EVM Application #5

Detecting a Frequency Band Using a FIR Bandpass Filter and the TMS320F240 EVM

APPLICATION REPORT: SPRA414

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

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

EVM Application #5

Detecting a Frequency Band Using a FIR Bandpass Filter and the TMS320F240 EVM

Abstract

This document explains how the EVM Application #5 interprets an input signal using a bandpass FIR filter and includes the following:

❑ An overview which explains what this application does.

❑ A description of the components involved including:

- PLL module
- Digital I/O ports
- Event Manager
- Analog-to-digital converter
- FIR filter implementation
- Details on building an FIR filter using MATLAB and Microsoft Excel

❑ Assembly code used with the target application, the Texas Instruments (TI™) TMS320F240 Evaluation Module (EVM)

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 determines when an input signal is 500Hz using a bandpass FIR filter. The input signal is converted into a digital value using the analog-to-digital converter (ADC) of the TI TMS320F240 EVM. The input value and the result from the FIR filter can be viewed using the digital-to-analog converter (DAC) of the EVM. When a 500Hz signal is present, the LEDs of the EVM are to be lit, and when the signal is gone, the LEDs are to be off. This application is implemented using C2xx Assembly. The algorithm described in this application report was implemented using the TMS320F240 EVM.

Modules Used

- ❑ Event Manager
- ❑ General Purpose Timer 1
- ❑ Analog-to-digital converter
- ❑ Digital-to-analog converter

Input

ADCIN0

Output

DAC0

DAC1

LEDs - DS1-DS8

Background and Methodology

The initial setup of the program is important because the implementation of the FIR filter requires control of the sampling frequency. Before the FIR filter can be implemented the sampling frequency needs to be known; thus, the PLL module, digital I/O ports, ADC, and the Event Manager need to be configured.

In this application, the Event Manager provides the sampling time base via the General Purpose Timer. The timers are used to control the sampling frequency of the ADC by using an interrupt.

PLL Module

Since the timer component of the Event Manager module is used to control the sampling frequency of the FIR filter, the PLL module needs to be set up properly to create the proper sampling frequency. The PLL module is set up as in Application #1 (PWM0.ASM). The PLL divide by 2 is enabled and the multiplication ratio is set to 4; as a result, with a CLKIN (crystal oscillator) value of 10MHz, the CPUCLK is then 20MHz.

Digital I/O Ports

Similar to Application #4 (ADC0.ASM), output control register A (OCRA) is configured so that the ADC can receive an input since the ADCIN0 input shares with IOPA0.

Event Manager

Once the digital I/O port and the PLL module have been set, then the Event Manager module registers can be configured. To set the sampling frequency the timer needs to be set to create an interrupt at a constant interval. Because the CPUCLK is known, the period register of a timer can be set accordingly as in Application #1 (PWM0.ASM).

For this application, the sampling frequency is set for 4000Hz, thus the value loaded into the period register of the timer is 5000. To confirm the sampling frequency of this application, the PWM output of the timer can be used. By enabling the compare function and setting the compare register to half of the period (in this case, 2500), the output square wave will have a frequency equivalent to the sampling frequency.

$$\text{Period Value} = \frac{\dfrac{\text{CLKIN(OSC)}}{\text{Input Clock Prescaler}} \times \dfrac{\text{PLL Multiplication Ratio}}{\text{PLL Divide by 2}}}{\text{Desired Sampling Frequency}}$$

Analog-to-Digital Converter

The analog-to-digital converter in this application is set up slightly differently than in Application #4 (ADC0.ASM). Because the sampling frequency is controlled by the timer component of the Event Manager module, the ADC is set up to do only a single conversion rather than a continuous conversion.

FIR Filter Implementation

Once the sampling frequency has been set by the GP Timer of the Event Manger module, the FIR filter can be designed. The FIR filter in this application was designed using MATLAB®. The coefficients for the filter were calculated by setting the order, magnitude, and frequency requirements of the desired filter. Once the values for the *b* coefficients of the FIR filter were obtained, they were then converted into hexadecimal using Microsoft® Excel.[1]

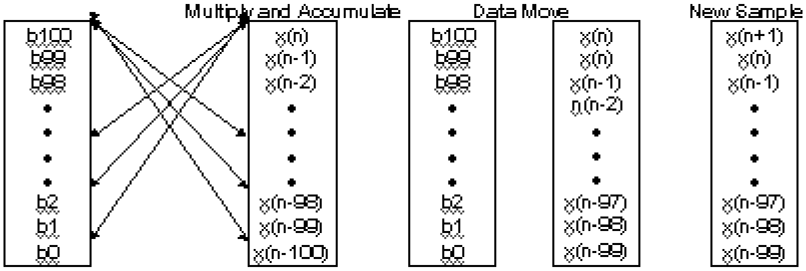
When the program first starts up, the ADC is started and then the timer. When the timer's period register matches the timer counter, an interrupt is generated. The interrupt service routine for the period interrupt performs the following

1) Fetches the converted value from ADCFIFO
2) Places the value into the data buffer of the FIR filter
3) Performs the FIR filter calculations
4) Outputs the value on the DAC
5) Lights the LEDs according what proportion the output value is of the input value
6) Restarts the ADC, and awaits for the next period interrupt

The FIR filter is a 100th order filter, as a result the buffer contains 101 samples. Initially, the buffer for the FIR filter is initialized to 0. Once an interrupt is generated, the converted value of the ADC is fetched and placed into the FIR buffer. In order to move the data through the buffer, after each calculation, the MACD command is used rather than using the MPY and the DMOV commands separately. The MACD instruction moves data from a low memory address to the next higher location after multiplying and accumulating. As a result, the most recent sample from the DC will have the lowest address in the buffer. Since the RPT instruction increments the pma in the MACD instruction with each pass, the coefficients for the filter will go in the opposite direction in memory. In other words, the coefficient to be multiplied with the oldest sample will have the lowest address in the area of memory that holds the coefficients for the FIR filter while the oldest sample will be in the highest memory location of the data buffer.

[1] Refer to the end to see how the FIR filter was designed using MATLAB® and Microsoft® Excel

Figure 1. Example Showing Data Moving Through the Buffer



The MACD command performs the equivalent of the following equation:

$$y = b_0 x(n) + b_1 x(n-1) + b_2 x(n-2) \cdots + b_{98} x(n-98) + b_{99} x(n-99) + b_{100} x(n-100)$$

Once the calculations for the FIR filter have been done, the results are then output to the DAC of the EVM to be viewed. Since the interrupt service routine of the is only accessed when the period register of the timer matches the counter register of the timer, the output from the DAC will be held until the appropriate registers in the DAC are updated by the ISR.

The LEDs are lit according to the amplitude of the FIR result compared with the input value. Because the maximum value of can vary as the amplitude of the input varies, retaining a single maximum value throughout the program would not be helpful in determining what proportion the maximum value of the FIR result is of the input value. As a result, the values of the input and the output are stored in two buffers with a size of 128 words. A routine passes through each buffer to determine the maximum input and output value. The maximum value of the input and the output are zeroed every 500 interrupts thereby setting a new maximum value in case the input value changes in amplitude.

The DC offset is subtracted from each of the maximum values and then the input and output values are compared. By having a set of Q15 variables equivalent to (1/9, 2/9, 3/9,etc.), the input value is multiplied by each Q15 threshold and then compared to the output value. The LEDs are lit according to what proportion the output value exceeds the input. Because it would be inefficient to pass through all of the threshold values for each LED, the section which determines what LEDs to light starts with the middle threshold and, depending on whether the output value is larger or smaller than the middle threshold, determines whether the value is compared to the lower fourth or upper fourth threshold. This routine continues until the correct threshold is found and the corresponding LEDs are lit. Instead of passing through all eight thresholds, the worst case is comparing four thresholds; the best case two thresholds.

Once the FIR filter application is run, the LEDs will light according to whether the frequency is 500Hz and the input frequency stays below the maximum frequency (i.e. half the sampling frequency). Once the maximum frequency is exceeded, then the filter will respond with the maximum frequency as the folding frequency. Thus, the FIR filter will light the LEDs at 3500Hz, 4500Hz, 7000Hz, 8500Hz, and so forth.

Designing a Filter Using MATLAB® and Microsoft® Excel

MATLAB® Commands

The following commands will provide the b coefficients for the FIR filter

```
» clear;
```

Clears all the variables that have been used previously

```
» f=[0:1/20:1];
```

Defines a matrix called *f* from 0 to 1 in steps of 1/20 = 0.05

This matrix is the frequency matrix

```
» m=[0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0];
```

Defines a matrix call *m*

This matrix is the magnitude matrix of the filter

```
» b=fir2(100,f,m);
```

Defines a matrix *b* that will contain the values of the b coefficients of the FIR filter. This command uses the library function *fir2*. The first number defines the order of the filter.

```
» a=[1];
```

Defines a matrix *a* which contains the values of the a coefficient

```
» [h,q]=freqz(b,a,4000,4000);
```

Defines 2 matrices *h* and *q*. *h* will contain the frequency response of the filter and *q* will contain the corresponding frequency vector in Hertz.

This command uses the library function *freqz* that provides the frequency response of the desired filter. The third number indicates the number of points to be used, and the fourth number indicates the sampling frequency.

```
» plot(q,abs(h))
```

Plots the frequency response of the filter.

```
» xlabel('Frequency');
```

Labels the x-axis of the plot.

```
» title('nth Order FIR Filter with f Hz Sampling Frequency');
```

Gives the plot a title.

» grid on

Turns the grid of the plot on.

» b

Gives all the coefficients located in the matrix *b*.

Converting the Coefficients from MATLAB® Using Microsoft® Excel

1) Highlight and copy the coefficients from MATLAB®.
2) Paste the values into an Excel spreadsheet.
3) Parse the coefficients so that each coefficient has its own cell. Use the Text to Columns feature in the Data Menu.
4) Multiply each of the coefficients by 215.
5) Convert each of the products (coefficient × 215) to a hexadecimal value using the command DEC2HEX. Make sure that the Analysis ToolPak is loaded. To load the Analysis ToolPak, go to the Tools Menu and select Add Ins.
6) The resulting values from the DEC2HEX function are the equivalent Q15 values for each of the coefficients. The results can then be cut and pasted into the application.

```
;*******************************************************************
; File Name:     FIR0.ASM
; Originator:    Digital control systems Apps group - Houston
; Target Sys:    'C24x Evaluation Board
;
; Description:  FIR bandpass filter which detects the presence of a
;                500Hz signal.  If the tone is detected an LED is
;                lit by using the output port.  Sampling Frequency
;                forced to be 4kHz.
;
; Last Update:  9 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

;--------------------------------------------------------------------
; Variable Declarations 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    FIR_ISR  ; 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_INT10         B    PHANTOM  ; User S/W Interrupt
SW_INT11         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        ;DP = 224; Address for
                                     ;7000h - 707Fh

;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  #0000000011000011b,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 Enabled
;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 revsion 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             #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,CAPCON ;Clear Capture Control
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  2500
T1PERIOD         .set  5000    ;Sets up period for 4kHz frequency


LDP              #232          ;DP=232, Data Page for Event Manager
Addresses
SPLK             #T1COMPARE,T1CMPR;Compare value for 50% duty cycle

;                       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     ;Period value for 2kHz signal
SPLK        #0000h,T1CNT       ;Clear GP Timer 1 Counter
SPLK        #0000h,T2CNT       ;Clear GP Timer 2 Counter
SPLK        #0000h,T3CNT       ;Clear GP Timer 3 Counter


;                       5432109876543210
                 SPLK  #0001000000000010b,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  (00)  TCLD1,TCLD0 - Timer Compare Register Reload
;                   Condition
;                   When counter is 0
;Bit 1     (1)   TECMPR - Timer compare enable
;                   Enable timer compare operation
;Bit 0     (0)   Reserved

;                       5432109876543210
                 SPLK  #0000000000000000b,T2CON
                                            ;GP Timer 2 - 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
                                            ;GP Timer 3 - 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

;-----------------------------------
; Set up Digital I/O Port
;-----------------------------------
                 LDP   #225  ;DP=225, Data Page to Configure OCRA
;                       5432109876543210
                 SPLK  #0011100000001111b,OCRA

;OCRA  - Output Control Register A
;Bit 15          (0)   CRA.15 - IOPB7
;Bit 14          (0)   CRA.14 - IOPB6
;Bit 13          (1)   CRA.13 - T3PWM/T3CMP
;Bit 12          (1)   CRA.12 - T2PWM/T2CMP
;Bit 11          (1)   CRA.11 - T1PWM/T1CMP
;Bit 10          (0)   CRA.10 - IOPB2
;Bit 9           (0)   CRA.9 - IOPB1
;Bit 8           (0)   CRA.8 - IOPB0
;Bits 7-4        (0000)Reserved
;Bit 3           (1)   CRA.3 - ADCIN8
;Bit 2           (1)   CRA.2 - ADCIN9
;Bit 1           (1)   CRA.1 - ADCIN1
;Bit 0           (1)   CRA.0 - ADCIN0

;-----------------------------------
; Set up ADC Module
;-----------------------------------
                 LDP   #224

;                       5432109876543210
                 SPLK  #1000100100000000b,ADCTRL1

;ADCTRL1 - ADC Control Register 1
;Bit 15          (1)   Suspend-SOFT -
;                   Complete Conversion before halting emulator
;Bit 14          (0)   Suspend-FREE -
;                   Operations is determined by Suspend-SOFT
;Bit 13          (0)   ADCIMSTART - ADC start converting immediately
;                   No Action
```

```
;Bit 12           (0)   ADC2EN - Enable/Disable ADC2
;                    Disable ADC2
;Bit 11           (1)   ADC1EN - Enable/Disable ADC1
;                    Enable ADC1
;Bit 10           (0)   ADCCONRUN - ADC Continuous Conversion Mode
;                    Disable Continuous Conversion
;Bit 9            (0)   ADCINTEN - Enable ADC Interrupt
;                    Mask ADC Interrupt
;Bit 8            (1)   ADCINTFLAG - ADC Interrupt Flag
;                    Clear Interrupt Flag Bit
;Bit 7            (0)   ADCEOC - End of Conversion Bit  READ ONLY
;Bits 6-4         (000) ADC2CHSEL - ADC2 Channel Select
;                    Channel 8
;Bits 3-1         (000) ADC1CHSEL - ADC1 Channel Select
;                    Channel 0
;Bit 0            (0)   ADCSOC - ADC Start of conversion bit
;                    No Action

;                        5432109876543210
                  SPLK  #0000000000000101b,ADCTRL2

;ADCTRL2 - ADC Control Register 2
;Bits 15-11       (00000)Reserved
;Bit 10           (0)   ADCEVSOC - Event Manager SOC mask bit
;                    Mask ADCEVSOC
;Bit 9            (0)      ADCEXTSOC - External SOC mask bit
;                    Mask ADCEXTSOC
;Bit 8            (0)   Reserved
;Bits 7-6         (00)  ADCFIFO1 - Data Register FIFO1 Status  READ
ONLY
;Bit 5            (0)   Reserved
;Bits 4-3         (00)  ADCFIFO2 - Data Register FIFO2 Status  READ
ONLY
;Bits 2-0         (101) ADCPSCALE - ADC Input Clock Prescaler
;                    Prescale Value 16
;                    SYSCLK Period = 0.1usec
;                    0.1usec x 16 x 6 = 9.6 usec >= 6usec


;-----------------------------------
; Set up DAC Module
;-----------------------------------

;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
```

```
;--------------------------------------------------------------------
;  MAIN LINE
;--------------------------------------------------------------------

          .sect   ".blk0"

XVALUE    .word   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
          .word   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
          .word   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
          .word   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
          .word   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0


        .sect        ".blk1"
VALUEIN  .word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
         .word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
         .word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
         .word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
VALUEOUT .word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
         .word  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
         .word 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
         .word  0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

                 .data

;Coefficients for 500Hz Bandpass filter for 4kHz Sampling Frequency
BCOEFF           .word   0000h,0002h,0002h,0001h
                 .word   0000h,0000h,0000h,0FFFFh
                 .word   0FFFFh,0000h,0000h,0002h
                 .word   0002h,0FFFFh,0FFF9h,0FFF7h
                 .word   0000h,0013h,0025h,0021h
                 .word   0FFFBh,0FFBCh,0FF90h,0FFA7h
                 .word   0011h,00A4h,00FDh,00BFh
                 .word   0FFDEh,0FEC3h,0FE2Ah,0FEA7h
                 .word   0033h,0206h,02F1h,0220h
                 .word   0FFC2h,0FD19h,0FBD6h,0FD05h
                 .word   003Ch,03B8h,054Ah,03C5h
                 .word   0FFD4h,0FBB4h,0F9EAh,0FBADh
                 .word   0010h,0484h,0660h,0484h
                 .word   0010h,0FBADh,0F9EAh,0FBB4h
                 .word   0FFD4h,03C5h,054Ah,03B8h
                 .word   003Ch,0FD05h,0FBD6h,0FD19h
                 .word   0FFC2h,0220h,02F1h,0206h
                 .word   0033h,0FEA7h,0FE2Ah,0FEC3h
                 .word   0FFDEh,00BFh,00FDh,00A4h
                 .word   0011h,0FFA7h,0FF90h,0FFBCh
                 .word   0FFFBh,0021h,0025h,0013h
                 .word   0000h,0FFF7h,0FFF9h,0FFFFh
                 .word   0002h,0002h,0000h,0000h
                 .word   0FFFFh,0FFFFh,0000h,0000h
                 .word   0000h,0001h,0002h,0002h
                 .word   0000h
```

```
LEDS             .set  000Ch      ;I/O Address for LEDS register
WINDOW           .set  500        ;Number of smpls to check before
                                     ;reset’g MAX values
                 .bss  LEDSOUT,1  ;Variable for which LEDS to light
                 .bss  MAXIN,1    ;Maximum value input value
                 .bss  MAXOUT,1   ;Maxumum FIR result value
                 .bss  DIFFIN,1   ;Maximum Input Value - DC Offset
                                     ;(7ffh)
                 .bss  DIFFOUT,1  ;Maximum Output value - DC Offset
                                     ;(7ffh)
                 .bss  THRESHOLD1,1  ;Threshold value for 1st LED
                 .bss  THRESHOLD2,1  ;Threshold value for 2nd LED
                 .bss  THRESHOLD3,1  ;Threshold value for 3rd LED
                 .bss  THRESHOLD4,1  ;Threshold value for 4th LED
                 .bss  THRESHOLD5,1  ;Threshold value for 5th LED
                 .bss  THRESHOLD6,1  ;Threshold value for 6th LED
                 .bss  THRESHOLD7,1  ;Threshold value for 7th LED
                 .bss  THRESHOLD8,1  ;Threshold value for 8th LED

                 .bss  RESET_MAX,1   ;Counter to determine when to
                                        ;reset MAX values

                 .bss  TEMP,1        ;Variable for temporary storage
                                        ;of values

      .text

MAIN  LAR        AR1,#ADCFIFO1 ; AR1 = ADCFIFO1 address
      LAR        AR2,#ADCTRL1  ; AR2 = ADCTRL1 address
      LAR        AR3,#BCOEFF   ; AR3 = BCOEFF address
      LAR        AR5,#LEDS     ; AR5 = LEDS Output

                 LDP   #232
                 LACC  EVIFRA   ;ACC = Event Module Type A Interrupt
                                  ;Flags
                 SACL  EVIFRA   ;EVIFRA = ACC; Clears the current
                                  ;set flags
                 SPLK  #0080h,EVIMRA ; Enable Timer 1 Period
                                        ; Interrupt

                 MAR   *,AR2         ;ARP = AR2
                 LACC  *             ;ACC = ADCTRL1
                 ADD   #1            ;SET BIT FOR SINGLE CONVERSION
                 SACL  *,0,AR1       ;STARTS ADC CONVERSION

                 SBIT1 T1CON,B6_MSK  ;Sets Bit 6 of T1CON; Starts
                                        ;the timer

                 LDP   #0         ;DP = 0; Addresses 0000h - 007Fh
                 SPLK  #0000h,LEDSOUT   ;Clear the LEDS
                 OUT   LEDSOUT,LEDS
                 SPLK  #0E38h,THRESHOLD1;Q15 value for 1/9
                 SPLK  #1C71h,THRESHOLD2;Q15 value for 2/9
                 SPLK  #2AAAh,THRESHOLD3;Q15 value for 3/9
                 SPLK  #38E3h,THRESHOLD4;Q15 value for 4/9
                 SPLK  #471Ch,THRESHOLD5;Q15 value for 5/9
                 SPLK  #5555h,THRESHOLD6;Q15 value for 6/9
                 SPLK  #638Eh,THRESHOLD7;Q15 value for 7/9
                 SPLK  #71C7h,THRESHOLD8;Q15 value for 8/9

                 SPLK  #0000h,MAXIN  ;Initialize Maxmimum input
                                        ;value
                 SPLK  #0000h,MAXOUT ;Initialize Maximum FIR output
                                        ;value

                 SPLK  #WINDOW,RESET_MAX   ;Initialize the maximum
                                              ;reset counter

                 CLRC  INTM                ;Enable Interrupts

                 WAIT  B  WAIT             ;Wait for interrupt

;--------------------------------------------------------------------
;  INTERRUPT SERVICE ROUTINES FOR FIR FILTER
;--------------------------------------------------------------------

FIR_ISR          LAR  AR4,#XVALUE+100      ;AR4 = DATA ADDRESS
                 MAR   *,AR1               ;ARP = AR1 = ADCFIFO1
                 LACC  *,0,AR4             ;ACC = ADCFIFO1; ARP =
AR4

                 LDP   #0      ;DP = 0
                                  ;Addresses 0000h - 007Fh
                 SACL  DAC1VAL ;DAC1VAL = ADCFIFO1

                 RPT   #7      ;Shift ADC value 8 places
                                  ; - Reduce to 8 bit value
                 SFR           ;Larger bit values produced
                                  ;large results
                 SUB   #7Fh    ;Subtract the equivalent 8 bit
                                  ;DC offset

                 LDP   #04h    ;DP = 4; Address 0200h - 027Fh
                 SACL  XVALUE  ;XVALUE = ADCFIFO1 / 256;

                 LACC  #0h     ;Initialize the ACCUMULATOR
                 MPY   #0h     ;Initialize the PROD REG

                 RPT   #100    ;Calculate Y
                 MACD  BCOEFF,*-     ;Multiply X with B, and add
                 APAC          ;final accumulation
```

```
                  LDP   #0
                  RPT   #7          ;Shift the result 8 places to left
                  SFL
                  SACH  DAC0VAL,1   ;DAC0VAL = Y * 2; shift to
                                       ;remove extra sign bit
                                       ;FIR result to output

;Multiply the values by 5/4 because the maximum gain is 4/5

                  LT DAC0VAL        ;TREG = DAC0VAL
                  MPY   #5          ;PREG = DAC0VAL * 5
                  PAC               ;ACC = PREG = DAC0VAL * 5
                  SFR               ;ACC = DAC0VAL * 5 / 2
                  SFR               ;ACC = DAC0VAL * 5 / 4
                  SACL  DAC0VAL     ;DAC0VAL = DAC0VAL * 5/4

                  LACC  DAC0VAL     ;ACC = DAC0VAL
                  RPT   #3          ;Shift right 4 times
                                       ; = 16 bit value to
                  SFR               ;12 bit value because
                                       ;DAC is 12bits
                  ADD   #7FFh       ;Add DC offset
                  AND   #0FFFh      ;Ensure 12 bits
                  SACL  DAC0VAL     ;Store value for output on the DAC
                  LDP   #7          ;DP=7; Address for 0380h to 03FFh
                  SACL  VALUEOUT    ;Store value to find maximum
                                       ;value of the output values

                  LAR   AR6,#(VALUEOUT+127-1)  ;AR6 = End of VALUE OUT
                                             ;buffer
                  LAR   AR7,#126    ;AR7 = 127 - 1; Number of
                                       ;values to move
                  MAR   *,AR6       ;ARP = AR6
SHIFT1            DMOV  *-,AR7      ;Move all of the values in
                                       ;the VALUEOUT
                  BANZ  SHIFT1,*-,AR6    ;Data Buffer to the next
                                             ;higher address



                  LDP   #0          ;DP = 0; Addresses 0000h - 007Fh
                  LACC  DAC1VAL     ;ACC = DAC1VAL = Input Value
                  RPT   #3          ;Shift the value to the
                                       ;right 4 times
                  SFR               ;Convert the value from 16
                                       ;bits to 12 bits
                  SACL  DAC1VAL     ;DAC1VAL = 12 bit value for DAC
                  LDP   #6          ;DP = 6; Addresses 0300h - 037Fh
                  SACL  VALUEIN     ;VALUEIN = DAC1VAL

                  LAR   AR6,#(VALUEIN+127-1);AR6 = End of VALUE IN
                                                ;buffer
```

```
                 LAR   AR7,#126    ;AR7 = 127 - 1; Number of
                                      ;values to move
                 MAR   *,AR6       ;ARP = AR6
SHIFT2           DMOV  *-,AR7      ;Move all of the values in
                                      ;the VALUEIN
                 BANZ  SHIFT2,*-,AR6 ;Data Buffer to the next
                                         ;higher address


;Outputs the FIR results and the original value
;DAC0 has the FIR results and DAC1 has the original value

                 LDP   #0
                 OUT   DAC0VAL,DAC0  ;DAC0 = DAC0VAL; FIR result on
                                         ;DAC channel 0
                 OUT   DAC1VAL,DAC1  ;DAC1 = DAC1VAL; Input value
                                         ;on DAC channel 1
                 OUT   DAC0VAL,DACUPDATE
                                     ;Update the values on the DAC


;Find the maximum value among VALUEIN and VALUEOUT for the LEDs

                 LACC  RESET_MAX   ;ACC = RESET_MAX
                                      ; Max Reset Counter
                 SUB   #1          ;Decrement by 1
                 SACL  RESET_MAX   ;Store new value for RESET_MAX
                 BCND  NO_RESET,GT    ;If not WINDOWth value, don’t
                                         ;reset counter

                 SPLK  #WINDOW,RESET_MAX
                                  ;Else reset the max reset counter
                 SPLK  #0000h,MAXIN  ;Reset the MAXIN value
                 SPLK  #0000h,MAXOUT ;Reset the MAXOUT value

NO_RESET         LAR   AR6,#VALUEIN  ;AR6 = VALUEIN; Beginning of
                                        ;Data In Buffer
                 LAR   AR7,#127      ;AR7 = 128 - 1; Counter to find
                                        ;max value in

                 MAR   *,AR6         ;ARP = AR6
FIND_MAXIN       LACC  *+,0,AR7      ;ACC = Value pointed by AR6
                 SUB   MAXIN         ;Subtract MAXIN
                 BCND  RESUME1,LEQ   ;If the value results in a
                                        ;value less than 0,
                                        ;then the value is smaller
                                        ;than MAXIN, else the
                                        ;value is larger than MAXIN
                 ADD   MAXIN         ;ACC = Value pointed by AR6
                 SACL  MAXIN         ;Store new MAXIN value

RESUME1          BANZ FIND_MAXIN,*-,AR6    ;If smaller than MAXIN,
                                     ;decrement loop counter
                                     ;(AR7), move to next value in
                                     ;buffer


                 LAR   AR7,#127   ;Since VALUEIN buffer is
                                     ;adjacent to
                                     ;VALUEOUT buffer, only AR7
                                     ;needs to be reset

                                     ;ARP is already AR6
FIND_MAXOUT      LACC  *+,0,AR7      ;ACC = Value pointed by AR6
                 SUB   MAXOUT        ;Subtract MAXOUT
                 BCND  RESUME2,LEQ;If the value results in a
                                     ;value less than 0,
                                     ;then the value is smaller than
                                     ;MAXOUT, els
                                     ;the value is larger than
                                     ;MAXOUT
                 ADD   MAXOUT     ;ACC = Value pointed by AR6
                 SACL  MAXOUT     ;Store new MAXOUT value

RESUME2          BANZ  FIND_MAXOUT,*-,AR6  ;If smaller than MAXOUT,
                                     ;dec loop counter (AR7),
                                     ;move to next value in buffer

;The following section determines if the value meets the threshold
;requirement

                 LDP   #0         ;DP = 0; Addresses 0000h to 007Fh
                                     ;All variables used are in B2


;Need to remove the DC offset because if the FIR result is 0 it will
;equal 7ffh which is already 50% of the maximum input value

                 LACC  MAXIN            ;ACC = MAXIN
                 SUB   #7FFh            ;Subtract the DC offset
                 SACL  DIFFIN           ;DIFFIN = MAXIN - 7ffh

                 LACC  MAXOUT           ;ACC = MACOUT
                 SUB   #7FFh            ;Subtract the DC offset
                 SACL  DIFFOUT          ;DIFFOUT = MAXOUT - 7ffh

;Check if the output exceeds the middle threshold value, THRESHOLD4
                 LT DIFFIN           ;TREG = DIFFIN
TH4              MPY   THRESHOLD4    ;PREG = DIFFIN * THRESHOLD4
                 PAC                 ;ACC = PREG
                 SACH  TEMP,1        ;TEMP = ACC*2; Shift to remove
                                        ;extra sign bit
                 LACC  TEMP          ;ACC = TEMP
                 SUB   DIFFOUT       ;Subtract DIFFOUT
                 BCND  ABOVE4,LT   ;If DIFFOUT is greater than
                              ;TEMP, then the FIR result is
                              ;greater than VALUEIN * THRESHOLD4,
                              ;else, it is below THRESHOLD4 value

;Output is below THRESHOLD4.  Check if above THRESHOLD2
BELOW4           LT    DIFFIN
TH2              MPY   THRESHOLD2
                 PAC
                 SACH  TEMP,1
                 LACC  TEMP
                 SUB   DIFFOUT
                 BCND  ABOVE2,LT

;Output is below THRESHOLD4 & THRESHOLD2.  Check if above THRESHOLD1
BELOW2           LT    DIFFIN
TH1              MPY   THRESHOLD1
                 PAC
                 SACH  TEMP,1
                 LACC  TEMP
                 SUB   DIFFOUT
                 BCND  ABOVE1,LT

;Output is below THRESHOLD4, THRESHOLD2, & THRESHOLD1. Turn off LEDS
BELOW1           SPLK  #0000h,LEDSOUT
                 B     OUTLEDS

;Output is below THRESHOLD4, THRESHOLD2, but above THRESHOLD1. Turn
;on DS1
ABOVE1           SPLK  #0001h,LEDSOUT
                 B     OUTLEDS

;Output is below THRESHOLD4, but above THRESHOLD2.  Check if above
;THRESHOLD3
ABOVE2           LT    DIFFIN
TH3              MPY   THRESHOLD3
                 PAC
                 SACH  TEMP,1
                 LACC  TEMP
                 SUB   DIFFOUT
                 BCND  ABOVE3,LT

;Output is below THRESHOLD4 and THRESHOLD3, but above THRESHOLD2.
;Turn on DS1-DS2
BELOW3           SPLK  #0003h,LEDSOUT
                 B     OUTLEDS

;Output is below THRESHOLD4, but above THRESHOLD3 and THRESHOLD2.
;Turn on DS1-DS3
ABOVE3           SPLK  #0007h,LEDSOUT
                 B     OUTLEDS
```

```
;Output is above THRESHOLD4.  Check if above THRESHOLD6
ABOVE4            LT    DIFFIN
TH6               MPY   THRESHOLD6
                  PAC
                  SACH  TEMP,1
                  LACC  TEMP
                  SUB   DIFFOUT
                  BCND  ABOVE6,LT

;Output is above THRESHOLD4, but below THRESHOLD6.  Check if above
;THRESHOLD5.
BELOW6            LT    DIFFIN
TH5               MPY   THRESHOLD5
                  PAC
                  SACH  TEMP,1
                  LACC  TEMP
                  SUB   DIFFOUT
                  BCND  ABOVE5,LT

;Output is above THRESHOLD4, but below THRESHOLD6 & THRESHOLD5. Turn
;on DS1-DS4
BELOW5            SPLK  #000Fh,LEDSOUT
                  B     OUTLEDS

;Output is above THRESHOLD4 & THRESHOLD5, but below THRESHOLD6.
;Turn on DS1-DS5
ABOVE5            SPLK  #001Fh,LEDSOUT
                  B     OUTLEDS

;Output is above THRESHOLD4 & THRESHOLD6. Check if above THRESHOLD8.
ABOVE6            LT    DIFFIN
TH8               MPY   THRESHOLD8
                  PAC
                  SACH  TEMP,1
                  LACC  TEMP
                  SUB   DIFFOUT
                  BCND  ABOVE8,LT

;Output is above THRESHOLD4 & THRESHOLD6, but below THRESHOLD8.
;Check if above THRESHOLD7.
BELOW8            LT    DIFFIN
TH7               MPY   THRESHOLD7
                  PAC
                  SACH  TEMP,1
                  LACC  TEMP
                  SUB   DIFFOUT
                  BCND  ABOVE7,LT

;Output is above THRESHOLD4 & THRESHOLD6, but below THRESHOLD8 &
;THRESHOLD7. Turn on DS1-DS6
BELOW7            SPLK  #003Fh,LEDSOUT
                  B     OUTLEDS
```

```
;Output is above THRESHOLD4, THRESHOLD6, & THRESHOLD7, but below
;THRESHOLD8. Turn on ;DS1-DS7
ABOVE7           SPLK  #007Fh,LEDSOUT
                 B     OUTLEDS

;Output is above THRESHOLD4, THRESHOLD6, & THRESHOLD8.  Turn on
;DS1-DS8
ABOVE8           SPLK  #00FFh,LEDSOUT

OUTLEDS          OUT   LEDSOUT,LEDS  ;Turn on the LEDS

RESTART_ADC      MAR   *,AR2         ;ARP = AR2
                 LACC  *             ;ACC = ADCTRL1
                 ADD   #1h           ;Set bit to restart the ADC
                 SACL  *             ;Start converting next value

                 LDP   #232
                 LACC  EVIFRA        ;Clear the flag register of
                                        ;Event Manager
                 SACL  EVIFRA

                 CLRC  INTM          ;ENABLE INTERRUPTS
                 RET                 ;Return to main line

;====================================================================
; I S R  -  PHANTOM
;
; Description:   Dummy ISR, used to trap spurious interrupts.
;
; Modifies:
;
; Last Update:   16-06-95
;====================================================================

PHANTOM         B       PHANTOM
```