 87          inc    LoopCnt       ;Increment iteration count
124
125 00B0 6E 08 30        mov    #(mCLKSA),IN_TPMxSC ;bus clk & div by 1
126 00B3 6E 04 35        mov    #mELSLA,IN_TPMxCnSC ;capture on rising edge only
127
128 00B6 B6 35          lda    IN_TPMxCnSC   ;dummy read of the status register
129 00B8 1F 35          bclr  CH0F,IN_TPMxCnSC ;complete clearing if it was set
130
131 00BA CD 0173        loop:   jsr    read_timer   ;get the first edge

```

Metrowerks HC08-Assembler
(c) COPYRIGHT METROWERKS 1987-2003

Rel. Loc	Obj. code	Source line			
----	-----	-----			
132	00BD 35 82		sthx	SyncOffsetH	;save for later
133	00BF CD 0173		jsr	read_timer	;get the second edge
134	00C2 9F		txa		;low byte of second edge time
135	00C3 B0 83		sub	SyncOffsetL	;compute the time difference
136	00C5 B7 85		sta	ActualL	;low byte of Actual
137	00C7 8B		pshh		;high byte of second edge time
138	00C8 86		pula		
139	00C9 B2 82		sbc	SyncOffsetH	
140	00CB B7 84		sta	ActualH	;high byte of Actual
141	00CD B6 85		lda	ActualL	;Adjust the Actual value so...
142	00CF A0 00		sub	#Adjust	...the range fits into 1 byte
143	00D1 B7 85		sta	ActualL	
144	00D3 A1 5B		cmp	#ExpectedL	;Compare Actual with Expected value
145	00D5 27 55		beq	GotTrim	;If equal, osc is trimmed!
146	00D7 25 1D		blo	TooSlow	;If lower, osc is too slow
147	00D9 A0 5B		sub	#ExpectedL	;Otherwise, osc is too fast
148	00DB B7 86		sta	Delta	;Calculate Delta from Act-Exp
149	00DD B6 4E		lda	ICGTRM	
150	00DF A1 FF		cmp	#\$FF	;Check if trim=\$FF
151	00E1 27 49		beq	GotTrim	;if true, can't slow more
152	00E3 0A 87 46		brset	5,LoopCnt,GotTrim	;bit5=32 loops=limit
153	00E6 09 87 03		brclr	4,LoopCnt,AddDelta	;bit4 = 16 loops
154	00E9 6E 01 86		mov	#\$01,Delta	;after 8 loops, just add 1
155	00EC BB 86	AddDelta:	add	Delta	;acca still has OSCTRIM
156	00EE 24 02		bcc	AddTrim	;did carry bit get set?
157	00F0 A6 FF		lda	#\$FF	;if carry set, max osctrim
158	00F2 B7 4E	AddTrim:	sta	ICGTRM	;Trim increases to slow freq
159	00F4 20 B2		bra	TryAgain	;Try new value
160	00F6 A6 5B	TooSlow:	lda	#ExpectedL	
161	00F8 B0 85		sub	ActualL	

```

162 00FA B7 86          sta    Delta          ;Calculate Delta from Exp-Act
163 00FC B6 4E          lda    ICGTRM
164 00FE 27 2C          beq    GotTrim        ;if true, can't speed-up more
165 0100 0A 87 29       brset  5,LoopCnt,GotTrim ;bit5=32 loops=limit
166 0103 09 87 03       brclr  4,LoopCnt,SubDelta ;bit4 = 16 loops
167 0106 6E 01 86       mov    #$01,Delta     ;after 8 loops, just add 1
168 0109 B0 86          SubDelta: sub    Delta
169 010B 24 02          bcc    SubTrim        ;did carry bit get set?
170 010D A6 FF          lda    #$FF           ;if carry set, min osctrim
171 010F B7 4E          SubTrim: sta    ICGTRM
172 0111 20 95          bra    TryAgain
173
174                    ;***** Toggle PTD1 @ 12.4 kHz to 20.8 kHz if we have an error *****
175                    ;* There are two reasons to be here:
176                    ;* 1). NVICGTRIM is not errased
177                    ;* 2). There was a Flash error during prog'ing. This error is rare.
178
179                    ; The following timer code will output 1% of the CPU BUSCLOCK to
180                    ; the selected output compare channel for ICG frequency monitoting
181
182 0113 C6 FFBE        Trim_error: lda    NVICGTRIM    ;Current Trim value.
183 0116 B7 4E          sta    ICGTRM         ;Use the value that's there
184 0118 6E 08 30       mov    #mCLKSA,OUT_TPMxSC ;Use BUSCLK as timer clock
185 011B 6E 14 38       mov    #(mMS1A|mELS1A),OUT_TPMxCnSC ;Toggle on output cmp
186 011E 6E 00 33       mov    #zero,OUT_TPMxMODH
187 0121 6E 31 34       mov    #forty_nine,OUT_TPMxMODL ;Timer modulo reg = 49
188 0124 6E 00 39       mov    #zero,OUT_TPMxCnVH
189 0127 6E 01 3A       mov    #one,OUT_TPMxCnVL ;Timer output compare at 1
190 012A 20 FE          bra    $              ;Let the output TPM pin toggle
191                    ;at 1% of BUSCLOCK to indicate error
192
193                    ;** Program Trim Value to Flash *****
194
195 012C AD 16          GotTrim:  bsr    PrgTrim        ;Program the Trim value
    
```

Metrowerks HC08-Assembler
(c) COPYRIGHT METROWERKS 1987-2003

Rel.	Loc	Obj.	code	Source line
196	012E	C6	FFBE	lda NVICGTRIM ;Read value from Flash to make sure
197	0131	B7	4E	sta ICGTRM ;Verf trim value is prog'ed in flash
198				
199				; The following timer code will output 10% of the CPU BUSCLOCK to
200				; the selected output compare channel for ICG frequency monitoting
201				
202	0133	6E	14 38	mov #(mMS1A mELS1A),OUT_TPMxCnSC ;Toggle on output cmp
203	0136	6E	00 33	mov #zero,OUT_TPMxMODH
204	0139	6E	04 34	mov #four,OUT_TPMxMODL ;Timer modulo register = 4
205	013C	6E	00 39	mov #zero,OUT_TPMxCnVH
206	013F	6E	01 3A	mov #one,OUT_TPMxCnVL ;Timer output compare at 1
207				
208	0142	20	FE	bra \$;Let selected output TPM pin toggle
209				; at 10% of BUSCLOCK to indicate completion

```

210
211
212      ;*** Subroutines ProgTrim *****
213      ;* Changed for GB/GT Flash WLL
214      ; PrgTrim:
215      ; Programs the value in the ICGTRM register into the reserved Flash
216      ; location for the trim value, NVICGTRIM @ $FFBE.
217      ; Entry Conditions:
218      ; 1. The NVICGTRIM ($FFBE) location in flash is erased ($FF).
219      ; 2. The ICGTRM register contains the desired trim value.
220      ;
221      ; Exit conditions:
222      ; 1. NVICGTRIM is programmed with the value in ICGTRM.
223      ; 2. ICGTRM is not modified.
224      ; 3. no registers altered
225      ;
226
227 0144 87      PrgTrim:      psha
228 0145 A6 30      lda      #mFPVIOL|mFACCERR ;error flag bits to clear if set
229 0147 C7 1825     sta      FSTAT      ;clear any error flags
230 014A A6 08      lda      #mDIV3      ;1.666 MHz/8+1 = 185Khz Flash clock
231 014C C7 1820     sta      FCDIV      ;divide by 8+1 for 1.66 MHz bus.
232      ;
233      ; This is normally done at start-up, however we
234      ; will initialize here to only program one byte
234 014F B6 4E      lda      ICGTRM      ;trim value
235 0151 C7 FFBE     sta      NVICGTRIM   ;place in Flash
236 0154 A6 20      lda      #byte_pgm   ;page write command
237 0156 C7 1826     sta      FCMD        ;initiate the programming sequence
238 0159 C6 1825     lda      FSTAT
239 015C AA 80      ora      #mFCBEF
240 015E C7 1825     sta      FSTAT      ;launch the command
241 0161 21 FE      brn      *          ;burn bus cycs before chking status
242 0163 C6 1825     lda      FSTAT      ;look at status register for errors
243 0166 A4 30      and      #mFPVIOL|mFACCERR ;error flag bits
244 0168 26 A9      bne      Trim_error ;check for programming errors
245 016A C6 1825     pgm_loop:  lda      FSTAT      ;wait around until prog'ing complete
246 016D A4 40      and      #mFCCF     ;command complete yet?
247 016F 27 F9      beq      pgm_loop   ;not done yet..loop until complete
248 0171 86      pula
249 0172 81      EndPrgTrim: rts      ;done
250
251      ;*** Subroutine read_timer *****
252      ;* Changed for TB/GT device WLL
253      ; read_timer:
254      ; Waits for a reference clock positive going edge, reads the captured
255      ; value into H:X, clears the input capture flag and returns the timer
256      ; capture value in register H:X.
257      ; Entry Conditions:
258      ; None
259      ;

```

Metrowerks HC08-Assembler
(c) COPYRIGHT METROWERKS 1987-2003


```

77
78 0082          SyncOffsetH: ds      1          ;Timer offset hi byte
79 0083          SyncOffsetL: ds      1          ;Timer offset lo byte
80 0084          ActualH:      ds      1          ;Timer count hi byte
81 0085          ActualL:      ds      1          ;Timer count lo byte
82 0086          Delta:        ds      1          ;Actual delta from expected
83 0087          LoopCnt:      ds      1          ;Count how many measurements taken
84
85      0000 001F  Adjust:      equ     $1F          ;Adjust offset to get val $100-$1FF
86      0000 0001  ExpectedH:  equ     $1          ;range is $145->21E, ideal $1B2
87      0000 0093  ExpectedL:  equ     $B2-Adjust ;Expected Timer value if trimmed
88      0000 087F  stack:      equ     $87F          ;This will work for GB/GT 32 or 60
89      0000 0020  byte_pgm:  equ     $20          ;Command to program a single byte
90      0000 0000  zero:        equ     0          ;Constant 0
91      0000 0001  one:         equ     1          ;Constant 1
92      0000 0004  four:        equ     4          ;Constant 4
93      0000 0031  forty_nine equ     49          ;Constant 49
94
95      ;***** include "9S08GB_GT_Timer_Selection.inc" ;IC/OC timer *****
99
100     ;** Main *****
101
102 0088 45 087F  Start:      ldhx   #stack
103 008B 94          txs           ;SP--(H:X)
104 008C C6 1802  lda     SOPT
105 008F A4 42          and     #(mCOPT|mBKGDPE)
106 0091 C7 1802  sta     SOPT          ;disable the COP
107
108 0094 6E 08 30  mov     #mCLKSA,OUT_TPMxSC ;Use BUSCLK as timer clock
109
110 0097 6E 28 48  mov     #(mREFS|mCLKS0),ICGC1 ;xtal and FLL internal
111 009A 6E 12 49  mov     #(mMFD0|mRFD1),ICGC2 ;multiply by 6 and div by 4
112 009D 07 4A FD  lock_loop: brclr  LOCK,ICGS1,lock_loop ;wait for FLL lock @ 1.6 MHz
113
114 00A0 C6 FFBE  lda     NVICGTRIM      ;Current stored Trim value.
115 00A3 43          coma          ;$FF -> $00
116 00A4 26 6D          bne     Trim_error    ;If Trim <> $FF, Not erased skip cal
117
118 00A6 3F 87          clr     LoopCnt       ;Initialize the iteration counter
119 00A8 3F 84          TryAgain: clr     ActualH
120 00AA 3F 85          clr     ActualL
121 00AC 3F 86          clr     Delta
122 00AE 3C 87          inc     LoopCnt       ;Increment iteration count
123
124 00B0 6E 0E 30  mov     #(mCLKSA|mPS2|mPS1),IN_TPMxSC ;bus clk & div by 64
125 00B3 6E 04 35  mov     #mELS1A,IN_TPMxCnSC ;capture on rising edge only
126
127 00B6 B6 35          lda     IN_TPMxCnSC   ;dummy read of the status register
128 00B8 1F 35          bclr   CH0F,IN_TPMxCnSC ;complete clearing if it was set
129
130 00BA CD 0173  loop:   jsr     read_timer    ;get the first edge
131 00BD 35 82          sthx   SyncOffsetH   ;save for later
    
```

Metrowerks HC08-Assembler

Calibrating the MC9S08GB/GT Internal Clock Generator (ICG)

**For More Information On This Product,
Go to: www.freescale.com**

(c) COPYRIGHT METROWERKS 1987-2003

Rel. Loc	Obj. code	Source line
132	00BF CD 0173	jsr read_timer ;get the second edge
133	00C2 9F	txa ;low byte of second edge time
134	00C3 B0 83	sub SyncOffsetL ;compute the time difference
135	00C5 B7 85	sta ActualL ;low byte of Actual
136	00C7 8B	pshh ;high byte of second edge time
137	00C8 86	pula
138	00C9 B2 82	sbc SyncOffsetH
139	00CB B7 84	sta ActualH ;high byte of Actual
140	00CD B6 85	lda ActualL ;Adjust the Actual value so...
141	00CF A0 1F	sub #Adjust ;...the range fits into 1 byte
142	00D1 B7 85	sta ActualL
143	00D3 A1 93	cmp #ExpectedL ;Compare Actual with Expected value
144	00D5 27 52	beq GotTrim ;If equal, osc is trimmed!
145	00D7 25 1D	blo TooSlow ;If lower, osc is too slow
146	00D9 A0 93	sub #ExpectedL ;Otherwise, osc is too fast
147	00DB B7 86	sta Delta ;Calculate Delta from Act-Exp
148	00DD B6 4E	lda ICGTRM
149	00DF A1 FF	cmp #\$FF ;Check if trim=\$FF
150	00E1 27 46	beq GotTrim ;if true, can't slow more
151	00E3 0A 87 43	brset 5,LoopCnt,GotTrim ;bit5=32 loops=limit
152	00E6 09 87 03	brclr 4,LoopCnt,AddDelta ;bit4 = 16 loops
153	00E9 6E 01 86	mov #\$01,Delta ;after 8 loops, just add 1
154	00EC BB 86	AddDelta: add Delta ;acca still has OSCTRIM
155	00EE 24 02	bcc AddTrim ;did carry bit get set?
156	00F0 A6 FF	lda #\$FF ;if carry set, max osctrim
157	00F2 B7 4E	AddTrim: sta ICGTRM ;Trim increases to slow freq
158	00F4 20 B2	bra TryAgain ;Try new value
159	00F6 A6 93	TooSlow: lda #ExpectedL
160	00F8 B0 85	sub ActualL
161	00FA B7 86	sta Delta ;Calculate Delta from Exp-Act
162	00FC B6 4E	lda ICGTRM
163	00FE 27 29	beq GotTrim ;if true, can't speed-up more
164	0100 0A 87 26	brset 5,LoopCnt,GotTrim ;bit5=32 loops=limit
165	0103 09 87 03	brclr 4,LoopCnt,SubDelta ;bit4 = 16 loops
166	0106 6E 01 86	mov #\$01,Delta ;after 8 loops, just add 1
167	0109 B0 86	SubDelta: sub Delta
168	010B 24 02	bcc SubTrim ;did carry bit get set?
169	010D A6 FF	lda #\$FF ;if carry set, min osctrim
170	010F B7 4E	SubTrim: sta ICGTRM
171	0111 20 95	bra TryAgain
172		
173		***** Toggle PTD1 @ 12.4 kHz to 20.8 kHz if we have an error *****
174		* There are two reasons to be here:
175		* 1). NVICGTRIM is not errased
176		* 2). There was a Flash error during prog'ing. This error is rare.
177		
178		; The following timer code will output 1% of the CPU BUSCLOCK to
179		; the selected output compare channel for ICG frequency monitoting
180		
181	0113 C6 FFBE	Trim_error: lda NVICGTRIM ;Current Trim value.
182	0116 B7 4E	sta ICGTRM ;Use the value that's there

```

183 0118 6E 14 38      mov     #(mMS1A|mELS1A),OUT_TPMxCnSC ;Toggle on output cmp
184 011B 6E 00 33      mov     #zero,OUT_TPMxMODH
185 011E 6E 31 34      mov     #forty_nine,OUT_TPMxMODL ;Timer modulo reg = 49
186 0121 6E 00 39      mov     #zero,OUT_TPMxCnVH
187 0124 6E 01 3A      mov     #one,OUT_TPMxCnVL ;Timer output compare at 1
188 0127 20 FE          bra     $                ;Let the output TPM pin toggle
189                                ;at 1% of BUSCLOCK to indicate error
190
191                                ;*** Program Trim Value to Flash *****
192
193 0129 AD 19      GotTrim:  bsr     PrgTrim        ;Program the Trim value
194 012B C6 FFBE    lda     NVICGTRIM      ;Read value from Flash to make sure
195 012E B7 4E      sta     ICGTRM        ;Verf trim value is prog'ed in flash
    
```

Metrowerks HC08-Assembler
(c) COPYRIGHT METROWERKS 1987-2003

Rel. Loc	Obj. code	Source line
196		
197		; The following timer code will output 10% of the CPU BUSCLOCK to
198		; the selected output compare channel for ICG frequency monitoting
199		
200	0130 6E 08 30	mov #(mCLKSA),IN_TPMxSC ;bus clk
201	0133 6E 14 38	mov #(mMS1A mELS1A),OUT_TPMxCnSC ;Toggle on output cmp
202	0136 6E 00 33	mov #zero,OUT_TPMxMODH
203	0139 6E 04 34	mov #four,OUT_TPMxMODL ;Timer modulo register = 4
204	013C 6E 00 39	mov #zero,OUT_TPMxCnVH
205	013F 6E 01 3A	mov #one,OUT_TPMxCnVL ;Timer output compare at 1
206		
207	0142 20 FE	bra \$;Let selected output TPM pin toggle
208		; at 10% of BUSCLOCK to indicate completion
209		
210		
211		;*** Subroutines ProgTrim *****
212		* Changed for GB/GT Flash WLL
213		; PrgTrim:
214		; Programs the value in the ICGTRM register into the reserved Flash
215		; location for the trim value, NVICGTRIM @ \$FFBE.
216		; Entry Conditions:
217		; 1. The NVICGTRIM (\$FFBE) location in flash is erased (\$FF).
218		; 2. The ICGTRM register contains the desired trim value.
219		;
220		; Exit conditions:
221		; 1. NVICGTRIM is programmed with the value in ICGTRM.
222		; 2. ICGTRM is not modified.
223		; 3. no registers altered
224		;
225		
226	0144 87	PrgTrim: psha
227	0145 A6 30	lda #mFPVIOL mFACCERR ;error flag bits to clear if set
228	0147 C7 1825	sta FSTAT ;clear any error flags
229	014A A6 08	lda #mDIV3 ;1.666 MHz/8+1 = 185Khz Flash clock
230	014C C7 1820	sta FCDIV ;divide by 8+1 for 1.66 MHz bus.

```

231          ;                               This is normally done at start-up, however we
232          ;                               will initialize here to only program one byte
233 014F B6 4E          lda    ICGTRM          ;trim value
234 0151 C7 FFBE       sta    NVICGTRIM       ;place in Flash
235 0154 A6 20          lda    #byte_pgm      ;page write command
236 0156 C7 1826       sta    FCMD          ;initiate the programming sequence
237 0159 C6 1825       lda    FSTAT
238 015C AA 80          ora    #mFCBEF
239 015E C7 1825       sta    FSTAT          ;launch the command
240 0161 21 FE         brn    *                ;burn bus cycs before chking status
241 0163 C6 1825       lda    FSTAT          ;look at status register for errors
242 0166 A4 30          and    #mFPVIOL|mFACCERR ;error flag bits
243 0168 26 A9         bne    Trim_error     ;check for programming errors
244 016A C6 1825       pgm_loop:  lda    FSTAT          ;wait around until prog'ing complete
245 016D A4 40          and    #mFCCF         ;command complete yet?
246 016F 27 F9         beq    pgm_loop       ;not done yet..loop until complete
247 0171 86           pula
248 0172 81           EndPrgTrim: rts          ;done
249
250          ;*** Subroutine read_timer *****
251          ;* Changed for TB/GT device WLL
252          ; read_timer:
253          ; Waits for a reference clock positive going edge, reads the captured
254          ; value into H:X, clears the input capture flag and returns the timer
255          ; capture value in register H:X.
256          ; Entry Conditions:
257          ; None
258          ;
259          ; Exit conditions:

```

Metrowerks HC08-Assembler
(c) COPYRIGHT METROWERKS 1987-2003

Rel. Loc	Obj. code	Source line
-----	-----	-----
260		; Register A destroyed, register X contains the captured timer value
261		;
262		
263	0173 0F 35 FD	read_timer: brclr CH0F,IN_TPMxCnSC,read_timer ;loop until see edge
264	0176 55 36	ldhx IN_TPMxCnVH ;get the counter value
265	0178 B6 35	lda IN_TPMxCnSC ;dummy read of the status register
266	017A 1F 35	bclr CH0F,IN_TPMxCnSC ;complete the clearing process
267	017C 81	rts
268		

Freescale Semiconductor, Inc.

How to Reach Us:

Home Page:

www.freescale.com

E-mail:

support@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 +1-480-768-2130
support@freescale.com

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)
support@freescale.com

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

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

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
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor 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 and all liability, including without limitation consequential or incidental damages. "Typical" parameters which 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 the 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 and hold 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 claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

AN2496/D
6/2003



**For More Information On This Product,
Go to: www.freescale.com**