```
//###########################################################################
//
// FILE:    Flax_DelfinoEvbGpioToggle.c
//
// TITLE:   DSP2833x Device GPIO toggle test program.
// This template was written by Eli Flaxer for DelfinoEvb Evaluation Board
//
//
//###########################################################################
// $TI Release: 2833x/2823x Header Files and Peripheral Examples V133 $
// $Release Date: June 8, 2012 $
//###########################################################################

#include "DSP28x_Project.h"     // Device Header file and Examples Include File
#include "LCD2x16Display.h"

void DelfinoEvbGpioSelect(void);
void MyMainProg(void);

int32 MyDelayLoop = 200000L;

/*****************************************************************************/
void GpioCSetClear(int k,int x)
{
    if (x)
        GpioDataRegs.GPCSET.all = (1L<<k);
    else
        GpioDataRegs.GPCCLEAR.all = (1L<<k);
}
/*****************************************************************************/
void main(void)
{

// Step 1. Initialize System Control:
// PLL, WatchDog, enable Peripheral Clocks
// This example function is found in the DSP2833x_SysCtrl.c file.
   InitSysCtrl();

// Step 2. Initalize GPIO:
// This example function is found in the DSP2833x_Gpio.c file and
// illustrates how to set the GPIO to it's default state.
// InitGpio();  // Skipped for this example

// For this example use the following configuration:
   DelfinoEvbGpioSelect();

// Step 3. Clear all interrupts and initialize PIE vector table:
// Disable CPU interrupts
   DINT;

// Initialize PIE control registers to their default state.
// The default state is all PIE interrupts disabled and flags
// are cleared.
// This function is found in the DSP2833x_PieCtrl.c file.
   InitPieCtrl();

// Disable CPU interrupts and clear all CPU interrupt flags:
```

```
   IER = 0x0000;
   IFR = 0x0000;

// Initialize the PIE vector table with pointers to the shell Interrupt
// Service Routines (ISR).
// This will populate the entire table, even if the interrupt
// is not used in this example.  This is useful for debug purposes.
// The shell ISR routines are found in DSP2833x_DefaultIsr.c.
// This function is found in DSP2833x_PieVect.c.
   InitPieVectTable();


// Step 4. Initialize all the Device Peripherals:
// This function is found in DSP2833x_InitPeripherals.c
// InitPeripherals(); // Not required for this example

// Step 5. User specific code:

    // This example uses DATA, SET, CLEAR and TOGGEL registers to toggle I/O's
   MyMainProg();
}
/*****************************************************************************/
void MyMainProg(void)
{
   GpioDataRegs.GPASET.bit.GPIO27 = 1;      // Turn on the buzzer
   DELAY_US(200000);
   GpioDataRegs.GPACLEAR.bit.GPIO27 = 1;    // Turn off the buzzer

//   BackLightLCD(1);
//   InItLCD();
//   DELAY_US(10000);
//   ClearLCD();
//   PrintLCD("AFEKA LAB");

   while(1)
       {
// Toggle I/Os using DATA, SET, CLEAR and TOGGEL registers
       GpioDataRegs.GPADAT.all    = 0xA;            // DAT Register (hazardous !!!)
       GpioDataRegs.GPCCLEAR.all  = 0x00000F00;     // CLEAR Register
       GpioDataRegs.GPCTOGGLE.all = 0x0000F000;     // TOGGEL Register
       DELAY_US(MyDelayLoop);

       GpioDataRegs.GPADAT.all    = 0x5;            // DAT Register (hazardous !!!)
       GpioDataRegs.GPCSET.all    = 0x00000F00;     // SET Register
       GpioDataRegs.GPCTOGGLE.all = 0x0000F000;     // TOGGEL Register
       DELAY_US(MyDelayLoop);

// Reading the switches
       GpioDataRegs.GPCDAT.bit.GPIO64 = GpioDataRegs.GPADAT.bit.GPIO8;  //(hazardous !!!)
       GpioDataRegs.GPCDAT.bit.GPIO65 = GpioDataRegs.GPADAT.bit.GPIO9;  //(hazardous !!!)
       GpioCSetClear(2, GpioDataRegs.GPADAT.bit.GPIO10);    // No Risk !!!
       GpioCSetClear(3, GpioDataRegs.GPADAT.bit.GPIO11);    // No Risk !!!
       }
}
/*****************************************************************************/
void DelfinoEvbGpioSelect(void)
{
```

```
    EALLOW;

    GpioCtrlRegs.GPAMUX1.all = 0x00000000;      // All GPIO
    GpioCtrlRegs.GPAMUX2.all = 0x00000000;      // All GPIO

    GpioCtrlRegs.GPADIR.all = 0x0000000F;       // Outputs 4 Leds
    GpioCtrlRegs.GPADIR.bit.GPIO27 = 1;         // Buzzer
    GpioCtrlRegs.GPBDIR.all = 0x07FF0000;       // Outputs LCD 8 Bus 3 Control
    //GpioCtrlRegs.GPBDIR.all = 0x07FF??00;     // Extended Bus Direction GPIO40-GPIO47 KB
    GpioCtrlRegs.GPCDIR.all = 0x0000FFFF;       // Outputs 8 Leds 4 TP 4 TestLed
    //GpioCtrlRegs.GPCDIR.all = 0x000?FFFF;     // Extended Bus Direction GPIO80-GPIO83 Button

    //GpioCtrlRegs.GPBPUD.all = 0x0000FF00;     // Extended Bus Pull-Up Resistors

    EDIS;
}
/*****************************************************************************/
```