CHAPTER 4

Timer and Interrupt Operations

Example 2

Write a program that generates a 1 kHz square-wave output at pin T1PWM/T1CMP/IOPB4.

Solution:

Since the output is required at the T1PWM/T1CMP/IOPB4 pin, we have to use the GP timer1. There are many ways of generating a square wave. In this example, we will use the continuous up counting mode of the timer. The period register is loaded with the appropriate value. The compare register is loaded with a count corresponding to half the period - so as to get a square wave.

Registers involved:

T1PR = 7500

The calculation of the value to be loaded in T1PR is as follows-

Period Value = CPUCLK / PRESCALER / DESIRED FREQ = 30×10^6 / 4 / 1000 = 7500

CPUCLK = 30MHz.

PRESCALER = 4. This is set by bits 10-8 of T1CON.

T1CMPR = 3750

GPTCONA = 004Ah

Enable Compare outputs of all GPTs. GPT1 compare output - active high

T1CON = 1242h

Select Continuous-Up counting mode, set input pre-scaler to 4, select internal clock, enable timer operation, program the counter to stop immediately on emulation suspend.

MCRA = 3000h

Configure the o/p pin T1PWM/T1CMP/IOPB4 pin for the T1PWM function.

Following is the complete code-

```
;*********************************************************************
; File Name:       ch4_e2.asm
; Target System: C240x Evaluation Board
; Description:     This program generates a 1 kHz square-wave output at
;                       pin T1PWM/T1CMP/IOPB4 to be observed on   an
oscilloscope
;*********************************************************************

;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Global symbol declarations
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        .def _c_int0,PHANTOM,GISR1,GISR2,GISR3,GISR4,GISR5,GISR6
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Address definitions
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        .include f2407.h
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Uninitialized global variable definitions
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        .bss    GPR0,1              ;general purpose variable
;======================================================================
; M A I N   C O D E  - starts here
;======================================================================
            .text
_c_int0:
            NOP
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Configure the System Control and Status Registers
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #DP_PF1         ;set data page
        SPLK    #0000000011111101b, SCSR1
*                ||||||||||||||||
*                FEDCBA9876543210
* bit 15        0:      reserved
* bit 14        0:      CLKOUT = CPUCLK
* bit 13-12     00:     IDLE1 selected for low-power mode
* bit 11-9      000:    PLL x4 mode
* bit 8         0:      reserved
* bit 7         1:      1 = enable ADC module clock
* bit 6         1:      1 = enable SCI module clock
* bit 5         1:      1 = enable SPI module clock
* bit 4         1:      1 = enable CAN module clock
* bit 3         1:      1 = enable EVB module clock
* bit 2         1:      1 = enable EVA module clock
* bit 1         0:      reserved
* bit 0         1:      clear the ILLADR bit
        LACC    SCSR2                   ;ACC = SCSR2 register
        OR      #0000000000001011b      ;OR in bits to be set
        AND     #0000000000001111b      ;AND out bits to be cleared
*                ||||||||||||||||
*                FEDCBA9876543210
* bit 15-6      0's:    reserved
* bit 5         0:      do NOT clear the WD OVERRIDE bit
* bit 4         0:      XMIF_HI-Z, 0=normal mode, 1=Hi-Z'd
* bit 3         1:      disable the boot ROM, enable the FLASH
* bit 2     no change   MP/MC* bit reflects the state of the MP/MC* pin
* bit 1-0      11:      11 = SARAM mapped to prog and data (default)

        SACL    SCSR2                   ;store to SCSR2 register
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup the core interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #0h                   ;set data page
```

```
        SPLK    #0h,IMR                 ;clear the IMR register
        SPLK    #111111b,IFR            ;clear any pending core interrupts
        SPLK    #000000b,IMR            ;enable desired core interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup the event manager interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #DP_EVA                  ;set data page
        SPLK    #0FFFFh, EVAIFRA         ;clear all EVA group A interrupts
        SPLK    #0FFFFh, EVAIFRB         ;clear all EVA group B interrupts
        SPLK    #0FFFFh, EVAIFRC         ;clear all EVA group C interrupts
        SPLK    #00000h, EVAIMRA ;enabled desired EVA group A interrupts
        SPLK    #00001h, EVAIMRB ;enabled desired EVA group B interrupts
        SPLK    #00000h, EVAIMRC ;enabled desired EVA group C interrupts

        LDP     #DP_EVB                  ;set data page
        SPLK    #0FFFFh, EVBIFRA         ;clear all EVB group A interrupts
        SPLK    #0FFFFh, EVBIFRB         ;clear all EVB group B interrupts
        SPLK    #0FFFFh, EVBIFRC         ;clear all EVB group C interrupts
        SPLK    #00000h, EVBIMRA ;enabled desired EVB group A interrupts
        SPLK    #00000h, EVBIMRB ;enabled desired EVB group B interrupts
        SPLK    #00000h, EVBIMRC ;enabled desired EVB group C interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Enable global interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        CLRC    INTM                      ;enable global interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Disable the watchdog timer
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #DP_PF1                   ;set data page

        SPLK    #0000000011101000b, WDCR
*                ||||||||||||||||
*                FEDCBA9876543210
* bits 15-8     0's     reserved
* bit 7         1:      clear WD flag
* bit 6         1:      disable the dog
* bit 5-3       101:    must be written as 101
* bit 2-0       000:    WDCLK divider = 1
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup external memory interface for LF2407 EVM
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #GPR0                 ;set current data page to
                                      ;the data page of variable GPR0
        SPLK    #0000000001000000b, GPR0
*                ||||||||||||||||
*                FEDCBA9876543210
* bit 15-11     0's:    reserved
* bit 10-9      00:     bus visibility off
* bit 8-6       001:    1 wait-state for I/O space
* bit 5-3       000:    0 wait-state for data space
* bit 2-0       000:    0 wait state for program space
        OUT     GPR0, WSGR
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup shared I/O pins
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #DP_PF2                   ;set data page

        SPLK    #0011000000000000b,MCRA ;set TxPWM pins
*                ||||||||||||||||
*                FEDCBA9876543210
* bit 15        0:      0=IOPB7,     1=TCLKINA
* bit 14        0:      0=IOPB6,     1=TDIRA
* bit 13        1:      0=IOPB5,     1=T2PWM/T2CMP
* bit 12        1:      0=IOPB4,     1=T1PWM/T1CMP
```

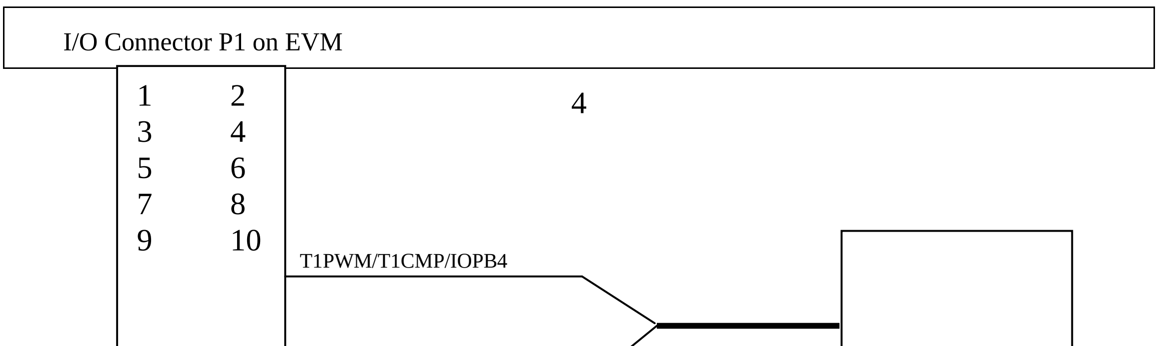
```
* bit 11        0:      0=IOPB3,     1=PWM6
* bit 10        0:      0=IOPB2,     1=PWM5
* bit 9         0:      0=IOPB1,     1=PWM4
* bit 8         0:      0=IOPB0,     1=PWM3
* bit 7         0:      0=IOPA7,     1=PWM2
* bit 6         0:      0=IOPA6,     1=PWM1
* bit 5         0:      0=IOPA5,     1=CAP3
* bit 4         0:      0=IOPA4,     1=CAP2/QEP2
* bit 3         0:      0=IOPA3,     1=CAP1/QEP1
* bit 2         0:      0=IOPA2,     1=XINT1
* bit 1         0:      0=IOPA1,     1=SCIRXD
* bit 0         0:      0=IOPA0,     1=SCITXD
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup timers 1 and 2, and the PWM configuration
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #DP_EVA                 ;set data page
        SPLK    #0000h, T1CON           ;disable timer 1
        SPLK    #0000h, T2CON           ;disable timer 2

        SPLK    #0000000001001010b, GPTCONA
*                ||||||||||||||||
*                FEDCBA9876543210
* bit 15        0:      reserved
* bit 14        0:      T2STAT, read-only
* bit 13        0:      T1STAT, read-only
* bit 12-11     00:     reserved
* bit 10-9      00:     T2TOADC, 00 = no timer2 event starts ADC
* bit 8-7       00:     T1TOADC, 00 = no timer1 event starts ADC
* bit 6         1:      TCOMPOE, 0 = Hi-z all timer compare outputs
* bit 5-4       00:     reserved
* bit 3-2       10:     T2PIN, 10 = active high
* bit 1-0       10:     T1PIN, 10 = active high

            SPLK  #7500,T1PR         ;Load period register
            SPLK  #3750,T1CMPR       ;Load count in compare register
            SPLK  #0h, T1CNT         ;Set initial count=0
            SPLK  #1242h, T1CON      ;Select Continuous-Up counting mode
                                     ;Set input pre-scaler to 4
                                     ;Select internal clock
                                     ;Enable Timer Operation
                                     ;Program the counter to stop
                                     ;immediately on emulation suspend
END         B     END

;======================================================================
; I S R  -  PHANTOM
;
; Description:    Dummy ISR, used to trap spurious interrupts.
;
; Modifies: Nothing
;======================================================================
PHANTOM     B     PHANTOM
GISR1       RET
GISR2       RET
GISR3       RET
GISR4       RET
GISR5       RET
GISR6       RET
```

To check the square wave output on the oscilloscope, the following connections should be made-



Generating Pulse-Width Modulation outputs

A pulse width modulation signal is a fixed frequency on-off signal with variable duty cycle. This signal plays important role in electromechanical system, especially for electric machine drives. Recall that the duty cycle is defined as the percentage of time the signal is high compared to the signal’s period. Figure 4.14 illustrate PWM signals with different duty cycle. Note that a square wave is a special case of PWM signal with 50% duty cycle.

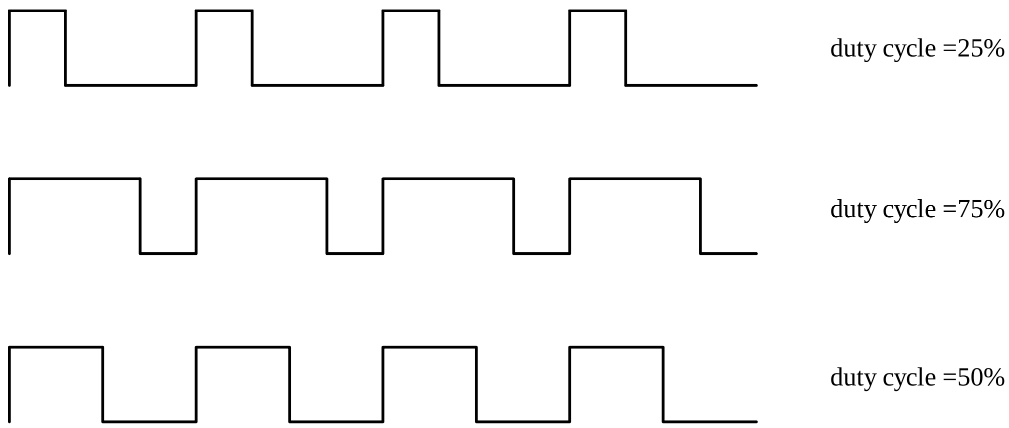


Figure 4.14 *PWM signals*

The TMS320F2407 offers a number of ways for generation of the PWM wave.

1. Using the general-purpose timers.
2. Using the simple compare unit.
3. Using the full compare unit.

Example 3

Write a program that generates an asymmetric PWM waveform of frequency 1 kHz at 25% duty cycle.

Period Value = **CPUCLK / PRESCALER / DESIRED FREQ** = 30×10^{7} / 1 / 1000= 30000

The main part of the code is listed below:

```
per_val           .set  30000
cmpr_val          .set  7500
        SPLK    #0000000001000101b, GPTCONA
*                |||||||||||||||
*                FEDCBA9876543210
* bit 15        0:      reserved
* bit 14        0:      T2STAT, read-only
* bit 13        0:      T1STAT, read-only
* bit 12-11     00:     reserved
* bit 10-9      00:     T2TOADC, 00 = no timer2 event starts ADC
* bit 8-7       00:     T1TOADC, 00 = no timer1 event starts ADC
* bit 6         1:      TCOMPOE, 0 = Hi-z all timer compare outputs
* bit 5-4       00:     reserved
* bit 3-2       01:     T2PIN, 01 = active low
* bit 1-0       01:     T1PIN, 01 = active low

            SPLK  #per_val,T1PR     ;Load period register
            SPLK  #cmpr_val,T1CMPR  ;Load count in compare register
            SPLK  #0h, T1CNT        ;Set initial count=0
            SPLK  #1042h, T1CON     ;Select Continuous-Up counting mode
                                    ;Set input pre-scaler to 1
                                    ;Select internal clock
                                    ;Enable Timer Operation
                                    ;Program the counter to stop
```

```
                                                ;immediately on emulation suspend

END          B      END
```

Measuring Period (Frequency, Speed)

The period of a repetitive signal includes both the high and low parts of the cycle. To measure period, a program needs to capture the time of two successive rising (or falling) edges. Below is an example program that measures the frequency of a signal. The basic scheme is to measure the number of rising edges encountered in one second. The rising edges are captured by the CAP1 input. The capture interrupt is enabled. Thus, the processor is interrupted at every rising edge. The ISR for the capture interrupt increments its counter CNT thus keeping track of the number of rising edges. The Timer1 is set for a period of 1 second and the period interrupt is enabled. This ISR stores the number of rising edges in "FREQ" and resets the counter for rising edges CNT.

Example 4

Write a program that measures the frequency of a square wave signal at CAP1 pin.

Solution:

Count the number of input pulses in 0.25 sec. and multiply it by 4, which is the number of pulses in 1 sec. This is right the value of frequency in Hertz.

Registers involved:

T1PR = 58594 ≈ E4E2h for 0.25 second.

The calculation of the value to be loaded in T1PR is as follows-

Period Value = CPUCLK / PRESCALER / DESIRED FREQ = 30×10^7 / 128 / 4 = 58594.75

CPUCLK = 30MHz.

PRESCALER = 128. This is set by bits 10-8 of T1CON.

GPTCONA = 0045h

Enable Compare outputs of all GPTs. GPT1 compare output - active high

T1CON = 1746h

Select continuous up counting mode, Prescaler = 128, enable timer compare operation, enable timer operation.

MCRA = 3038h

Configure pin TxPWM/TxCMP (x=1, 2) and CAP1/QEP1/IOPA3 to be primary function

CAPCONA = A040h

Bits 14-13: 01 - Enable capture units 1 and 2.

Bit 9: 0 - Select GP Timer 2 as time base for capture unit 1.

Bits 7-6: 01- Detect rising edge on capture unit 1.

CAPFIFOA = 0h

This clears the capture unit FIFO initially.

EVAIMRA = 0080h

Enable timer1 period interrupt

EVAIMRC = 1

Enable capture unit 1 interrupt

The main part of the code is as follows:

```
            .bss  CNT,1       ;Counter for the rising edges
            .bss  FREQ,1            ;Measurement result

;=====================================================================
; M A I N   C O D E  - starts here
;=====================================================================
            .text
_c_int0:
            NOP
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup the core interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #0h                     ;set data page
        SPLK    #0h,IMR                 ;clear the IMR register
        SPLK      #111111b,IFR                  ;clear  any  pending  core
interrupts
        SPLK    #001010b,IMR            ;enable INT2, INT4 interrupts

;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup shared I/O pins
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        LDP     #DP_PF2                 ;set data page
        SPLK    #0011000000111000b,MCRA ;set TxPWM pins
*                |||||||||||||||
*               FEDCBA9876543210
* bit 15        0:      0=IOPB7,     1=TCLKINA
* bit 14        0:      0=IOPB6,     1=TDIRA
* bit 13        1:      0=IOPB5,     1=T2PWM/T2CMP
* bit 12        1:      0=IOPB4,     1=T1PWM/T1CMP
```

```
* bit 11        0:      0=IOPB3,     1=PWM6
* bit 10        0:      0=IOPB2,     1=PWM5
* bit 9         0:      0=IOPB1,     1=PWM4
* bit 8         0:      0=IOPB0,     1=PWM3
* bit 7         0:      0=IOPA7,     1=PWM2
* bit 6         0:      0=IOPA6,     1=PWM1
* bit 5         1:      0=IOPA5,     1=CAP3
* bit 4         1:      0=IOPA4,     1=CAP2/QEP2
* bit 3         1:      0=IOPA3,     1=CAP1/QEP1
* bit 2         0:      0=IOPA2,     1=XINT1
* bit 1         0:      0=IOPA1,     1=SCIRXD
* bit 0         0:      0=IOPA0,     1=SCITXD

        LDP       #DP_EVA     ;set data page
      SPLK   #58594, T1PR     ;Period=0.25 second (E4E2h)
                              ;T1PR=30000000/128/4 for 30MHz DSP
      SPLK   #0045h, GPTCONA  ;Enable compare outputs of all
                              ;GPTs. GPT1 compare output "Active
                              ;High"
      SPLK   #0A040h, CAPCONA ;Enable capture units 1 and 2
                              ;Select GP Timer 2 as time base for
                              ;capture unit 1. This time base is not
                              ;used in this program. Detect rising edge
      SPLK   #0, CAPFIFOA     ;Clear the capture unit FIFO initially
      SPLK   #0080h, EVAIMRA  ;Enable timer1 period interrupt
      SPLK   #1, EVAIMRC      ;Enable capture unit 1 interrupt

      LDP    #CNT
      SPLK   #0, CNT          ;Initialize the edge counter

      LDP    #DP_EVA

      SPLK   #1746h, T1CON    ;ENABLE GPT1
                              ;Input clock prescalar 1/128
                              ;Enable timer operations

      CLRC   INTM             ;Enable maskable interrupts
WAIT  B      WAIT
;=====================================================================
; ISR - GPT1_ISR
; Description:    Store the number of rising edges of input signal
;           every 1 second. Resets the counter which counts
;           the number of rising edges.
; Modifies: FREQ, CNT
;=====================================================================
GISR2
      LDP   #CNT                    ;Set data page
      LT    CNT                     ;Load rising edge number into TREG
      MPY   #4                      ;Times 4 for 1 second
      SPL   FREQ                    ;Store the product as frequency
      SPLK  #0, CNT                 ;Reset rising edge counter
      LDP   #DP_EVA                 ;Set data page
      SPLK  #0FFFFh, EVAIFRA        ;clear all EVA group A interrupts
      CLRC  INTM                    ;Enable maskable interrupts
      RET                           ;Return from interruption
;=====================================================================
; ISR - CAP1_ISR
; Description:    Counts the number of rising edges of input signal
;
; Modifies: FREQ, CNT
```

```
;=====================================================================
GISR4
      LDP   #CNT
      LACC  CNT
      ADD   #1
      SACL  CNT                    ;Increment the edge counter by 1
      LDP   #DP_EVA
      SPLK  #0FFFFh, EVAIFRC       ;clear all EVA group C interrupts
      CLRC  INTM                   ;Enable maskable interrupts
      RET                          ;Return from interruption

;=====================================================================
; I S R  -  PHANTOM
;
; Description:    Dummy ISR, used to trap spurious interrupts.
;
; Modifies: Nothing
;=====================================================================
PHANTOM     B     PHANTOM
GISR1       RET
;GISR2            RET
GISR3       RET
;GISR4            RET
GISR5       RET
GISR6       RET
```

Example 5

Write a program that displays the number of seconds on the 4 LEDS on EVM.

Solution:

The GP timer1 is set to interrupt the processor every 1ms. The ISR has 2 counters, MSEC_CTR that counts the number of milliseconds and SEC_CTR that counts the number of seconds and outputs the number to the LEDs.

Registers involved:

T1PR = 7500

The calculation of the value to be loaded in T1PR is as follows-

Period Value = CPUCLK / PRESCALER / DESIRED FREQ = $30 \times 10^7 / 4 / 1000 = 7500$

CPUCLK = 30MHz.

PRESCALER = 4. This is set by bits 10-8 of T1CON.

GPTCONA = 0h

T1CON = 1244h

Set input clock prescaler = 4, disable timer compare operation, enable timer operation.

The main part of the code is as follows:

```
;-----------------------------------------------------------------------
; I/O Mapped EVM Register Declarations
;-----------------------------------------------------------------------
LEDS  .set  000Ch       ;LEDs Register

;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Uninitialized global variable definitions
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      .bss  GPR0,1      ;general purpose variable
      .bss  MSEC_CTR,1  ;Milli-second counter
      .bss  SEC_CTR,1   ;Second counter

;=======================================================================
; M A I N   C O D E  - starts here
;=======================================================================
      .text
_c_int0:
      NOP
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
;Setup the core interrupts
;~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      LDP   #0h                     ;set data page
      SPLK  #0h,IMR                 ;clear the IMR register
      SPLK  #111111b,IFR            ;clear any pending core interrupts
      SPLK  #000010b,IMR            ;enable INT2 interrupts



      LDP   #DP_EVA           ;Set data page
      SPLK  #0h,GPTCONA       ;GP timers are configured (no compare)
      SPLK  #0080h, EVAIMRA   ;Enable timer1 period interrupt
      SPLK  #7500,T1PR        ;Period = 1 ms
      SPLK  #0h,T1CNT         ;Initial value of the counter
      SPLK  #1244h,T1CON      ;Input clock prescaler = 4
                              ;Disable timer compare operation
                              ;Enable timer operations
      CLRC   INTM             ;Enable maskable interrupts

WAIT  B      WAIT

;=======================================================================
; I S R  -  GISR2
;
; Description:    Calculates the number of seconds elapsed and
;           accordingly outputs the numbers to the LEDs
;
; Modifies: MSEC_CTR, SEC_CTR
;=======================================================================
GISR2 LDP   #MSEC_CTR         ;Set data page
      LACC  MSEC_CTR
      ADD   #1
      SACL  MSEC_CTR          ;Increment millisecond counter
      SUB   #1000             ;Test if it reaches 1000
      BCND  BR1,NEQ           ;If not, branch to BR1
      SPLK  #0, MSEC_CTR      ;Else reset millisec counter
      LACC  SEC_CTR
      ADD   #1
      SACL  SEC_CTR           ;Increment second counter
```

```
      OUT   SEC_CTR,LEDS       ;Display second counter content
      SUB   #00Fh              ;Test if it reaches its maximum
      BCND  BR1, NEQ           ;If not, branch to BR1
      SPLK  #0, SEC_CTR        ;Else reset second counter
BR1   LDP   #DP_EVA            ;Set data page
      SPLK  #0FFFFh, EVAIFRA  ;clear all EVA group A interrupts
      CLRC  INTM               ;Enable maskable interrupts
      RET                      ;Return from interruption

;=====================================================================
; I S R  -  PHANTOM
;
; Description:    Dummy ISR, used to trap spurious interrupts.
;
; Modifies: Nothing
;=====================================================================
PHANTOM     B     PHANTOM
GISR1       RET
;GISR2      RET
GISR3       RET
GISR4       RET
GISR5       RET
GISR6       RET
```

LABORATORY EXPERIMENT 3

TIMER OPERATIONS

Objectives

In this lab, the students will learn the operations of timer functions found in the Event module of TMS320F2407. The timer functions covered in this lab are GP timer compare, input capture, and pulse-width modulation functions. Students will write programs to control the operations of these timer functions for various applications including waveforms generations and period/frequency measurements.

Procedure

Setup

1. Make sure that the EVM system has been properly setup as in the previous lab.
2. Turn on the PC and run the EVM Testing program.

Laboratory Assignments

1. Square-wave signals generation using output compare function

Write a program for that outputs a square-wave with frequency of 2 kHz at pin T2PWM/T2CMPR/IOPB5.

1) Compile the program and download it to the EVM.
2) Connect the pin T2PWM/T2CMPR/IOPB5 (pin 13 on connector P1) and GND (pin 33 on connector P1) to oscilloscope.
3) Run the downloaded program.
4) Verify that the signal has the desired frequency on oscilloscope.
5) Repeat step 1 to 4 above for the following square wave signal frequencies :
   (a) 20 kHz
   (b) 100 kHz
   (c) 500 Hz

(d) 60 Hz

2. PWM signals generation

1) Write a program that outputs a 500 Hz PWM signal with 30% duty cycle at pin T2PWM/T2CMPR/IOPB5.
2) Compile the program and download it.
3) Run the downloaded program.
4) Verify that the signal has the desired frequency and duty cycle on oscilloscope.
5) Change the values of the duty cycle in your program to the following:

   (a) 75% (b) 50% (c) 25% (d) 10 %

   and repeat step 2 to 4
6) Investigate the result of your program for high duty cycle PWM signals. Try different duty cycle values which are close to 100 % and observe the results. Does you program produce the intended results? Explain what happens when your program tries to generate PWM signals with duty cycle close to 100%. Obtain the maximum duty cycle your program can generate! Does your program have the same problems when generating PWM signal with low duty cycle? Explain.

3. Period and Frequency Measurements using Capture Functions

1) Write a program that measures the period and frequency of a square wave signal. Use the input capture pin CAP2 as the signal input.
2) Connect a square wave input from the signal generator to CAP2 (pin 22 of connector P1) and GND (pin 33 of connector P1)
3) Compile, download and run your program.
4) Tabulate results for the following input frequencies from signal generator:
   - ❑ 100 Hz
   - ❑ 1 kHz
   - ❑ 50 Hz