OPTIMIZING PERFORMANCE VERSUS POWER CONSUMPTION IN CAPACITIVE TOUCH SENSING DESIGNS

By Pushkar Nandkar, and Bala Satish Gurlinka, Cypress Semiconductor Corp.

Capacitive sensing has replaced mechanical buttons in a wide range of consumer appliances, including washing machines, music players, and mobile phones, to name a few. This has been possible largely because capacitive sensors are durable, more reliable, and provide better aesthetics through their simple yet elegant user interface that supports multiple functionalities at literally the tip of a finger. The quality of a capacitive sensing implementation, however, depends upon its response time and power consumption. Highly responsive systems give a real-time feel while reducing power consumption improves operating life in the case of battery-based devices. In this article, we discuss some standard capacitive sensing elements and applications with a focus on balancing power consumption and response time for embedded engineers developing capacitive sensing interfaces.

**Basic principle of physics:**

Capacitive sensing can sense the movement of any conductor. The human body being a conductor, capacitive sensing can be used to detect the presence and absence of a finger or trace the movement of a finger. This gives an opportunity for designers to have human finger driven GUI (Graphical User Interface) menus, adding aesthetic value to their designs.

The change in capacitance due to the presence of a finger, or any conducting material for that matter, is measured using a capacitive sensor. A capacitance sensor would typically be a metal pad separated from the ground as shown in Figure 1. The Electric Field lines follow the path of least resistance which start from the sensor and end in the ground plane.

![Electric Field Distribution in presence and absence of a finger.](image)

**Figure 1: Electric Field Distribution in presence and absence of a finger.**
The introduction of a finger would form a capacitor, the finger being one plate, the metal pad being the other and air serving as the dielectric. It can be termed as finger capacitance $C_F$. Before the introduction of the finger there would be some parasitic capacitance $C_P$ between the metal pad and ground. The finger capacitance adds in parallel to $C_P$ and net capacitance becomes $C_P + C_F$.

**Capacitive Sensing Methods:**

There are many capacitive sensing controllers available on the market today using various sensing algorithms such as the relaxation oscillator method, methods based on switched capacitor (SC) front-ends, mutual capacitance sensing, phase delay and amplitude measurement, etc. One important aspect for overcoming noise in capacitive systems is to have a robust sensing method which has low input impedance. The two most prevalent methods are:

1) Relaxation oscillator method (RO method) and its variants – The RO method contracts the oscillator using the capacitance of sensor and the variation in frequency is converted into counts.

2) Switched capacitance front-end (SC method) – The SC method converts the capacitance to counts by charging or discharging a reference capacitor with the sensor capacitance at a high frequency. Switched capacitor methods have a lower input impedance than relaxation oscillator methods and so provide better noise immunity.

**Balancing power consumption and response time**

With advances in technology, more electronic products are becoming battery-operated and power consumption has become a critical factor in design. With this in mind, system designer need to adjust system parameters so that system functionality is provided with the minimum active current.

When doing a complete system design for an application, care should be taken to estimate the power consumption of the system as a whole instead of the controller alone. For capacitive sensing, the total power consumption of the controller and the external circuitry should be considered.

**External circuit:** The aim should be to avoid any voltage difference between resistors. This is controller independent and depends mainly on the circuit design. A typical circuit using a capacitive sensor is shown below.
Figure 3 shows a Cypress CapSense device with three capacitive sensing buttons and three indicator LEDs. For example, in sleep mode, P0[5]-P0[7] and P1[0],P1[1] in the figure above are driven to high impedance state and P0[0]-P0[2], which are connected to capacitive sensing pads, are set to CMOS drive with a logical 0 driven to the port pins.

**Controller power consumption**

This mainly depends upon the efficiency of the controller. For a System-on-Chip (SoC), this depends on the number of resources active at that particular instance. Sleep mode is an effective method for extending battery life in capacitive sensing applications. Important system parameters that affect battery life are the supply voltage, CPU speed, and the active and sleep time of the controller. This section discusses how to minimize active current and incorporate sleep mode into button sensing applications.

Power consumption is reduced by turning off unused resources or putting them to sleep. Resources can be the controller itself or external peripherals and circuit elements. In case of an SoC, individual analog and global resources can also be put to sleep.

Sleep current values are typically much less compared to active current values. Thus, when a system is not being used, it can be put into sleep for idle time so that the average current consumption is reduced.
The above equation shows how the I_{average} can be minimized to these requirements. The appropriate way to do that is minimize I_{active}, T_{active} and I_{sleep} and maximize T_{sleep}. However, this will affect system response time.

**Response Time**

The response time of a system is defined as the time after which a valid output is given by the system after an input has been provided. The controller goes into sleep mode and turns on the sleep timer. At periodic intervals, the controller then wakes up and scans the input for any activity. Suppose an input occurs while the controller is in sleep mode. The controller can only sense the input only after it wakes up. Thus, the worst case response time of the system becomes the sleep timer interval. The graph in Figure 5 shows a typical response time vs. current consumption curve. Developers can choose a response time and current consumption depending on the specific application requirements.
As per above the discussion, if the sleep timer interval is increased, the response time is degraded. However, the average current consumption also decreases according to eq 1. So when designing an application, one should arrive at a tradeoff between power consumption and response time. Here we discuss an example of a 3-button capacitive sensing system that illustrates how to balance power consumption and response time.

Consider a Bluetooth headset having three capacitive sensing buttons for answer/disconnect, Volume Up and Volume down. For a person using such a Bluetooth handset, assume the following usage profile:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Calls/day</td>
</tr>
<tr>
<td>2</td>
<td>Usage of attend/disconnect button</td>
</tr>
<tr>
<td>3</td>
<td>Usage of Volume button</td>
</tr>
<tr>
<td>4</td>
<td>Miscellaneous Use</td>
</tr>
<tr>
<td>5</td>
<td>Button usage per call</td>
</tr>
<tr>
<td>6</td>
<td>Button usage per day</td>
</tr>
<tr>
<td>7</td>
<td>Total seconds in a day</td>
</tr>
<tr>
<td>8</td>
<td>Percentage usage of Capacitive Buttons</td>
</tr>
<tr>
<td>9</td>
<td>Scan Time for Each sensor</td>
</tr>
<tr>
<td>10</td>
<td>Scan Time for Three sensor</td>
</tr>
</tbody>
</table>

Figure 5: Average Current vs. Scan Rate
### Optimizing performance vs. power consumption in your capacitive touch sensing design

#### Published in Embedded.com (http://www.Embedded.com)  May 2011

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Firmware Overhead</strong></td>
<td>2ms</td>
<td></td>
</tr>
<tr>
<td><strong>Total Active Time</strong></td>
<td>4.25ms</td>
<td></td>
</tr>
<tr>
<td><strong>Scanning Interval without finger press</strong></td>
<td>500ms</td>
<td></td>
</tr>
<tr>
<td><strong>Scanning Interval with finger press</strong></td>
<td>20ms</td>
<td></td>
</tr>
<tr>
<td><strong>Active Current</strong></td>
<td>4mA</td>
<td></td>
</tr>
<tr>
<td><strong>Sleep Current</strong></td>
<td>1uA</td>
<td></td>
</tr>
</tbody>
</table>
| **Average Current when finger not present** | \[
\frac{(500 - 4.25) \times 10^{-6} + (4.25) \times 4 \times 10^{-3}}{500} \times \frac{100 - 1.7}{100} \\
= 34.4 \text{ uA}
\] |
| **Average Current when finger present** | \[
\frac{(20 - 4.25) \times 10^{-6} + (4.25) \times 4 \times 10^{-3}}{20} \times \frac{1.7}{100} \\
= 14.46 \text{ uA}
\] |
| **Average Current Consumption** | 48.86uA |

In this way, power consumption for the design may be generated to choose the best values that fit the design, depending on the specifications of the controller being used.

**Techniques specific to capacitive sensing for reducing power consumption:**

As discussed earlier, for reducing power consumption in capacitive sensing solutions, a sleep timer can be implemented to make the controller sleep and wake up periodically and scan the capacitive sensor I/Os to check for user activity. Optimal settings should be selected so that users do not have slow response and therefore a poor use experience.

Once any activity is detected, it is necessary to set up a ‘time without activity’ check while in normal operation that would wait some time interval before going to sleep mode. This would make sure that the normal operation routine is not constantly going to sleep after a calculation. The advantage of this would be that the system could likely have longer sleep duration during idle mode with minimal effect on the user experience. Here are some techniques specific to reducing power consumption in capacitive sensing systems.

**Gang Sensing:** As the number of capacitive sensors increases, for a constant scan rate, power consumption increases. In a similar way, for the same number of sensors, if the scanning rate is increased then power consumption increases. Decreasing the number of sensors basically means reducing the number of sensors available for scan. It is possible to combine all the sensors and scan them only once. This is called Gang Capacitive Sensing. The gang is considered as a single sensor and the capacitive sensing algorithm scans only one sensor. Once activity is detected and confirmed, the sensors are disconnected and scanned individually.

**CPU utilization during scanning:** While scanning a sensor, the CPU keeps waiting for the maximum time to let the hardware finish the job of scanning. This idle CPU time may be utilized to do other necessary system computations. For example, if a software filter is applied to sensor data, while scanning the second sensor, the filter computation of the first sensor can be completed in parallel. This reduces the CPU active time, in turn reducing power consumption.

**Slowing the CPU during scanning:** While the CPU waits for scanning to complete, the CPU frequency is reduced to its lowest speed to reduce the active current. However, there is a problem when an interrupt is posted and serviced, since the CPU time for this will increase. This can be countered by increasing the CPU frequency in the Interrupt Service Routine.

**Use partial sensing:** If ganging sensors is not possible, scan the minimum number sensors to check for any activity. For instance, in a track pad or slider design, one obviously would not need to check every sensor to look for activity. A subset could be checked to register any activity.

**Use proximity sensing:** Instead of scanning all the sensors, every time the controller wakes up, only a proximity sensor is scanned. In the case of any activity, the other sensors can be scanned.
Various types of Capacitive Sensors and related power consumption issues:

Capacitive sensors such as buttons, sliders, and proximity sensors can be implemented in an embedded design. The type of the sensor varies depending on their use and the applications where they fit in. Selecting the right sensors for an application gives a better feel to the user. However, there are some power consideration issues that need to be considered.

Buttons

Capacitive sensing buttons can be used to implement the ON/OFF switch. Some examples are the user interfaces of TVs, monitors, audio systems, photo frames, home security systems, washing machines, MP3 players, refrigerators, remote controls, and microwave ovens. Matrix pattern buttons are used in applications such as calculator keypad, QWERTY keyboards, etc.

Sliders

Sliders are used for controls requiring gradual adjustments. Examples include lighting control (dimmer), volume control, graphic equalization, and speed control. In a linear slider, the segments are placed in a straight line pattern. In radial Slider, the sensors are arranged in a circular fashion.

The power consumption for buttons and sliders will depend on the parasitic capacitance of the sensor. The greater the parasitic capacitance, the longer the scan time and to achieve the desired SNR. Increased scan time will contribute to increased active time and hence leads to increased average current consumption. Proper layout guidelines should be followed with utmost care to keep parasitic capacitance at a minimum.

Figure 6: Various capacitive sensors and their applications
Proximity sensors
The purpose of a proximity sensor is to detect the presence of an approaching object rather than determine its exact position. Examples include the glowing of backlight LEDs of buttons when user brings a hand near a car audio system, bringing up a menu on the screen, and lighting LEDs to show where the capacitive sensors are located in the case of a digital frame.

In such a case, the sensor parasitic capacitance is very high and sensitivity is high as well. Both contribute to increased scan time and hence increased power consumption. In the case of a proximity sensor, developers should note that the sleep time interval cannot more as the other sensors (buttons and/or slider) should become active within 200 ms to have a real time experience for the user.

Through the course of article, we have been discussing how to balance responsiveness and power consumed by a system. Cypress provides a family of capacitive sensing devices of which some are specifically designed for low power capacitive sensing solutions. Also, the PSoC1 / PSoC3 controllers have mixed-signal processing capabilities which simplify capacitive touch sensing design. PSoC3 has a high pin count that allows interfacing more number of capacitive sensors and has a lot of other analog and digital resources like ADCs, DACs, DFB and a DMA. These features allow PSoC to implement innovative capacitive sensing techniques in its CapSense portfolio. For more information on PSoC CapSense, visit http://www.cypress.com/capsense.