EVM Application #3

Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM

APPLICATION REPORT: SPRA412

David Figoli

Digital Signal Processing Solutions
January 1999

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE ("CRITICAL APPLICATIONS"). TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF TI PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty, or endorsement thereof.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Contents

Figures

Tables

EVM Application #3

Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM

Abstract

This document explains how the EVM Application #3 creates a 2-channel sine wave generator using the 12-bit digital-to-analog converter (DAC) of the Texas Instruments (TI™) TMS320F240 Evaluation Module (EVM). It contains:

- An overview that explains how this application functions
- Information on the two modules used
- Information on how to use commands in the debugger environment
- Equations for calculating either the frequency of a sine wave or the displacement value to load into the modulo register of the corresponding sine wave
- Graphics showing two types of interpolation schemes
- Tables showing the phase differences between examples with interpolation and without interpolation
- The C2xx Assembly code that implements the application

Product Support

World Wide Web

Our World Wide Web site at www.ti.com contains the most up to date product information, revisions, and additions. Users registering with TI&ME can build custom information pages and receive new product updates automatically via email.

Email

For technical issues or clarification on switching products, please send a detailed email to dsph@ti.com. Questions receive prompt attention and are usually answered within one business day.

Overview

This application creates a two-channel sine wave generator using the 12-bit digital-to-analog converter (DAC) of the EVM. The generator provides the user the ability to modify the frequency of the sine waves, the phase difference between the 2 waves, and the magnitude (peak to peak) of the waves. The sine wave generator is implemented using C2xx Assembly code. The algorithm described in this application report was implemented using the TI TMS320F240 EVM.

Module(s) Used

- Event Manager Module
- General Purpose Timer 1

Input

None

Output

DAC0OUT

DAC1OUT

Background and Methodology

This implementation for producing a dual channel sine wave generator with the DAC is similar in its setup to that shown in Application #2 (PWM1.ASM). However, rather than placing the sine value from the look up table into the compare register of the timer, the value from the sine look up table is placed into the channel register of the DAC. The modulation of the sine wave is performed using the modulo counting register, as in Application #2.

The DAC module requires wait states for proper operation because the DAC registers are mapped to the I/O space of the F240. Consequently, the wait state generator (WSGR) of the F240 needs to be set to generate one software wait state for I/O space access. Additionally, the CPUCLK needs to be output on the CLKOUT pin of the device so that the GAL may generate the additional hardware wait states required by the DAC module.

The generation of the sine wave is performed using a look up table. As in Application #2, a rolling 16 bit counter is used to determine the location of the value to be placed in the DAC. A step value is added to the counter every time a new value from the sine table is to be loaded. By changing the value of the step, one can accurately control the frequency of the sine wave. However, to be able to calculate the frequency that will be produced, a “sampling” method is used as in the previous application. The “sampling” is accomplished using an interrupt service routine that is executed when an interrupt is generated by the timer counter. Thus, by manipulating the period register, one can set how often the DAC register will be updated with a new sine value.

As in the previous application, the frequency of the sine wave that will be output can be calculated using the following equation

$$f(step) = \frac{step}{T_S \times 2^n}$$

where

f(step) = desired frequency

T_S = the period of the frequency at which the DAC register values are updated

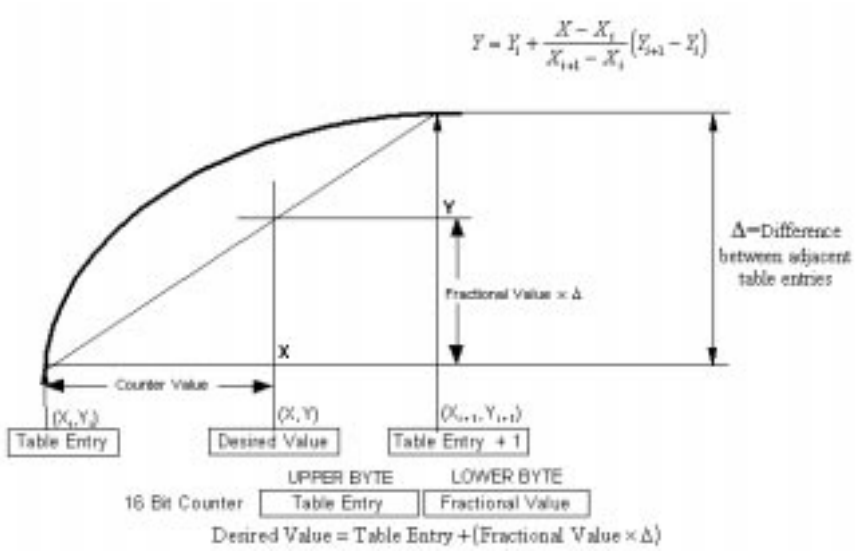
step = the step increment that is added to the counter to change the frequency

n = the number of bits in the counting register

One difference between this application and the previous application (PWM1.ASM) is that instead of using the look up table for all of the data points, this application interpolates the output value by making use of the lower byte of the 16 bit counter. In Application #2, the value that was in the upper byte determined the location of the next look-up-table-value to output regardless of the value in lower byte.

Since the value in the lower byte has valid information (i.e., it is based on the step size), the value to be output on the DAC can be determined based on a fractional difference between the value pointed to by the upper byte of the counter register and the value adjacent to the pointed value. Using the fractional bits of the modulo register (the lower byte), one can calculate the appropriate sine value when the 16 bit value in the counting register falls between the adjacent values on the sine table.

Figure 1. Graphical Representation of the Interpolation Scheme

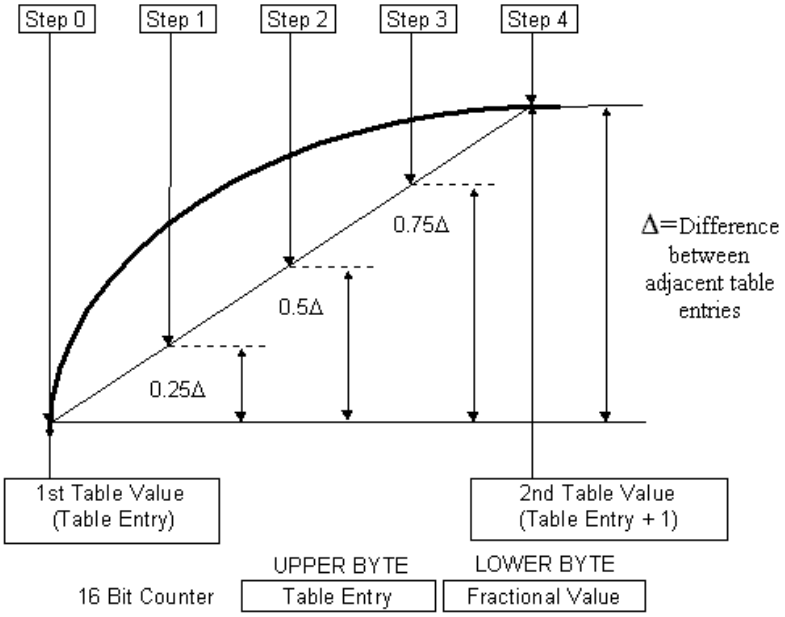


Referring back to the example used in the previous application (PWM1.ASM). The counter is set initially to 0000h and the step value is chosen to be the value of 40h. For example, then:

Step	Counter	Pointer	Fraction(Q15)	Step Value = 40h
0	00 00h	00h	00h(00h)→0	1st value in look up table
1	00 40h	00h	40h(20h)→0.25 1st value + 0.25Δ	
2	00 80h	00h	80h(40h)→0.5	1st value + 0.5Δ
3	00 C0h	00h	C0h(60h)→0.75 1st value + 0.75Δ	
4	01 00h	01h	00h(00h)→0	2nd value in look up table

Δ = 2nd value – 1st value ➔ difference between adjacent values

Figure 2. Graphical Representation of Interpolation Scheme Using a Step Size of 40



Because this application interpolates the values that will be input into the DAC, controlling the difference in the phase between the 2 sine waves becomes a simple procedure. Interpolation provides a way to output values that fall between table entry values. As a result, a user could theoretically increment the phase difference by 0.0007°, which equates to a difference of 1 between the two counting registers.

Without the interpolation, the phase could only be modified in steps of 1.41° (360°/256) because the look-up table only contains 256 entries. Thus, unless the counting register of one of the sine waves was pre-loaded with a value that had a value in the upper byte, the phase difference will be initially zero with a potential difference of 1.41° or any multiple of 1.41°.

For example, two sine waves need to be generated each with a step size of 40h. One counter is set to 0000h and the other 0080h. Notice the phase difference between the two examples, one with interpolation and the other without interpolation.

Step size = 40h

T = Table Entry Pointer

F = Fractional Value

*Table 1. **Without** Interpolation*

Step	Counter1	Counter2	Value1	Value2	Phase		
	T	F	T	F			
0	00	00h	00	80h	1^{st}	1^{st}	0°
1	00	40h	00	C0h	1^{st}	1^{st}	0°
2	00	80h	01	00h	1^{st}	2^{nd}	1.41°
3	00	C0h	01	40h	1^{st}	2^{nd}	1.41°
4	01	00h	01	80h	2^{nd}	2^{nd}	0°

*Table 2. **With** Interpolation*

Step	Counter1	Counter2	Value1	Value2	Phase		
	T	F	T	F			
0	00	00h	00	80h	1^{st}	$1^{st} + 0.5\Delta$	0.7°
1	00	40h	00	C0h	$1^{st} + 0.25\Delta$	$1^{st} + 0.75\Delta$	0.7°
2	00	80h	01	00h	$1^{st} + 0.5\Delta$	2^{nd}	0.7°
3	00	C0h	01	40h	$1^{st} + 0.75\Delta$	$2^{nd} + 0.25\Delta$	0.7°
4	01	00h	01	80h	2^{nd}	$2^{nd} + 0.5\Delta$	0.7°

Δ = difference between the value that the pointer is pointing to and the next value

With the implementation of the interpolation routine, any value that is input into the modulo register will output a proportional value, thus the phase difference will be accurate and consistent.

To calculate the displacement value to load into the modulo register of the corresponding sine wave, the following formula can be used

$$d(\phi) = \frac{\phi}{360^\circ} \times 2^n$$

where

d(ϕ) = displacement value for the modulo counter register

ϕ = desired phase difference in degrees

n = number of bits in the counter register

To modify the peak to peak voltage of the sine wave that is output on the DAC, one can modify the values by multiplying the value by a scaling factor. Since the maximum output of the DAC is 5V, by determining the fraction that the desired peak to peak value is of 5V and by multiplying the fraction by the most positive Q value, the output value will be scaled accordingly. The following formula provides the value to be used in the magnitude register of the appropriate sine wave.

$$A(m) = \frac{m}{5\text{V}} \times \left(2^Q - 1\right)$$

where

A(m) = value for the magnitude register

m = desired peak to peak voltage

Q = Q format used (e.g. 16 bit word size, Q = 15)

The sine waves that are output can be modified in the debugger environment. By using the previous equations, the values controlling the phase, frequency, and magnitude can be modified in a watch window.

By entering the following commands in the debugger environment, one can view the values loaded in the registers corresponding to each sine wave and the values can be manipulated.

wa *FREQSTEP1,,u

wa *MODREG1,,x

wa *MAG1,,x

wa *FREQSTEP2,,u

wa *MODREG2,,x

wa *MAG2,,x

FREQSTEPx - will modify the frequency of the corresponding sine wave on the DAC output channels

MODREGx - will modify the starting point of the wave, thus if two sine waves have the same frequency, then the phase difference between the two wave forms can be set

MAGx - will modify the peak to peak voltage of the sine wave output on the corresponding DAC channel.

Since this program is interrupt driven, the program can be ended with an unconditional branch and the sine waves will continue to be output. To modify the registers, the program should be halted, registers changed, and program resumed. Restarting the program will return the registers back to their original values.

```
;*****************************************************************
; File Name:        dac0.asm
; Originator:       Digital Control systems Apps group - Houston
; Target System: 'C24x Evaluation Board
;
; Description:   Outputs 2 Sine Waves on the EVM DAC - DAC0 and DAC1
;                Sine waves generated through a look up table and
;                interpolation.
;
;                By entering the following commands in the debugger
;                environment, one can view the values loaded in the
;                registers corresponding to each sine wave and the
;                values can be manipulated.
;
;                        wa *FREQSTEP1,,u
;                        wa *MODREG1,,x
;                        wa *MAG1,,x
;                        wa *FREQSTEP2,,u
;                        wa *MODREG2,,x
;                        wa *MAG2,,x
;
;                FREQSTEPx - will modify the frequency of the
;                corresponding sine wave on the DAC output channels
;
;                MODREGx - will modify the starting point of the
;                wave, thus if two sine waves have the same
;                frequency, then the phase difference between the
;                two waveforms can be set
;
;                MAGx - will modify the peak to peak voltage of the
;                sine wave output on the corresponding DAC channel
;
; Last Updated:   20 June 1997
;
;*****************************************************************
                   .include f240regs.h

;-----------------------------------------------------------------
; I/O Mapped EVM Registers
;-----------------------------------------------------------------
DAC0            .set  0000h    ;Input Data Register for DAC0
DAC1            .set  0001h    ;Input Data Register for DAC1
DAC2            .set  0002h    ;Input Data Register for DAC2
DAC3            .set  0003h    ;Input Data Register for DAC3
DACUPDATE       .set  0004h    ;DAC Update Register

;-----------------------------------------------------------------
; Variables Declaration for B2
;-----------------------------------------------------------------
                .bss  GPR0,1     ;General Purpose Register
                .bss  DAC0VAL,1  ;DAC0 Channel Value
                .bss  DAC1VAL,1  ;DAC1 Channel Value
```

```
                .bss  DAC2VAL,1   ;DAC2 Channel Value
                .bss  DAC3VAL,1   ;DAC3 Channel Value

;-----------------------------------------------------------------
; Vector address declarations
;-----------------------------------------------------------------
                .sect   ".vectors"

RSVECT          B    START    ; Reset Vector
INT1            B    PHANTOM  ; Interrupt Level 1
INT2            B    SINE     ; Interrupt Level 2
INT3            B    PHANTOM  ; Interrupt Level 3
INT4            B    PHANTOM  ; Interrupt Level 4
INT5            B    PHANTOM  ; Interrupt Level 5
INT6            B    PHANTOM  ; Interrupt Level 6
RESERVED        B    PHANTOM  ; Reserved
SW_INT8         B    PHANTOM  ; User S/W Interrupt
SW_INT9         B    PHANTOM  ; User S/W Interrupt
SW_INT1         B    PHANTOM  ; User S/W Interrupt
SW_INT1         B    PHANTOM  ; User S/W Interrupt
SW_INT12        B    PHANTOM  ; User S/W Interrupt
SW_INT13        B    PHANTOM  ; User S/W Interrupt
SW_INT14        B    PHANTOM  ; User S/W Interrupt
SW_INT15        B    PHANTOM  ; User S/W Interrupt
SW_INT16        B    PHANTOM  ; User S/W Interrupt
TRAP            B    PHANTOM  ; Trap vector
NMINT           B    PHANTOM  ; Non-maskable Interrupt
EMU_TRAP        B    PHANTOM  ; Emulator Trap
SW_INT20        B    PHANTOM  ; User S/W Interrupt
SW_INT21        B    PHANTOM  ; User S/W Interrupt
SW_INT22        B    PHANTOM  ; User S/W Interrupt
SW_INT23        B    PHANTOM  ; User S/W Interrupt


;=================================================================
; M A I N  C O D E  - starts here
;=================================================================
                .text
                NOP
START:    SETC  INTM             ;Disable interrupts
                SPLK  #0002h,IMR ;Mask all core interrupts
                                 ;  except INT2

                LACC  IFR        ;Read Interrupt flags
                SACL  IFR        ;Clear all interrupt flags

                CLRC  SXM        ;Clear Sign Extension Mode
                CLRC  OVM        ;Reset Overflow Mode
                CLRC  CNF        ;Config Block B0 to Data mem

;-----------------------------------
; Set up PLL Module
```

```
;-----------------------------------
                LDP     #00E0h

;The following line is necessary if a previous program set the PLL
;to a different setting than the settings which the application
;uses.  By disabling the PLL, the CKCR1 register can be modified so
;that the PLL can run at the new settings when it is re-enabled.

SPLK            #0000000001000001b,CKCR0  ;CLKMD=PLL Disable
                                          ;SYSCLK=CPUCLK/2

;                     5432109876543210
                SPLK  #0000000010111011b,CKCR1  ;CLKIN(OSC)=10MHz
                                                ;CPUCLK=20MHz

;CKCR1 - Clock Control Register 1
;   Bits 7-4    (1011) CKINF(3)-CKINF(0) - Crystal or Clock-In
;                          Frequency
;                      Frequency = 10MHz
;   Bit 3       (1)   PLLDIV(2) - PLL divide by 2 bit
;                      Divide PLL input by 2
;   Bits 2-0    (011) PLLFB(2)-PLLFB(0) - PLL multiplication ratio
;                      PLL Multiplication Ration = 4


;                     5432109876543210
                SPLK  #0000000011000001b,CKCR0  ;CLKMD=PLL Enable
                                                ;SYSCLK=CPUCLK/2

;CKCR0 - Clock Control Register 0
;   Bits 7-6    (11)  CLKMD(1),CLKMD(0) - Operational mode of Clock
;                      Module
;                      PLL Enabled; Run on CLKIN on exiting low
;                      power mode
;   Bits 5-4    (00)  PLLOCK(1),PLLOCK(0) - PLL Status. READ ONLY
;   Bits 3-2    (00)  PLLPM(1),PLLPM(0) - Low Power Mode
;                      LPM0
;   Bit 1       (0)   ACLKENA - 1MHz ACLK Enable
;                      ACLK Disabled
;   Bit 0       (1)   PLLPS - System Clock Prescale Value
;                      f(sysclk)=f(cpuclk)/2


;                     5432109876543210
                SPLK  #0100000011000000b,SYSCR ;CLKOUT=CPUCLK

;SYSCR - System Control Register
;   Bit 15-14   (01)  RESET1,RESET0 - Software Reset Bits
;                      No Action
;   Bits 13-8   (000000) Reserved
;   Bit 7-6     (11)  CLKSRC1,CLKSRC0 - CLKOUT-Pin Source Select
;                      CPUCLK: CPU clock output mode
```

```
;   Bit 5-0      (000000)   Reserved


   SPLK       #006Fh, WDCR  ;Disable WD if VCCP=5V (JP5 in pos. 2-3)
   KICK_DOG                 ;Reset Watchdog
```

```
;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
;- Event Manager Module Reset
;*
;-    This section resets all of the Event Manager Module
;*    Registers. This is necessary for silicon revision
;-    1.1; however, for silicon revisions 2.0 and later,
;*    this is not necessary.
;-
;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

LDP             #232    ;DP=232 Data Page for the Event Manager
SPLK            #0000h,GPTCON ;Clear General Purpose Timer Control
SPLK            #0000h,T1CON  ;Clear GP Timer 1 Control
SPLK            #0000h,T2CON  ;Clear GP Timer 2 Control
SPLK            #0000h,T3CON  ;Clear GP Timer 3 Control

SPLK            #0000h,COMCON ;Clear Compare Control
SPLK            #0000h,ACTR   ;Clear Full Compare Action
                              ;  Control Register
SPLK            #0000h,SACTR  ;Clear Simple Compare Action
                              ;  Control Register
SPLK            #0000h,DBTCON ;Clear Dead-Band Timer
                              ;  Control Register

SPLK            #0000h,CAPCON ;Clear Capture Control

SPLK            #0FFFFh,EVIFRA;Clear Interrupt Flag Register A
SPLK            #0FFFFh,EVIFRB;Clear Interrupt Flag Register B
SPLK            #0FFFFh,EVIFRC;Clear Interrupt Flag Register C

SPLK            #0000h,EVIMRA ;Clear Event Manager Mask Register A
SPLK            #0000h,EVIMRB ;Clear Event Manager Mask Register B
SPLK            #0000h,EVIMRC ;Clear Event Manager Mask Register C

;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
;-              End of RESET section for silicon revision 1.1    *
;*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

;-----------------------------------
; Set up Event Manager Module
;-----------------------------------

T1COMPARE       .set  0     ;Compare value not necessary
T1PERIOD        .set  610   ;T1PERIOD set to value
                           ;equivalent to
                           ;32.768kHz with CPULCLK =
                           ;20MHz

                .text
                LDP   #232        ;DP=232, Data Page for Event
                                  ;Manage Addresses
```

```
                SPLK  #T1COMPARE,T1CMPR

;                      2109876543210
                SPLK  #0000001010101b,GPTCON

;GPTCON - GP Timer Control Register
;Bit 15          (0)   T3STAT - GP Timer 3 Status.  READ ONLY
;Bit 14          (0)   T2STAT - GP Timer 2 Status.  READ ONLY
;Bit 13          (0)   T1STAT - GP Timer 1 Status.  READ ONLY
;Bits 12-11      (00)  T3TOADC - ADC start by event of GP Timer 3
;                        No event starts ADC
;Bits 10-9       (00)  T2TOADC - ADC start by event of GP Timer 2
;                        No event starts ADC
;Bits 8-7        (00)  T1TOADC - ADC start by event of GP Timer 1
;                        No event starts ADC
;Bit 6           (1)   TCOMPOE - Compare output enable
;                        Enable all three GP timer compare outputs
;Bits 5-4        (01)  T3PIN - Polarity of GP Timer 3 compare output
;                        Active Low
;Bits 3-2        (01)  T2PIN - Polarity of GP Timer 2 compare output
;                        Active Low
;Bits 1-0        (01)  T1PIN - Polarity of GP Timer 1 compare output
;                        Active Low

                SPLK    #T1PERIOD,T1PR
                SPLK    #0000h,T1CNT
                SPLK    #0000h,T2CNT
                SPLK    #0000h,T3CNT


;                      5432109876543210
                SPLK  #0001000000000100b,T1CON

;T1CON - GP Timer 1 Control Register
;Bits 15-14      (00)  FREE,SOFT - Emulation Control Bits
;                        Stop immediately on emulation suspend
;Bits 13-11      (010) TMODE2-TMODE0 - Count Mode Selection
;                        Continuous-Up Count Mode
;Bits 10-8       (000) TPS2-TPS0 - Input Clock Prescaler
;                        Divide by 1
;Bit 7           (0)   Reserved
;Bit 6           (0)   TENABLE - Timer Enable
;                        Disable timer operations
;Bits 5-4        (00)  TCLKS1,TCLKS0 - Clock Source Select
;                        Internal Clock Source
;Bits 3-2        (01)  TCLD1,TCLD0 - Timer Compare Register Reload
;                        Condition
;                        When counter is 0 or equals period
;                        register value
;Bit 1           (0)   TECMPR - Timer compare enable
;                        Disable timer compare operation
;Bit 0           (0)   Reserved
```

```
;                       5432109876543210
                SPLK  #0000000000000000b,T2CON     ;Not Used
```

```
;T2CON - GP Timer 2 Control Register
;Bits 15-14      (00)  FREE,SOFT - Emulation Control Bits
;                        Stop immediately on emulation suspend
;Bits 13-11      (000) TMODE2-TMODE0 - Count Mode Selection
;                        Stop/Hold
;Bits 10-8       (000) TPS2-TPS0 - Input Clock Prescaler
;                        Divide by 1
;Bit 7           (0)   TSWT1 - GP Timer 1 timer enable bit
;                        Use own TENABLE bit
;Bit 6           (0)     TENABLE - Timer Enable
;                        Disable timer operations
;Bits 5-4        (00)  TCLKS1,TCLKS0 - Clock Source Select
;                        Internal Clock Source
;Bits 3-2        (00)  TCLD1,TCLD0 - Timer Compare Register Reload
;                           Condition
;                        When counter is 0
;Bit 1           (0)   TECMPR - Timer compare enable
;                        Disable timer compare operation
;Bit 0           (0)   SELT1PR - Period Register select
;                        Use own period register

;                       5432109876543210
                SPLK  #0000000000000000b,T3CON  ;  Not Used

;T3CON - GP Timer 3 Control Register
;Bits 15-14      (00)  FREE,SOFT - Emulation Control Bits
;                        Stop immediately on emulation suspend
;Bits 13-11      (000) TMODE2-TMODE0 - Count Mode Selection
;                        Stop/Hold
;Bits 10-8       (000) TPS2-TPS0 - Input Clock Prescaler
;                        Divide by 1
;Bit 7           (0)   TSWT1 - GP Timer 1 timer enable bit
;                        Use own TENABLE bit
;Bit 6           (0)   TENABLE - Timer Enable
;                        Disable timer operations
;Bits 5-4        (00)  TCLKS1,TCLKS0 - Clock Source Select
;                        Internal Clock Source
;Bits 3-2        (00)  TCLD1,TCLD0 - Timer Compare Register Reload
;                           Condition
;                        When counter is 0
;Bit 1           (0)   TECMPR - Timer compare enable
;                        Disable timer compare operation
;Bit 0           (0)   SELT1PR - Period Register select
;                        Use own period register

                SBIT1   T1CON,B6_MSK  ;Sets Bit 6 of T1CON

;T1CON - GP Timer 1 Control Register
;Bit 6           (1)   TENABLE - Timer Enable
;                        Enable Timer Operations
```

```
              SPLK  #0080h,EVIMRA ;Enable Timer 1 Period Interrupt

;---------------------------------------------------------------------
; Initialize Variables for Generation of Sine Wave on DAC
;---------------------------------------------------------------------

;The DAC module requires that wait states be generated for proper
;operation.

    LDP           #0000h        ;Set Data Page Pointer
                                ;to 0000h, Block B2
    SPLK          #4h,GPR0      ;Set Wait State
                                ;Generator for
    OUT           GPR0,WSGR     ;Program Space, 0WS
                                ;Date Space, 0WS
                                ;I/O Space, 1WS

    .bss          TABLE,1       ;Keeps address of the pointer
                                ;in the SINE Table
    .bss          TOPTABLE,1    ;Keeps the reset value for the
                                ;pointer
    .bss          COMPARET1,1   ;A register to do calculations
                                ;since the T1CMPR register is double
                                ;buffered.
   .bss           REMAINDER,1   ;Remainder of the MODREGx
                                      ;values
   .bss           VALUE,1       ;SINE Table Value
   .bss           NEXTVALUE,1   ;Next entry in the SINE Table
   .bss           DIFFERENCE,1  ;Difference between Entries

   .bss           FREQSTEP1,1   ;Frequency modulation of the
                                ;1st sine wave
   .bss           MODREG1,1     ;Rolling Modulo Register for
                                ;1st sine wave
   .bss           MAG1,1        ;Magnitude of the frequency for
                                ;1st sine wave

   .bss           FREQSTEP2,1   ;Frequency modulation of the
                                ;2nd sine wave
   .bss           MODREG2,1     ;Rolling Modulo Register for
                                ;2nd sine wave
   .bss           MAG2,1        ;Magnitude of the frequency for
                                ;2nd sine wave

NORMAL .SET       500

   .text
   SPLK           #0000h,TABLE
   SPLK           #STABLE,TOPTABLE

   SPLK           #1000,FREQSTEP1  ;Controls the frequency for
```

```
                                    ;DAC0
   SPLK          #0000h,MODREG1     ;Sets the starting point
   SPLK          #7FFFh,MAG1        ;Maximum value, Q15
   SPLK          #1000,FREQSTEP2    ;Controls the frequency for
                                    ;DAC1
   SPLK          #4000h,MODREG2     ;Sets the starting point
   SPLK          #7FFFh,MAG2        ;Maximum value, Q15

                 CLRC  INTM

END              B          END

;---------------------------------------------------------------------
; Generate Sine Wave ISR
;---------------------------------------------------------------------

;The following section performs the necessary calculations for the
;first sine wave

SINE             LDP   #0
                 LACC  MODREG1       ;ACC loaded with the counting
                                     ;register
                 ADD   FREQSTEP1     ;Counting Register increased by
                                     ;specific step
                 SACL  MODREG1       ;Store the updated the counter
                                     ;value
                 LACC  MODREG1,8     ;Reload the new cntr val, shift
                                     ;left by 8 bits
                 SACH  TABLE         ;Store the high bit into the
                                     ;TABLE as pointer
                 SFR                 ;Shift the value to the right
                                     ;convert to Q15
                 AND   #07FFFh       ;Make sure the Q15 value is
                                     ;positive
                 SACL  REMAINDER     ;Store the frac value of the
                                     ;counting reg.
                 LACC  TABLE         ;Load the acc with the proper
                                     ;index value
                 ADD   TOPTABLE      ;Displace the ACC with the
                                     ;starting address
                 TBLR  VALUE         ;Read the value from the table
                                     ;and store
                 ADD   #1            ;Increment the ACC to the next
                                     ;address
                 TBLR  NEXTVALUE     ;Read the next val from the
                                     ;table and store
                 LACC  NEXTVALUE     ;Load the ACC with NEXTVALUE
                 SUB   VALUE         ;Subtract the previous value
                 SACL  DIFFERENCE    ;Store the difference between
                                     ;the values
                 LT    DIFFERENCE    ;Load the TREG with DIFFERENCE
                 MPY   REMAINDER     ;Multiply the DIFFERENCE with
```

```
                                    ;REMAINDER
                    PAC             ;Move the product to the ACC
                    SACH  REMAINDER,1   ;Store the upper byte and shift
                                        ;left by 1, Q15
                    LACC  REMAINDER     ;Load ACC with new REMAINDER
                    ADD   VALUE         ;Add VALUE to get the new
                                        ;interpolated value
                    SACL  VALUE         ;Store the interpolated value
                                        ;into VALUE
                    LT    VALUE         ;Load the TREG with the new
                                        ;interpolated VALUE
                    MPY   MAG1          ;Multiply VALUE by a magnitude
                    PAC                 ;Move the product to ACC
                    SACH  DAC0VAL,1     ;Store the new value, shift to
                                        ;get Q15

;The following section performs the necessary calculations for the
;second sine wave

                    LACC  MODREG2       ;ACC loaded with the counting
                                        ;register
                    ADD   FREQSTEP2     ;Counting Register increased by
                                        ;specific step
                    SACL  MODREG2       ;Store the updated the counter
                                        ;value
                    LACC  MODREG2,8     ;Reload new ctr value but shift
                                        ;left by 8 bits
                    SACH  TABLE         ;Store the high bit as pointer
                                        ;to lookup table
                    SFR                 ;Shift the value to the right
                                        ;convert to Q15
                    AND   #07FFFh       ;Make sure the Q15 value is
                                        ;positive
                    SACL  REMAINDER     ;Store the frac value of the
                                        ;counting reg
                    LACC  TABLE         ;Load the acc with the proper
                                        ;index value
                    ADD   TOPTABLE      ;Displace the ACC with the
                                        ;starting address
                    TBLR  VALUE         ;Read the value from the table
                                        ;and store
                    ADD   #1            ;Increment the ACC to the next
                                        ;address
                    TBLR  NEXTVALUE     ;Read the next value from the
                                        ;table and store
                    LACC  NEXTVALUE     ;Load the ACC with NEXTVALUE
                    SUB   VALUE         ;Subtract the previous value
                    SACL  DIFFERENCE    ;Store the difference between
                                        ;the values

                    LT    DIFFERENCE    ;Load the TREG with DIFFERENCE
                    MPY   REMAINDER     ;Multiply the DIFFERENCE with
```

```
                        ;REMAINDER
PAC                     ;Move the product to the ACC
SACH  REMAINDER,1       ;Store the upper byte, shift
                        ;left by 1, Q15
LACC  REMAINDER         ;Load ACC with new REMAINDER
ADD   VALUE             ;Add VALUE to get the new
                        ;interpolated value
SACL  VALUE             ;Store the interpolated value
                        ;into VALUE

LT    VALUE             ;Load the TREG with the new
                        ;interpolated VALUE
MPY   MAG2              ;Multiply VALUE by a magnitude
PAC                     ;Move the product to ACC
SACH  DAC1VAL,1         ;Store the new value, shift to
                        ;get Q15


LDP   #0                ;This section outputs the SINE
                        ;wave to the DAC
LACC  DAC0VAL           ;ACC = DAC0VAL - entry from the
                        ;lookup table
ADD   #8000h            ;Displace the value half the
                        ;maximum
SFR                     ;Shift over 4 places since the
                        ;DAC is 12bits
SFR
SFR
SFR
SACL  DAC0VAL           ;Store the new 12 bit value
                        ;into DAC0VAL

LACC  DAC1VAL           ;ACC = DAC0VAL - entry from the
                        ;lookup table
ADD   #8000h            ;Displace the value half the
                        ;maximum
SFR                     ;Shift over 4 places since the
                        ;DAC is 12bits
SFR
SFR
SFR
SACL  DAC1VAL           ;Store the new 12 bit value
                        ;into DAC0VAL


OUT   DAC0VAL,DAC0      ;Stores the 12 bit value into
                        ;DAC0 register
OUT   DAC1VAL,DAC1      ;Stores the 12 bit value into
                        ;DAC1 register
OUT   DAC0VAL,DACUPDATE      ;Causes the DAC to
                             ; output the value
```

```
RESUME          LDP   #232          ;DP = 232 - DP for Event
                                    ;Manager
                LACC  EVIFRA        ;Load EVIFRA - Type A Interrupt
                                    ;Flags
                SACL  EVIFRA        ;Clear the Interrupt Flags
                CLRC  INTM          ;Enable Interrupts
                RET                 ;Return from Interrupt
```

```
;---------------------------------------------------------------------
; Sine look-up table
; No. Entries       : 256
; Angle Range       : 360 deg
; Number format     : Q15 with range -1 < N < +1
;---------------------------------------------------------------------
;                  SINVAL       ;       Index      Angle  Sin(Angle)
STABLE  .word          0       ;           0          0     0.0000
        .word        804       ;           1       1.41     0.0245
        .word       1608       ;           2       2.81     0.0491
        .word       2410       ;           3       4.22     0.0736
        .word       3212       ;           4       5.63     0.0980
        .word       4011       ;           5       7.03     0.1224
        .word       4808       ;           6       8.44     0.1467
        .word       5602       ;           7       9.84     0.1710
        .word       6393       ;           8      11.25     0.1951
        .word       7179       ;           9      12.66     0.2191
        .word       7962       ;          10      14.06     0.2430
        .word       8739       ;          11      15.47     0.2667
        .word       9512       ;          12      16.88     0.2903
        .word      10278       ;          13      18.28     0.3137
        .word      11039       ;          14      19.69     0.3369
        .word      11793       ;          15      21.09     0.3599
        .word      12539       ;          16      22.50     0.3827
        .word      13279       ;          17      23.91     0.4052
        .word      14010       ;          18      25.31     0.4276
        .word      14732       ;          19      26.72     0.4496
        .word      15446       ;          20      28.13     0.4714
        .word      16151       ;          21      29.53     0.4929
        .word      16846       ;          22      30.94     0.5141
        .word      17530       ;          23      32.34     0.5350
        .word      18204       ;          24      33.75     0.5556
        .word      18868       ;          25      35.16     0.5758
        .word      19519       ;          26      36.56     0.5957
        .word      20159       ;          27      37.97     0.6152
        .word      20787       ;          28      39.38     0.6344
        .word      21403       ;          29      40.78     0.6532
        .word      22005       ;          30      42.19     0.6716
        .word      22594       ;          31      43.59     0.6895
        .word      23170       ;          32      45.00     0.7071
        .word      23731       ;          33      46.41     0.7242
        .word      24279       ;          34      47.81     0.7410
        .word      24811       ;          35      49.22     0.7572
        .word      25329       ;          36      50.63     0.7730
        .word      25832       ;          37      52.03     0.7883
        .word      26319       ;          38      53.44     0.8032
        .word      26790       ;          39      54.84     0.8176
        .word      27245       ;          40      56.25     0.8315
        .word      27683       ;          41      57.66     0.8449
        .word      28105       ;          42      59.06     0.8577
        .word      28510       ;          43      60.47     0.8701
```

```
        .word   28898   ;       44      61.88   0.8819
        .word   29268   ;       45      63.28   0.8932
        .word   29621   ;       46      64.69   0.9040
        .word   29956   ;       47      66.09   0.9142
        .word   30273   ;       48      67.50   0.9239
        .word   30571   ;       49      68.91   0.9330
        .word   30852   ;       50      70.31   0.9415
        .word   31113   ;       51      71.72   0.9495
        .word   31356   ;       52      73.13   0.9569
        .word   31580   ;       53      74.53   0.9638
        .word   31785   ;       54      75.94   0.9700
        .word   31971   ;       55      77.34   0.9757
        .word   32137   ;       56      78.75   0.9808
        .word   32285   ;       57      80.16   0.9853
        .word   32412   ;       58      81.56   0.9892
        .word   32521   ;       59      82.97   0.9925
        .word   32609   ;       60      84.38   0.9952
        .word   32678   ;       61      85.78   0.9973
        .word   32728   ;       62      87.19   0.9988
        .word   32757   ;       63      88.59   0.9997
        .word   32767   ;       64      90.00   1.0000
        .word   32757   ;       65      91.41   0.9997
        .word   32728   ;       66      92.81   0.9988
        .word   32678   ;       67      94.22   0.9973
        .word   32609   ;       68      95.63   0.9952
        .word   32521   ;       69      97.03   0.9925
        .word   32412   ;       70      98.44   0.9892
        .word   32285   ;       71      99.84   0.9853
        .word   32137   ;       72      101.25  0.9808
        .word   31971   ;       73      102.66  0.9757
        .word   31785   ;       74      104.06  0.9700
        .word   31580   ;       75      105.47  0.9638
        .word   31356   ;       76      106.88  0.9569
        .word   31113   ;       77      108.28  0.9495
        .word   30852   ;       78      109.69  0.9415
        .word   30571   ;       79      111.09  0.9330
        .word   30273   ;       80      112.50  0.9239
        .word   29956   ;       81      113.91  0.9142
        .word   29621   ;       82      115.31  0.9040
        .word   29268   ;       83      116.72  0.8932
        .word   28898   ;       84      118.13  0.8819
        .word   28510   ;       85      119.53  0.8701
        .word   28105   ;       86      120.94  0.8577
        .word   27683   ;       87      122.34  0.8449
        .word   27245   ;       88      123.75  0.8315
        .word   26790   ;       89      125.16  0.8176
        .word   26319   ;       90      126.56  0.8032
        .word   25832   ;       91      127.97  0.7883
        .word   25329   ;       92      129.38  0.7730
        .word   24811   ;       93      130.78  0.7572
        .word   24279   ;       94      132.19  0.7410
        .word   23731   ;       95      133.59  0.7242
```

```
        .word       23170       ;           96        135.00    0.7071
        .word       22594       ;           97        136.41    0.6895
        .word       22005       ;           98        137.81    0.6716
        .word       21403       ;           99        139.22    0.6532
        .word       20787       ;           100       140.63    0.6344
        .word       20159       ;           101       142.03    0.6152
        .word       19519       ;           102       143.44    0.5957
        .word       18868       ;           103       144.84    0.5758
        .word       18204       ;           104       146.25    0.5556
        .word       17530       ;           105       147.66    0.5350
        .word       16846       ;           106       149.06    0.5141
        .word       16151       ;           107       150.47    0.4929
        .word       15446       ;           108       151.88    0.4714
        .word       14732       ;           109       153.28    0.4496
        .word       14010       ;           110       154.69    0.4276
        .word       13279       ;           111       156.09    0.4052
        .word       12539       ;           112       157.50    0.3827
        .word       11793       ;           113       158.91    0.3599
        .word       11039       ;           114       160.31    0.3369
        .word       10278       ;           115       161.72    0.3137
        .word       9512        ;           116       163.13    0.2903
        .word       8739        ;           117       164.53    0.2667
        .word       7962        ;           118       165.94    0.2430
        .word       7179        ;           119       167.34    0.2191
        .word       6393        ;           120       168.75    0.1951
        .word       5602        ;           121       170.16    0.1710
        .word       4808        ;           122       171.56    0.1467
        .word       4011        ;           123       172.97    0.1224
        .word       3212        ;           124       174.38    0.0980
        .word       2410        ;           125       175.78    0.0736
        .word       1608        ;           126       177.19    0.0491
        .word        804        ;           127       178.59    0.0245
        .word          0        ;           128       180.00    0.0000
        .word       64731       ;           129       181.41   -0.0245
        .word       63927       ;           130       182.81   -0.0491
        .word       63125       ;           131       184.22   -0.0736
        .word       62323       ;           132       185.63   -0.0980
        .word       61524       ;           133       187.03   -0.1224
        .word       60727       ;           134       188.44   -0.1467
        .word       59933       ;           135       189.84   -0.1710
        .word       59142       ;           136       191.25   -0.1951
        .word       58356       ;           137       192.66   -0.2191
        .word       57573       ;           138       194.06   -0.2430
        .word       56796       ;           139       195.47   -0.2667
        .word       56023       ;           140       196.88   -0.2903
        .word       55257       ;           141       198.28   -0.3137
        .word       54496       ;           142       199.69   -0.3369
        .word       53742       ;           143       201.09   -0.3599
        .word       52996       ;           144       202.50   -0.3827
        .word       52256       ;           145       203.91   -0.4052
        .word       51525       ;           146       205.31   -0.4276
        .word       50803       ;           147       206.72   -0.4496
```

```
.word       50089       ;       148       208.13    -0.4714
.word       49384       ;       149       209.53    -0.4929
.word       48689       ;       150       210.94    -0.5141
.word       48005       ;       151       212.34    -0.5350
.word       47331       ;       152       213.75    -0.5556
.word       46667       ;       153       215.16    -0.5758
.word       46016       ;       154       216.56    -0.5957
.word       45376       ;       155       217.97    -0.6152
.word       44748       ;       156       219.38    -0.6344
.word       44132       ;       157       220.78    -0.6532
.word       43530       ;       158       222.19    -0.6716
.word       42941       ;       159       223.59    -0.6895
.word       42365       ;       160       225.00    -0.7071
.word       41804       ;       161       226.41    -0.7242
.word       41256       ;       162       227.81    -0.7410
.word       40724       ;       163       229.22    -0.7572
.word       40206       ;       164       230.63    -0.7730
.word       39703       ;       165       232.03    -0.7883
.word       39216       ;       166       233.44    -0.8032
.word       38745       ;       167       234.84    -0.8176
.word       38290       ;       168       236.25    -0.8315
.word       37852       ;       169       237.66    -0.8449
.word       37430       ;       170       239.06    -0.8577
.word       37025       ;       171       240.47    -0.8701
.word       36637       ;       172       241.88    -0.8819
.word       36267       ;       173       243.28    -0.8932
.word       35914       ;       174       244.69    -0.9040
.word       35579       ;       175       246.09    -0.9142
.word       35262       ;       176       247.50    -0.9239
.word       34964       ;       177       248.91    -0.9330
.word       34683       ;       178       250.31    -0.9415
.word       34422       ;       179       251.72    -0.9495
.word       34179       ;       180       253.13    -0.9569
.word       33955       ;       181       254.53    -0.9638
.word       33750       ;       182       255.94    -0.9700
.word       33564       ;       183       257.34    -0.9757
.word       33398       ;       184       258.75    -0.9808
.word       33250       ;       185       260.16    -0.9853
.word       33123       ;       186       261.56    -0.9892
.word       33014       ;       187       262.97    -0.9925
.word       32926       ;       188       264.38    -0.9952
.word       32857       ;       189       265.78    -0.9973
.word       32807       ;       190       267.19    -0.9988
.word       32778       ;       191       268.59    -0.9997
.word       32768       ;       192       270.00    -1.0000
.word       32778       ;       193       271.41    -0.9997
.word       32807       ;       194       272.81    -0.9988
.word       32857       ;       195       274.22    -0.9973
.word       32926       ;       196       275.63    -0.9952
.word       33014       ;       197       277.03    -0.9925
.word       33123       ;       198       278.44    -0.9892
.word       33250       ;       199       279.84    -0.9853
```

```
        .word   33398   ;       200     281.25  -0.9808
        .word   33564   ;       201     282.66  -0.9757
        .word   33750   ;       202     284.06  -0.9700
        .word   33955   ;       203     285.47  -0.9638
        .word   34179   ;       204     286.88  -0.9569
        .word   34422   ;       205     288.28  -0.9495
        .word   34683   ;       206     289.69  -0.9415
        .word   34964   ;       207     291.09  -0.9330
        .word   35262   ;       208     292.50  -0.9239
        .word   35579   ;       209     293.91  -0.9142
        .word   35914   ;       210     295.31  -0.9040
        .word   36267   ;       211     296.72  -0.8932
        .word   36637   ;       212     298.13  -0.8819
        .word   37025   ;       213     299.53  -0.8701
        .word   37430   ;       214     300.94  -0.8577
        .word   37852   ;       215     302.34  -0.8449
        .word   38290   ;       216     303.75  -0.8315
        .word   38745   ;       217     305.16  -0.8176
        .word   39216   ;       218     306.56  -0.8032
        .word   39703   ;       219     307.97  -0.7883
        .word   40206   ;       220     309.38  -0.7730
        .word   40724   ;       221     310.78  -0.7572
        .word   41256   ;       222     312.19  -0.7410
        .word   41804   ;       223     313.59  -0.7242
        .word   42365   ;       224     315.00  -0.7071
        .word   42941   ;       225     316.41  -0.6895
        .word   43530   ;       226     317.81  -0.6716
        .word   44132   ;       227     319.22  -0.6532
        .word   44748   ;       228     320.63  -0.6344
        .word   45376   ;       229     322.03  -0.6152
        .word   46016   ;       230     323.44  -0.5957
        .word   46667   ;       231     324.84  -0.5758
        .word   47331   ;       232     326.25  -0.5556
        .word   48005   ;       233     327.66  -0.5350
        .word   48689   ;       234     329.06  -0.5141
        .word   49384   ;       235     330.47  -0.4929
        .word   50089   ;       236     331.88  -0.4714
        .word   50803   ;       237     333.28  -0.4496
        .word   51525   ;       238     334.69  -0.4276
        .word   52256   ;       239     336.09  -0.4052
        .word   52996   ;       240     337.50  -0.3827
        .word   53742   ;       241     338.91  -0.3599
        .word   54496   ;       242     340.31  -0.3369
        .word   55257   ;       243     341.72  -0.3137
        .word   56023   ;       244     343.13  -0.2903
        .word   56796   ;       245     344.53  -0.2667
        .word   57573   ;       246     345.94  -0.2430
        .word   58356   ;       247     347.34  -0.2191
        .word   59142   ;       248     348.75  -0.1951
        .word   59933   ;       249     350.16  -0.1710
        .word   60727   ;       250     351.56  -0.1467
        .word   61524   ;       251     352.97  -0.1224
```

```
        .word       62323       ;          252       354.38    -0.0980
        .word       63125       ;          253       355.78    -0.0736
        .word       63927       ;          254       357.19    -0.0491
        .word       64731       ;          255       358.59    -0.0245
        .word       65535       ;          256       360.00     0.0000

;=====================================================================
; I S R  -  PHANTOM
;
; Description:   Dummy ISR, used to trap spurious interrupts.
;
; Modifies:      Nothing
;
; Last Update:   16 June 95
;=====================================================================
PHANTOM           KICK_DOG          ;Resets WD counter
                  B  PHANTOM
```