Low Power Capacitive Sensing

1. Relevant Devices

C8051F990, C8051F991, C8051F996, and C8051F997

2. Supporting Documentation

- AN367: Understanding Capacitive Sensing Signal to Noise Ratios and Setting Reliable Thresholds
- AN447: Printed Circuit Design Notes for Capacitive Sensing with the CS0 Module

3. Introduction

Capacitive sensing for human interface applications has rapidly grown in popularity due to its ability to reduce manufacturing cost, increase product lifespan by eliminating mechanical components, and enhance product look and feel. In many applications, mechanical push-button switches and potentiometers are being replaced by capacitive switches, sliders, and control wheels to implement functions such as contrast, volume control, and power on.

When capacitive sensing was first introduced into the market, it was primarily for use during the active mode of a system, where the power of the sensing function is small compared to the power draw of the overall system in active mode. To save power, battery powered systems would disable the sensing function in the inactive mode and wake up using a mechanical stimulus.

With the advancement of capacitive sensing technology and its ability to be ultra low power, even the power switch may now be implemented with a capacitive sensor.

This application note discusses how the user interface of ultra low power applications can be designed using capacitive sensing technology.

4. System Definition

The key to designing an ultra low power capacitive sensing system is in its system definition, which is the starting point of every embedded system design. The steps necessary to complete the definition are creating a power budget, defining the power modes, and identifying the system tasks that need to be performed in each power mode. Once a rough definition is in place, power targets for each power mode should be established to provide the system designer a starting point for software and hardware development.

4.1. Power Budget

An application's power budget is one of the first parameters that should be specified in the system definition. The first step in determining the power budget is deciding how long the system is required to operate without replacing batteries. For some systems, the battery only needs to last 6 months. However, some systems are sealed or placed in remote areas where battery replacement would not be possible, and the required battery life may be 10–15 years. After required battery life is established, the battery which will be used to power the system should be selected. In many cases, cost or form factor determine the largest battery that will be available to the system. Other factors such as shelf life should be considered for systems that need an extremely long battery life. Finally, the battery capacity information found in the battery data sheet of the selected battery should be used to create the power budget.

The maximum average current (power budget) for a battery powered embedded system can be calculated using the following equation:

$$\text{Maximum Average Current [mA]} = \frac{\text{Battery Capacity [mA - H]}}{\text{Required Battery Life [H]}}$$

The power modes and system tasks should be designed such that the total average current of the system is less than the maximum average current specified by the power budget.

4.2. Power Modes

There are two primary power modes in any low power system: active and inactive. A typical device with a usage profile that involves interaction with a human will spend the vast majority of its life in the inactive mode, only switching to the active mode when interacting with the human. There may be multiple "active" modes depending on the level of interaction with the user or the task being performed, but these will all be grouped under the main "active" mode for the purpose of this discussion. The primary focus of the active mode is to provide a good user interface for short periods of time. The main goal of the inactive mode is to preserve battery life.

The active/inactive power scheme allows the system to provide a good user interface and have a long battery life because for a given battery, the average current of the system is what determines the battery life. Let us analyze a typical user interface that is used for an average of 15 minutes per day. Since there are 1440 minutes in a day, the system would be in the active mode only 1% of the time. This means that if the active mode supply current is 100 µA, its contribution to the average current is only 1 µA, a factor of 100 less than the actual active mode current. It is clear that small variations in the active current do not have a significant impact on battery life. On the other hand, the average current contribution from the inactive mode cannot be divided down (1 µA inactive mode current = 1 µA contribution to average current) since the system stays in this mode for greater than 99% of the time. If we analyze this scenario carefully, we can observe that when designing a human interface, it is most important to focus on the inactive current because reducing it by a small quantity (e.g., < 1 µA) can significantly improve battery life and increasing it by even "tens of micro amps" can cause a significant reduction in battery life.

4.3. System Tasks

The next step in creating the system definition is to identify the required tasks in each power mode. Power targets for each power mode should also be established at this point.

4.3.1. Active Mode

The active mode is enabled when a user is actively interacting with the device. The user interface should be fully functional and respond quickly to user stimulus. All capacitive sense inputs used by the application need to be sampled and gesture detection performed on sliders and control wheels. The switch sampling rate in active mode is typically set in the range of 20–125 Hz to ensure a responsive user interface. The active mode target current for most applications is in the range of 100 to 500 µA.

To achieve such a low active mode current, it is necessary to break up the active mode tasks into two categories. The first category of tasks requires the CPU to be active, such as determining the system state, processing of capacitive sensor data, or other housekeeping tasks. The second category of hardware tasks can be accomplished with the CPU in an idle mode, such as sampling the capacitive sensors, taking an ADC measurement, etc. Splitting these tasks into different power modes allows a supply current to be assigned to each power mode. The two active power modes can then be averaged to obtain the final value of “active mode current”.

4.3.2. The Inactive Mode

The primary goal of the inactive mode is to preserve battery life. The inactive mode should implement a low power “wake-on-touch” algorithm to determine when the system needs to be switched to the active mode. A common “wake-on-touch” algorithm periodically samples one switch at a rate of 1–10 Hz to check if a finger has been placed on the capacitive touch pad. The ultra low power RTC is used to schedule the “wake-on-touch” checks. Target power consumption for the inactive mode in most applications is 1 to 3 µA.

5. Hardware Design

The hardware design of an ultra low power capacitive sensing embedded system consists of three easy steps:

- Determine the number of capacitive touch pads required.
- Select an ultra low power capacitive sensing MCU.
- Design a PCB with the MCU and the capacitive touch pads.

5.1. Determining the Capacitive Touch Requirements

The number of capacitive touch inputs required depends on the complexity of the user interface. Each button in the user interface requires one capacitive touch input. Sliders and control wheels are typically implemented with 4 to 8 inputs. Once the number of capacitive sense inputs is determined, it is time to move on to MCU selection.

5.2. Selecting an Ultra Low Power Capacitive Sensing MCU

For ultra low power capacitive sensing applications, it is important to select a capacitive sensing MCU that is power efficient and that has an ultra low power sleep mode that supports periodic wake-up (e.g., real time clock). One example of such an MCU is the C8051F99x family of ultra low power capacitive sensing MCUs.

Key Features of the ‘F99x MCU Family:

- 8 kB Flash, 512 bytes RAM, 25 MIPS CPU with 150 µA/MHz active mode current
- Autonomous capacitive sensing peripheral with less than 40 µs conversion time (adjustable via the CS0MD2 register)
- Ultra low power sleep mode (300 nA) with internal LFO and 2 µs wake-up time
- 10-bit, 300 ksps or 12-bit, 75 ksps ADC with internal voltage reference
- 13/14 capacitive sense inputs in a 3x3/4x4 mm package

In addition to the key features listed above, the capacitive sensing peripheral on the C8051F99x MCU family has some built-in architectural power saving features.

5.2.1. Sensing Multiple Channels in a Single Conversion

Each capacitive sensing conversion requires a finite amount of energy to complete. Reducing the total number of required conversions can reduce the amount of energy needed to perform the necessary sensing. The C8051F99x family features a “multiple channel sense” feature where multiple channels may be bonded together at runtime and sensed using a single conversion. This feature is useful for implementing low power wake-on-touch on any button in a multi-button arrangement. For best performance, the channels bonded together should have capacitive pads that are similar in size and shape. Having capacitive pads with similar size and shape allows bonded channels to provide equal weight to each pad in the combined measurement.

5.2.2. Suspend Mode Wake-Up Source

The capacitive sensing peripheral (CS0) on the C8051F99x MCUs has a built in oscillator that controls conversion timing that is independent of the system clock. This allows the MCU to enter a low power suspend mode while a conversion is taking place. The CS0 peripheral has the ability to wake up the MCU from suspend mode after the capacitive sensing conversion is complete. This allows the supply current while taking conversions to be as low as 120 µA.

5.2.3. Autonomous Hardware Averaging

In noisy environments, multiple capacitive sensing conversions are needed in order to improve resolution. The CS0 peripheral can automatically average 1, 4, 8, 16, 32, or 64 conversions for each convert start without any CPU intervention. The CPU may also enter suspend mode to be awoken after all conversions are accumulated and averaged.

5.3. Designing a PCB with an MCU and capacitive touch pads

User interfaces that use capacitive touch technology require very few external components and can be very simple to design as long as a few basic rules are followed. To demonstrate the simplicity of such designs, we will use the ‘F990 Slider Evaluation Board shown in Figure 1 as an example.

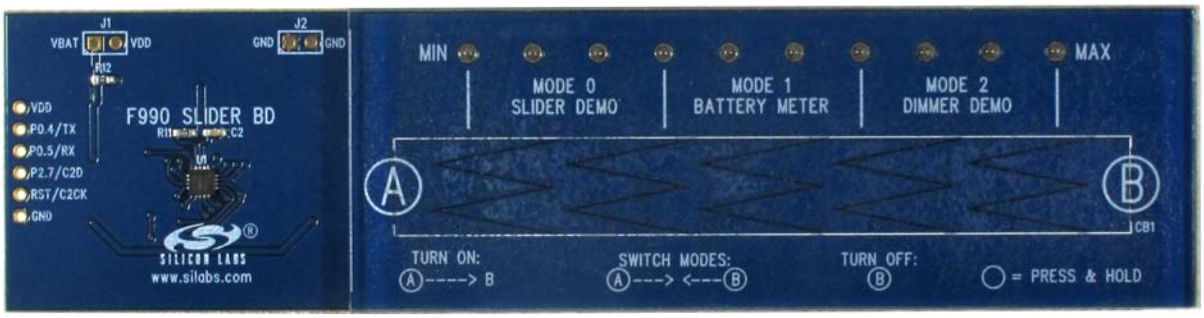


Figure 1. C8051F990 Slider Evaluation Board

The bill of materials used for this board is as follows:

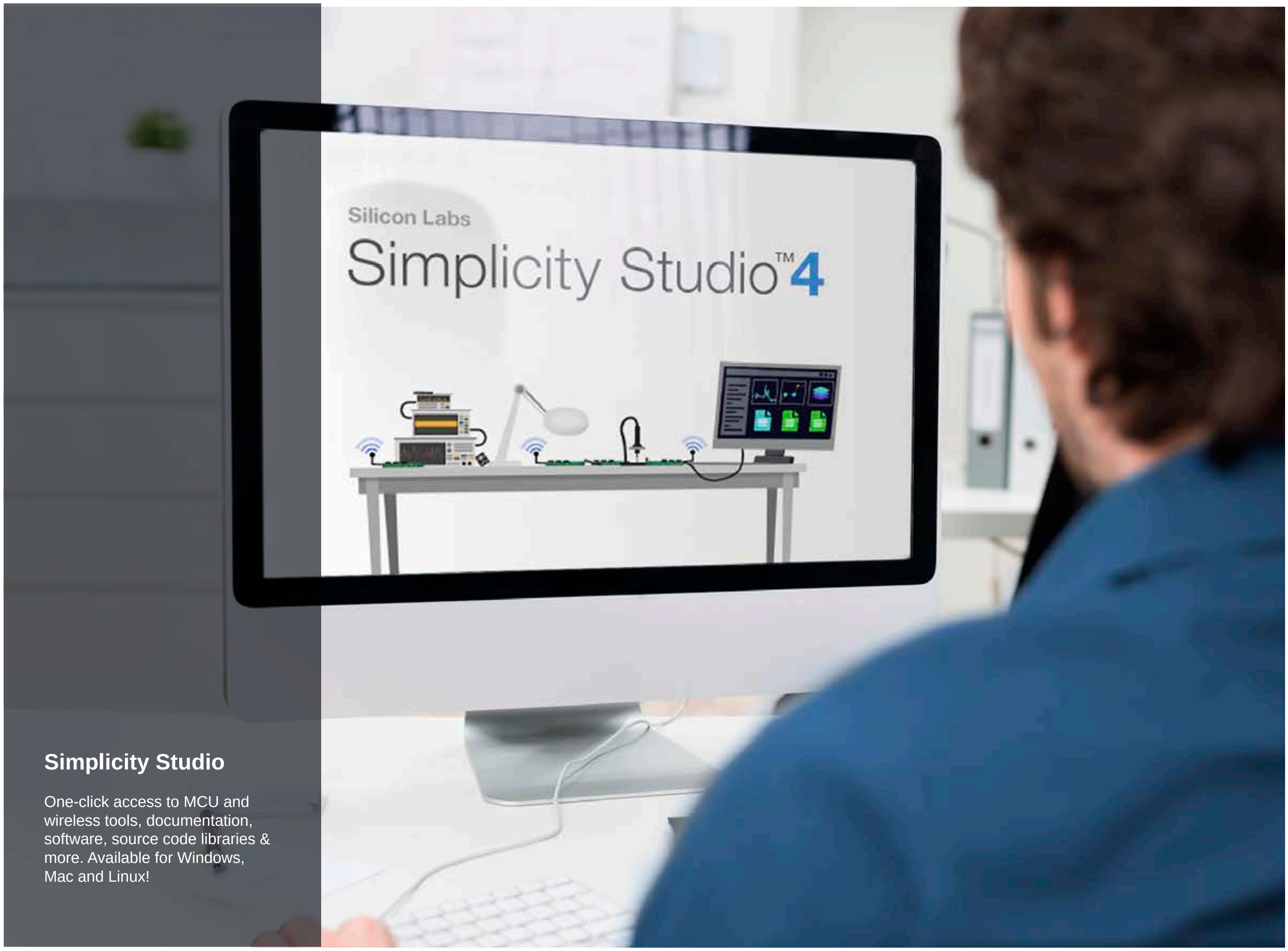
- Power Source: CR2032 battery holder and 1 µF bulk decoupling capacitor. In this design, a bulk capacitor is used because a CR2032 battery has high output impedance and its peak output current is limited. Adding a capacitor preserves the battery and increases the peak output current of the power source by instantaneously providing the system with the necessary charge to meet peak current demands. The battery only needs to provide the “average current” to the capacitor in order to maintain the supply voltage.
- MCU circuit: The MCU used is a C8051F990, which comes in a tiny 3x3 mm package and provides 16 I/O pins. The only external components needed are a pull-up resistor for the reset pin (to provide noise immunity) and a small decoupling capacitor close to the VDD pin.
- LEDs: There are 10 LEDs used in this design and 10 current limiting resistors associated with the LEDs. These are application specific and will not be needed for most systems.
- Capacitive Sense Pads: Six capacitive sense pads are used on this evaluation board to provide a method of user input. The pads are arranged in a chevron slider pattern and can be used as “buttons” or combined to form a “slider”. An acrylic overlay is typically placed over the capacitive pads to protect the system from ESD and to provide a uniform surface that may be touched by the end user.

Capacitive sense pads may be connected directly to Capacitive Sensing pins on the MCU. The C8051F990 provides 13 pins with capacitive sensing capability. It is a good idea to route the capacitive pads to the MCU on the bottom layer of a 2-layer board or an inner layer of a multi-layer board because they are sensitive to touch. Guidelines for designing capacitive sensing pads can be found in “AN447: Printed Circuit Design Notes for Capacitive Sensing with the CS0 Module.”

Document Change List

Revision 0.1 to Revision 0.2

- Removed “QuickSense”



IoT Portfolio	SW/HW	Quality	Support and Community
www.silabs.com/IoT	*www.silabs.com/simplicity*	*www.silabs.com/quality*	*community.silabs.com*

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com