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
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.

Copyright © 1999, Texas Instruments Incorporated
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
Figures
Figure 1. Graphical Representation of the Interpolation Scheme ........................................ 11
Figure 2. Graphical Representation of Interpolation Scheme Using a Step Size of 40 as an Example ................................................................. 12

Tables
Table 1. Without Interpolation ....................................................................................... 13
Table 2. With Interpolation ............................................................................................ 13
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

\[ f(\text{step}) = \frac{\text{step}}{T_s \times 2^n} \]

where

\( f(\text{step}) = \text{desired frequency} \)

\( T_s = \text{the period of the frequency at which the DAC register values are updated} \)

\( \text{step} = \text{the step increment that is added to the counter to change the frequency} \)

\( n = \text{the number of bits in the counting register} \)
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:

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

Δ = 2nd value – 1st value $\Rightarrow$ 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>
<thead>
<tr>
<th>Step</th>
<th>Counter1</th>
<th>Counter2</th>
<th>Value1</th>
<th>Value2</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>00h</td>
<td>00</td>
<td>80h</td>
<td>1st</td>
</tr>
<tr>
<td>1</td>
<td>00</td>
<td>40h</td>
<td>00</td>
<td>C0h</td>
<td>1st</td>
</tr>
<tr>
<td>2</td>
<td>00</td>
<td>80h</td>
<td>01</td>
<td>00h</td>
<td>2nd</td>
</tr>
<tr>
<td>3</td>
<td>00</td>
<td>C0h</td>
<td>01</td>
<td>40h</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>01</td>
<td>00h</td>
<td>01</td>
<td>80h</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Step</th>
<th>Counter1</th>
<th>Counter2</th>
<th>Value1</th>
<th>Value2</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>00h</td>
<td>00</td>
<td>80h</td>
<td>1st</td>
</tr>
<tr>
<td>1</td>
<td>00</td>
<td>40h</td>
<td>00</td>
<td>C0h</td>
<td>1st</td>
</tr>
<tr>
<td>2</td>
<td>00</td>
<td>80h</td>
<td>01</td>
<td>00h</td>
<td>2nd</td>
</tr>
<tr>
<td>3</td>
<td>00</td>
<td>C0h</td>
<td>01</td>
<td>40h</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>01</td>
<td>00h</td>
<td>01</td>
<td>80h</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Without Interpolation

Table 2. With Interpolation
\[ \Delta = \text{difference between the value that the pointer is pointing to and the next value} \]

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

- \( d(\phi) \) = displacement value for the modulo counter register
- \( \phi \) = 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.

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

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

Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM
Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM

```

.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_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
      SAACL 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
;CKR1 - Clock Control Register 1
; Bits 7-4  (1011) CKINF(3)-CKINF(0) - Crystal or Clock-In
;          Frequency
; 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   #0000000001100001b,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) PLLLOCK(1),PLLLOCK(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   #0100000001100000b,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

Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM
; Bit 5-0 (000000) Reserved

SPLK #006Fh, WDQR ;Disable WD if VCCP=5V (JP5 in pos. 2-3)
KICK_DOG ;Reset Watchdog
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*
<table>
<thead>
<tr>
<th>Event Manager Module Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>This section resets all of the Event Manager Registers. This is necessary for silicon revision 1.1; however, for silicon revisions 2.0 and later, this is not necessary.</td>
</tr>
</tbody>
</table>
*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*-*

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 #0FFFh,EVIFRA ;Clear Interrupt Flag Register A
SPLK #0FFFh,EVIFRB ;Clear Interrupt Flag Register B
SPLK #0FFFh,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 #000000000000000b,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

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

CLRC INTM

; 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
Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM

```assembly
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
```
Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM

PAC
SACH REMAINDER,1
LACC REMAINDER
ADD VALUE
SACL VALUE
LT VALUE
MPY MAG2
PAC
SACH DAC1VAL,1

LDP #0
LACC DAC0VAL
ADD #8000h
SFR
SFR
SFR
SACL DAC0VAL
LACC DAC1VAL
ADD #8000h
SFR
SFR
SFR
SACL DAC1VAL
OUT DAC0VAL,DAC0
OUT DAC1VAL,DAC1
OUT DAC0VAL,DACUPDATE
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

<table>
<thead>
<tr>
<th>Index</th>
<th>Angle</th>
<th>Sin(Angle)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0.0000</td>
</tr>
<tr>
<td>1</td>
<td>1.41</td>
<td>0.0245</td>
</tr>
<tr>
<td>2</td>
<td>2.81</td>
<td>0.0491</td>
</tr>
<tr>
<td>3</td>
<td>4.22</td>
<td>0.0736</td>
</tr>
<tr>
<td>4</td>
<td>5.63</td>
<td>0.0980</td>
</tr>
<tr>
<td>5</td>
<td>7.03</td>
<td>0.1224</td>
</tr>
<tr>
<td>6</td>
<td>8.44</td>
<td>0.1467</td>
</tr>
<tr>
<td>7</td>
<td>9.84</td>
<td>0.1710</td>
</tr>
<tr>
<td>8</td>
<td>11.25</td>
<td>0.1951</td>
</tr>
<tr>
<td>9</td>
<td>12.66</td>
<td>0.2191</td>
</tr>
<tr>
<td>10</td>
<td>14.06</td>
<td>0.2430</td>
</tr>
<tr>
<td>11</td>
<td>15.47</td>
<td>0.2667</td>
</tr>
<tr>
<td>12</td>
<td>16.88</td>
<td>0.2903</td>
</tr>
<tr>
<td>13</td>
<td>18.28</td>
<td>0.3137</td>
</tr>
<tr>
<td>14</td>
<td>19.69</td>
<td>0.3369</td>
</tr>
<tr>
<td>15</td>
<td>21.09</td>
<td>0.3599</td>
</tr>
<tr>
<td>16</td>
<td>22.50</td>
<td>0.3827</td>
</tr>
<tr>
<td>17</td>
<td>23.91</td>
<td>0.4052</td>
</tr>
<tr>
<td>18</td>
<td>25.31</td>
<td>0.4276</td>
</tr>
<tr>
<td>19</td>
<td>26.72</td>
<td>0.4496</td>
</tr>
<tr>
<td>20</td>
<td>28.13</td>
<td>0.4714</td>
</tr>
<tr>
<td>21</td>
<td>29.53</td>
<td>0.4929</td>
</tr>
<tr>
<td>22</td>
<td>30.94</td>
<td>0.5141</td>
</tr>
<tr>
<td>23</td>
<td>32.34</td>
<td>0.5350</td>
</tr>
<tr>
<td>24</td>
<td>33.75</td>
<td>0.5556</td>
</tr>
<tr>
<td>25</td>
<td>35.16</td>
<td>0.5758</td>
</tr>
<tr>
<td>26</td>
<td>36.56</td>
<td>0.5957</td>
</tr>
<tr>
<td>27</td>
<td>37.97</td>
<td>0.6152</td>
</tr>
<tr>
<td>28</td>
<td>39.38</td>
<td>0.6344</td>
</tr>
<tr>
<td>29</td>
<td>40.78</td>
<td>0.6532</td>
</tr>
<tr>
<td>30</td>
<td>42.19</td>
<td>0.6716</td>
</tr>
<tr>
<td>31</td>
<td>43.59</td>
<td>0.6895</td>
</tr>
<tr>
<td>32</td>
<td>45.00</td>
<td>0.7071</td>
</tr>
<tr>
<td>33</td>
<td>46.41</td>
<td>0.7242</td>
</tr>
<tr>
<td>34</td>
<td>47.81</td>
<td>0.7410</td>
</tr>
<tr>
<td>35</td>
<td>49.22</td>
<td>0.7572</td>
</tr>
<tr>
<td>36</td>
<td>50.63</td>
<td>0.7730</td>
</tr>
<tr>
<td>37</td>
<td>52.03</td>
<td>0.7883</td>
</tr>
<tr>
<td>38</td>
<td>53.44</td>
<td>0.8032</td>
</tr>
<tr>
<td>39</td>
<td>54.84</td>
<td>0.8176</td>
</tr>
<tr>
<td>40</td>
<td>56.25</td>
<td>0.8315</td>
</tr>
<tr>
<td>41</td>
<td>57.66</td>
<td>0.8449</td>
</tr>
<tr>
<td>42</td>
<td>59.06</td>
<td>0.8577</td>
</tr>
<tr>
<td>43</td>
<td>60.47</td>
<td>0.8701</td>
</tr>
<tr>
<td>.word</td>
<td>28898</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>29268</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>29621</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>29956</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>30273</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>30571</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>30852</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31113</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31356</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31580</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31785</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31971</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32137</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32285</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32412</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32521</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32609</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32678</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32728</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32757</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32767</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32757</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32728</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32678</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32609</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32521</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32412</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32285</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>32137</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31971</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31785</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31580</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31356</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>31113</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>30852</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>30571</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>30273</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>29956</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>29621</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>29268</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>28898</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>28510</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>28105</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>27683</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>27245</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>26790</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>26319</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>25832</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>25329</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>24811</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>24279</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>23731</td>
<td>;</td>
</tr>
<tr>
<td>.word</td>
<td>23170      ;</td>
<td>96</td>
</tr>
<tr>
<td>.word</td>
<td>22594      ;</td>
<td>97</td>
</tr>
<tr>
<td>.word</td>
<td>22005      ;</td>
<td>98</td>
</tr>
<tr>
<td>.word</td>
<td>21403      ;</td>
<td>99</td>
</tr>
<tr>
<td>.word</td>
<td>20787      ;</td>
<td>100</td>
</tr>
<tr>
<td>.word</td>
<td>20159      ;</td>
<td>101</td>
</tr>
<tr>
<td>.word</td>
<td>19519      ;</td>
<td>102</td>
</tr>
<tr>
<td>.word</td>
<td>18868      ;</td>
<td>103</td>
</tr>
<tr>
<td>.word</td>
<td>18204      ;</td>
<td>104</td>
</tr>
<tr>
<td>.word</td>
<td>17530      ;</td>
<td>105</td>
</tr>
<tr>
<td>.word</td>
<td>16846      ;</td>
<td>106</td>
</tr>
<tr>
<td>.word</td>
<td>16151      ;</td>
<td>107</td>
</tr>
<tr>
<td>.word</td>
<td>15446      ;</td>
<td>108</td>
</tr>
<tr>
<td>.word</td>
<td>14732      ;</td>
<td>109</td>
</tr>
<tr>
<td>.word</td>
<td>14010      ;</td>
<td>110</td>
</tr>
<tr>
<td>.word</td>
<td>13279      ;</td>
<td>111</td>
</tr>
<tr>
<td>.word</td>
<td>12539      ;</td>
<td>112</td>
</tr>
<tr>
<td>.word</td>
<td>11793      ;</td>
<td>113</td>
</tr>
<tr>
<td>.word</td>
<td>11039      ;</td>
<td>114</td>
</tr>
<tr>
<td>.word</td>
<td>10278      ;</td>
<td>115</td>
</tr>
<tr>
<td>.word</td>
<td>9512       ;</td>
<td>116</td>
</tr>
<tr>
<td>.word</td>
<td>8739       ;</td>
<td>117</td>
</tr>
<tr>
<td>.word</td>
<td>7962       ;</td>
<td>118</td>
</tr>
<tr>
<td>.word</td>
<td>7179       ;</td>
<td>119</td>
</tr>
<tr>
<td>.word</td>
<td>6393       ;</td>
<td>120</td>
</tr>
<tr>
<td>.word</td>
<td>5602       ;</td>
<td>121</td>
</tr>
<tr>
<td>.word</td>
<td>4808       ;</td>
<td>122</td>
</tr>
<tr>
<td>.word</td>
<td>4011       ;</td>
<td>123</td>
</tr>
<tr>
<td>.word</td>
<td>3212       ;</td>
<td>124</td>
</tr>
<tr>
<td>.word</td>
<td>2410       ;</td>
<td>125</td>
</tr>
<tr>
<td>.word</td>
<td>1608       ;</td>
<td>126</td>
</tr>
<tr>
<td>.word</td>
<td>804        ;</td>
<td>127</td>
</tr>
<tr>
<td>.word</td>
<td>0          ;</td>
<td>128</td>
</tr>
<tr>
<td>.word</td>
<td>64731      ;</td>
<td>129</td>
</tr>
<tr>
<td>.word</td>
<td>63927      ;</td>
<td>130</td>
</tr>
<tr>
<td>.word</td>
<td>63125      ;</td>
<td>131</td>
</tr>
<tr>
<td>.word</td>
<td>62323      ;</td>
<td>132</td>
</tr>
<tr>
<td>.word</td>
<td>61524      ;</td>
<td>133</td>
</tr>
<tr>
<td>.word</td>
<td>60727      ;</td>
<td>134</td>
</tr>
<tr>
<td>.word</td>
<td>59933      ;</td>
<td>135</td>
</tr>
<tr>
<td>.word</td>
<td>59142      ;</td>
<td>136</td>
</tr>
<tr>
<td>.word</td>
<td>58356      ;</td>
<td>137</td>
</tr>
<tr>
<td>.word</td>
<td>57573      ;</td>
<td>138</td>
</tr>
<tr>
<td>.word</td>
<td>56796      ;</td>
<td>139</td>
</tr>
<tr>
<td>.word</td>
<td>56023      ;</td>
<td>140</td>
</tr>
<tr>
<td>.word</td>
<td>55257      ;</td>
<td>141</td>
</tr>
<tr>
<td>.word</td>
<td>54496      ;</td>
<td>142</td>
</tr>
<tr>
<td>.word</td>
<td>53742      ;</td>
<td>143</td>
</tr>
<tr>
<td>.word</td>
<td>52996      ;</td>
<td>144</td>
</tr>
<tr>
<td>.word</td>
<td>52256      ;</td>
<td>145</td>
</tr>
<tr>
<td>.word</td>
<td>51525      ;</td>
<td>146</td>
</tr>
<tr>
<td>.word</td>
<td>50803      ;</td>
<td>147</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>.word 50089</td>
<td>148</td>
<td>208.13</td>
</tr>
<tr>
<td>.word 49384</td>
<td>149</td>
<td>209.53</td>
</tr>
<tr>
<td>.word 48689</td>
<td>150</td>
<td>210.94</td>
</tr>
<tr>
<td>.word 48005</td>
<td>151</td>
<td>212.34</td>
</tr>
<tr>
<td>.word 47331</td>
<td>152</td>
<td>213.75</td>
</tr>
<tr>
<td>.word 46667</td>
<td>153</td>
<td>215.16</td>
</tr>
<tr>
<td>.word 46016</td>
<td>154</td>
<td>216.56</td>
</tr>
<tr>
<td>.word 45376</td>
<td>155</td>
<td>217.97</td>
</tr>
<tr>
<td>.word 44748</td>
<td>156</td>
<td>219.38</td>
</tr>
<tr>
<td>.word 44132</td>
<td>157</td>
<td>220.78</td>
</tr>
<tr>
<td>.word 43530</td>
<td>158</td>
<td>222.19</td>
</tr>
<tr>
<td>.word 42941</td>
<td>159</td>
<td>223.59</td>
</tr>
<tr>
<td>.word 42365</td>
<td>160</td>
<td>225.00</td>
</tr>
<tr>
<td>.word 41804</td>
<td>161</td>
<td>226.41</td>
</tr>
<tr>
<td>.word 41256</td>
<td>162</td>
<td>227.81</td>
</tr>
<tr>
<td>.word 40724</td>
<td>163</td>
<td>229.22</td>
</tr>
<tr>
<td>.word 40206</td>
<td>164</td>
<td>230.63</td>
</tr>
<tr>
<td>.word 39703</td>
<td>165</td>
<td>232.03</td>
</tr>
<tr>
<td>.word 39216</td>
<td>166</td>
<td>233.44</td>
</tr>
<tr>
<td>.word 38745</td>
<td>167</td>
<td>234.84</td>
</tr>
<tr>
<td>.word 38290</td>
<td>168</td>
<td>236.25</td>
</tr>
<tr>
<td>.word 37852</td>
<td>169</td>
<td>237.66</td>
</tr>
<tr>
<td>.word 37430</td>
<td>170</td>
<td>239.06</td>
</tr>
<tr>
<td>.word 37025</td>
<td>171</td>
<td>240.47</td>
</tr>
<tr>
<td>.word 36637</td>
<td>172</td>
<td>241.88</td>
</tr>
<tr>
<td>.word 36267</td>
<td>173</td>
<td>243.28</td>
</tr>
<tr>
<td>.word 35914</td>
<td>174</td>
<td>244.69</td>
</tr>
<tr>
<td>.word 35579</td>
<td>175</td>
<td>246.09</td>
</tr>
<tr>
<td>.word 35262</td>
<td>176</td>
<td>247.50</td>
</tr>
<tr>
<td>.word 34964</td>
<td>177</td>
<td>248.91</td>
</tr>
<tr>
<td>.word 34683</td>
<td>178</td>
<td>250.31</td>
</tr>
<tr>
<td>.word 34422</td>
<td>179</td>
<td>251.72</td>
</tr>
<tr>
<td>.word 34179</td>
<td>180</td>
<td>253.13</td>
</tr>
<tr>
<td>.word 33955</td>
<td>181</td>
<td>254.53</td>
</tr>
<tr>
<td>.word 33750</td>
<td>182</td>
<td>255.94</td>
</tr>
<tr>
<td>.word 33564</td>
<td>183</td>
<td>257.34</td>
</tr>
<tr>
<td>.word 33398</td>
<td>184</td>
<td>258.75</td>
</tr>
<tr>
<td>.word 33250</td>
<td>185</td>
<td>260.16</td>
</tr>
<tr>
<td>.word 33123</td>
<td>186</td>
<td>261.56</td>
</tr>
<tr>
<td>.word 33014</td>
<td>187</td>
<td>262.97</td>
</tr>
<tr>
<td>.word 32926</td>
<td>188</td>
<td>264.38</td>
</tr>
<tr>
<td>.word 32857</td>
<td>189</td>
<td>265.78</td>
</tr>
<tr>
<td>.word 32807</td>
<td>190</td>
<td>267.19</td>
</tr>
<tr>
<td>.word 32778</td>
<td>191</td>
<td>268.59</td>
</tr>
<tr>
<td>.word 32768</td>
<td>192</td>
<td>270.00</td>
</tr>
<tr>
<td>.word 32778</td>
<td>193</td>
<td>271.41</td>
</tr>
<tr>
<td>.word 32807</td>
<td>194</td>
<td>272.81</td>
</tr>
<tr>
<td>.word 32857</td>
<td>195</td>
<td>274.22</td>
</tr>
<tr>
<td>.word 32926</td>
<td>196</td>
<td>275.63</td>
</tr>
<tr>
<td>.word 33014</td>
<td>197</td>
<td>277.03</td>
</tr>
<tr>
<td>.word 33123</td>
<td>198</td>
<td>278.44</td>
</tr>
<tr>
<td>.word 33250</td>
<td>199</td>
<td>279.84</td>
</tr>
</tbody>
</table>
Creating a Two Channel Sine Wave Generator Using the TMS320F240 EVM
; 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

.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

;===================================================================
; ISR - PHANTOM
; Description: Dummy ISR, used to trap spurious interrupts.
;
; Modifies: Nothing
;
; Last Update: 16 June 95

PHANTOM       KICK_DOG ; Resets WD counter
              B PHANTOM