Programming Assignment 5: Capacitive Sensing
Due March 2, 2010

Associated Cypress Board: CY3214
Associated Part Family: CY8C24894-24LFXI
PSoC Designer Version: 5.0

Summary
You will implement a system which senses finger touching and plays a song. The result is displayed as both visual (LCD) and audio (Loud Speaker) as follows.

- Touch P37 with your finger and watch “S0 Touched!” message on LCD screen and hear C4_note from LS1.

- Touch P30 with your finger and watch “S1 Touched!” message on LCD screen and hear D4_note from LS1.

- Touch both P30 and P37 simultaneously and watch “S0 & S1 Touched!” message on LCD screen and hear NO sound (You should touch both at the exactly same time. Otherwise you may hear sound either C4 or D4 note.).

- Touch and slide the P57-P56 slider and watch a bar graph following your finger movement on LCD screen and hear a note ranging from C4 to C5.

Setup
Open PSoC Designer. Click on File -> New Project. [New Project] window opens. Select Chip-level Project and Type your project name in the Name box.

Click Browse to select the directory where you are storing your Lab assignments. Click the check box to create directory for workspace. Click OK.

In the next [Select Project Type] window, select part CY8C24894 through the View Catalog… button and choose to Generate Main file using C. Click OK when done.

Adding User Module(s) & Connecting Port/Pin
Now add the user modules: LCD, CSD, and PWM8. In the User Module Catalog, expand Misc Digital, Right Click on LCD and select Place. Now you should see one user modules added in the Workspace Explorer as “LCD_1”. Open property window of LCD_1 and connect it to P4[0].

Next one is CSD. In the User Module Catalog, go to Cap Sensors -> CSD -> CSD with PRS16 as clock source and place it.

In the Workspace explorer right-mouse click "CSD_1". In the drop down menu click CSD Wizard. In the CSD Wizard window, do the following:
1. N switches: 2
2. N sliders: 1
3. Drag SW0 to P3[7]
4. Drag SW1 to P3[0].

Select the Slider and set:
1. N Sensor: 8
2. Resolution: 5
3. Diplex: false
4. Drag S0(0) to P5[7]
5. Drag S0(1) to P5[5]
6. Drag S0(2) to P5[3]
7. Drag S0(3) to P5[1]
8. Drag S0(4) to P5[0]
9. Drag S0(5) to P5[2]
10. Drag S0(6) to P5[4]
11. Drag S0(7) to P5[6]

In the Workspace explorer right-mouse click "CSD_1" and select properties. Change the following parameters as follows:
1. Scanning Speed: Fast
2. Resolution: 9
3. Modulator Cap: P0[5]

Next one is PWM8. In the User Module Catalog, go to PWMs, select PWM8 and place it. In the Workspace explorer right-mouse click "PWM8_1" and select properties. Set the parameters as follows:
1. clock: cpu_32_KHz
2. Enable: High
3. CompareOut: Row_0_Output_0
4. TerminalCountOut: None
5. Period: 0
6. PulseWidth: 0
7. CompareType: Less Than or Equal
8. InterruptType: Terminal Count
9. ClockSync: Sync to Sysclk
10. InvertEnable: Normal

Interconnection on [Chip] editor window
On the [Chip] editor window, connect Row_0_Output_0 to GOE_0(by left-mouse clicking the white box of RO0[0] first, then clicking the top triangle connected to GOE0). Then connect GOE_0 to Port_0_0(by left-mouse clicking the GOE0 wire first, then setting the parameters as follows: Pin: Port_0_0, Interconnect: InputToOutput).
Adding Software

Now that you have the hardware hooked up, you can write your embedded C program. As you know by this time, in the Workspace Explorer, expand the Project_name Folder and Source Files Tab and double click on Main.c.

**NOTE:** Use the provided fixed pulse width for the PWM. Change only the period of the PWM.

**HINT:** The PWM runs off a 32kHz clock. It will generate a pulse at a fraction of this clock depending on what’s put into the WritePeriod() function, i.e., if you write 100, you will have 100 clock ticks (32kHz) between pulse outputs, giving you a frequency 100th of the original 32kHz clock. Also, be sure to refer to all datasheets for any questions.

```c
#include <m8c.h> // part specific constants and macros
#include "PSoCAPI.h" // PSoC API definitions for all User Modules

#define PULSE_WIDTH 20
#define CLOCK 32000
#define C4_NOTE 261.63
#define D4_NOTE 293.66
#define E4_NOTE 329.63
#define F4_NOTE 349.23
#define G4_NOTE 392
#define A4_NOTE 440
#define B4_NOTE 493.88
#define C5_NOTE 523.25

int frequency;

void main()
{
    M8C_EnableGInt; // Enables interrupts, you'll learn about these in future courses.

    // Start LCD and CSD
    // Init LCD position and display “UCSD CSE30 09”
    // Scan all sensors first time, init baseline
    // Set to default finger thresholds
    while (1) {
        // Hint: You should use the following APIs. Look at their usage in the data sheet
        LCD_1_Start
        CSD_1_Start
        CSD_1_InitializeBaselines
        CSD_1_ScanAllSensors
        CSD_1_SetDefaultFingerThresholds
        CSD_1_UpdateAllBaselines
        LCD_1_Position
        CSD_1_bIsAnySensorActive
        CSD_1_bIsSensorActive
        LCD_1_PrCString
        PWM8_1_WritePeriod
    }
}
```
PWM8_1_WritePulseWidth
PWM8_1_Start
LCD_1_InitBG
LCD_1_DrawBG
PWM8_1_Stop

e.g., if ( CSD_1_bIsSensorActive(0) ) means that the first sensor(P37) is touched.
} // while
} //main

**Wiring (if necessary) and Test Your Design**
P00 - LS1 (for loud speaker audio), P05 - P15(for CSD sensing)