AN223616

AFE Implementation Using PSoC 4

Author: Dineshbabu Mani

Associated Part Family: PSoC® 4100PS

Associated Code Examples: CE223621, CE223620, CE223618

More code examples? We heard you.

To access an ever-growing list of hundreds of PSoC code examples, please visit our code examples webpage. You can also explore PSoC 4 video library here.

AN223616 explains how to use PSoC® 4 to implement analog front ends (AFEs) in sensing applications. It discusses commonly used AFE implementations for measurement of three types of sensors – with voltage, resistance, and capacitance outputs.

Contents

1 Introduction

PSoC 4 simplifies the design of sensor-based systems by delivering a scalable and reconfigurable architecture that integrates programmable AFEs to a signal processing engine. PSoC 4 enables designs to send aggregated, pre-processed, and formatted sensor data over serial communication interfaces to host processors.

Analog sensors generally come in five different types, depending on their output electrical signal – voltage, current, resistance, capacitance, or inductance. Each sensor type requires a specific AFE design. For example:

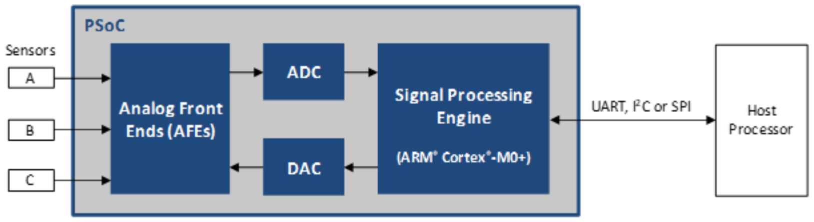
- A thermocouple, which is a voltage-output temperature sensor, requires an instrumentation amplifier
- An ambient light sensor, which is a current-output sensor, requires a trans-impedance amplifier (TIA)

Systems that use multiple analog sensors usually require multiple specialized ICs to implement the AFE, which increases BOM cost and PCB size. Systems designed for IoT applications must combine data from multiple sensors to enable new sensing capabilities, commonly known as sensor fusion. Sensor fusion solutions often require custom AFEs. PSoC 4 reduces the need for specialized ICs, offering the ability to create custom AFEs in a single-chip solution.

Figure 1 shows a generic block diagram of a sensor-based system. It includes:

1. An AFE to condition the sensor outputs by amplifying and filtering the signals.
2. An analog-to-digital converter (ADC) or a comparator (not shown) to convert the conditioned sensor output into digital data.
3. A programmable signal processing engine with a serial communication interface, to format the sensor data and send it to the host processor. A signal processing engine (32-bit Arm® Cortex®-M0+) can calibrate and tune the AFE in software.

Figure 1. Sensor-Based System



This application note explains various AFE implementation methods for voltage, resistance, and capacitance output sensors. It also discusses performance analysis for each type of sensor. Finally, it explains how to interface these sensors with PSoC 4 using code example references.

The Related Documents section provides a rich set of documents to accelerate your learning. If you are new to PSoC 4100PS, see AN79953, Getting Started with PSoC 4 to understand the device and to learn how to create a simple design.

2 Analog Front Ends for Different Sensor Types

Analog sensors are classified into five types: resistive, inductive, capacitive, voltage, or current, depending on their output parameter. The output of these sensors change based on a physical entity. For example, in a thermistor the resistance changes based on the temperature. Each sensor type requires the right AFE.

Although there are five types of output, the basic electrical parameters that can be measured by a microcontroller (MCU) are voltage, time, and frequency. The output of the sensor is converted to one of these three types of electrical parameters to determine the equivalent digital count. For example, in a thermistor, the resistance of the sensor changes due to temperature. The thermistor can be placed in series with a reference resistor to form a resistive divider which is driven by a constant voltage source, and then the voltage drop across the thermistor and reference resistor can be measured. The resistance of the thermistor can be calculated from the measured voltage drop values and the known reference resistor.

Each AFE explained in this application note uses a specific method to convert a sensor output to a basic electrical parameter – voltage, time, or frequency.

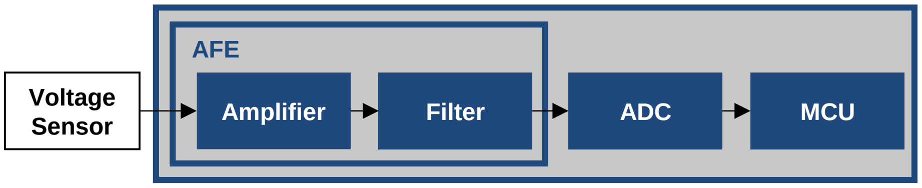
3 AFE for Voltage Output Sensor

In a voltage output sensor, the voltage across sensor terminals varies with respect to the physical quantity. Some examples of voltage output sensors are:

- Thermocouples, where the voltage output changes with temperature
- Hall-effect sensors, where the voltage output varies with the magnetic field caused by the current flowing in an adjacent or built-in conductor.
- PIR motion sensors, where the voltage output of the sensor changes when an infrared emitting object passes in front of the sensor

Figure 2 shows a generic block diagram of the signal chain for a voltage output sensor.

Figure 2. Voltage Measurement AFE Block Diagram



The output of the sensor is first connected to an amplifier. The function of the amplifier is to amplify the input signal to be compatible with the ADC's input range. For example, if a voltage output of the sensor has a maximum output of 30 mV and the ADC has a range of 1 V, then the amplifier may have a gain of 32 to amplify the signal to 0.96 V, allowing a 0.04-V margin for offset voltage, signal overshoot etc. For signals that are already in the ADC's measurement range, the amplifier can be a unity gain buffer or may not even be needed.

Many sensors require an amplifier with a high input impedance. If the sensor signal is high enough to bypass the amplifier stage, the output of sensor can be connected to an ADC through a unity gain buffer that provides high input impedance to ADC. For sensors such as thermocouples and current shunts, a differential amplifier is preferred. For the differential sensors with low output impedance and significantly higher output voltage signal, the output of the sensor can directly be connected to the differential ADC. For single-ended signals such as hall-effect sensors, a single ended amplifier may be used.

The output of the amplifier is then connected to an optional filter. The filter can have many functions, depending on the signal being measured. Some of the functions are:

- Low-pass filter for anti-aliasing
- Notch filter to attenuate power supply noise
- Band-pass or high-pass filter to remove DC bias and extract only the interested band of the signal

The output of the filter is then connected to the ADC. The MCU may further process the digitized signal from the ADC. Note that in a resource constraint design, an MCU like CM0+ can be used to implement firmware filters.

3.1 Implementation

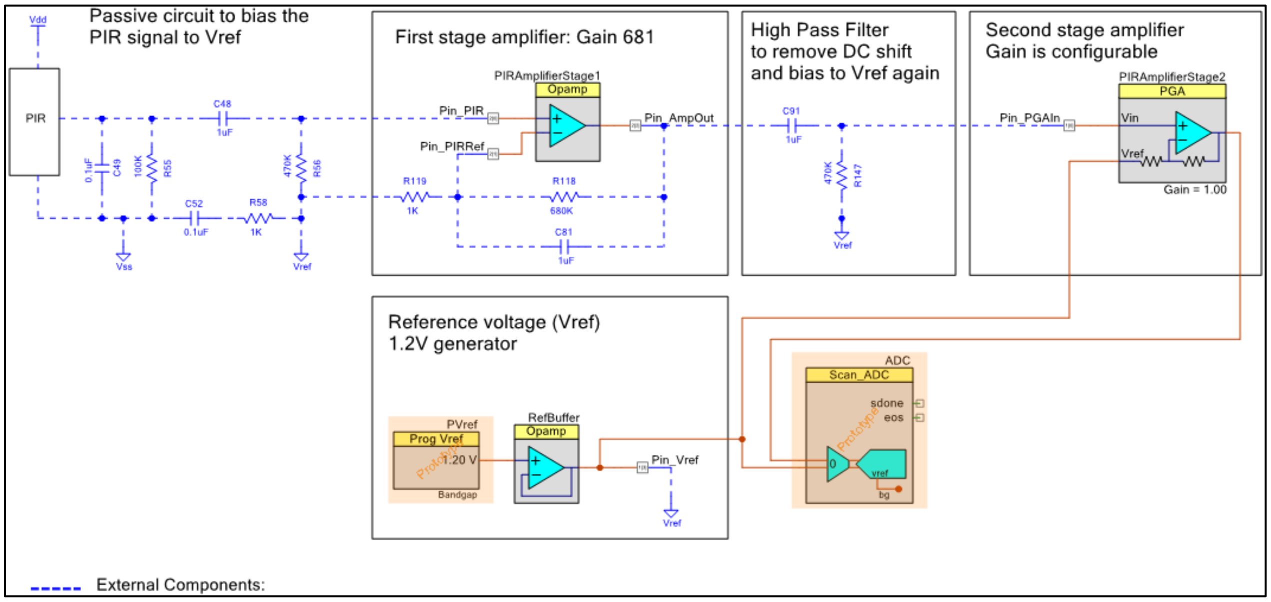
In this section let us discuss how to implement an AFE for a PIR motion sensor, which is a voltage output sensor. For details of firmware implementation, see the code example CE223618.

The PIR motion sensor uses infrared sensitive materials as the sensing elements. It is packaged with a field effect transistor (FET) in the source follower mode. An FET is required to buffer the high-impedance output of the sensor element.

When the sensor element is exposed to infrared radiation, a voltage is generated across the element. These elements are arranged such that the voltage generated by one element is subtracted by the other. This arrangement cancels the common signal and generates a voltage only when there is a difference in the incident infrared radiation level on the sensing elements. The sensor package is designed to have a unique field-of-view for each element. When an IR radiating source moves across the fields of view, the sensor generates a differential signal across the load resistor. To increase the field-of-view angle, a Fresnel lens is mounted on the PIR motion sensor. It improves the sensitivity and thus the detection distance.

Figure 3 shows the PSoC Creator™ schematic for a PSoC 4100PS configured as an AFE for a PIR motion sensor.

Figure 3. PSoC Creator Schematic


Passive circuit to bias the PIR signal to Vref
First stage amplifier: Gain 681
High Pass Filter to remove DC shift and bias to Vref again
Second stage amplifier Gain is configurable
Reference voltage (Vref) 1.2V generator
PIRAmplifierStage1
Opamp
Pin_PIR
Pin_PIRRef
Pin_AmpOut
PIRAmplifierStage2
PGA
Pin_PGAIn
Gain = 1.00
PVref
Prog Vref
1.20 V
Bandgap
RefBuffer
Opamp
Pin_Vref
ADC
Scan_ADC
sdone
eos
External Components:

The sensor voltage pulse is first passed through a high pass filter to block the DC voltage from the sensor into the next stages.

The peak-to-peak voltage signal from the high pass filter is on the order of millivolts, and must be amplified. To detect the motion of a human body at a distance of 10 feet, an amplifier with a gain of ~1000 is required. A single-stage amplifier with such a high gain causes the amplifier output to become saturated. Thus, a two-stage amplifier is best suited for amplifying with such a high gain.

The total gain is split between two stages. The first stage amplifier uses a non-inverting amplifier configuration, using an internal Opamp and external gain setting resistors. The second stage amplifier uses a PGA Component.

The reference voltage from the PVref Component is buffered using an Opamp Component, and then used to bias the PIR sensor and amplifier stages. The output of the second stage PGA is connected to the Scan_ADC Component. See the Scanning SAR ADC Datasheet for details on Scan_ADC.

The Scan_ADC results are compared against threshold values to detect the motion of an IR emitting object.

4 AFE for Resistive Sensor

In resistive sensors, the resistance of the sensor varies with respect to the physical quantity being measured. Some examples of resistive sensors are:

- Thermistor and Resistance Temperature Detector (RTD) sensor, where the resistance varies with temperature
- Strain gauge, where the resistance varies with load
- Light Dependent Resistor (LDR), where the resistance varies with intensity of light

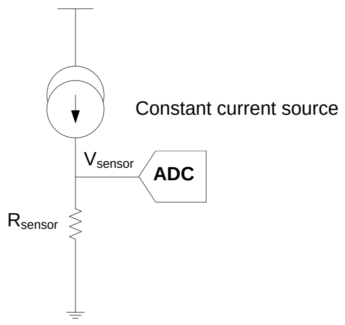
This section explains two methods for resistance measurement. It also explains an implementation in PSoC 4100PS, with a code example. A performance analysis section explains the errors associated with one of the resistance measurement methods, and tips to handle such errors.

4.1 Current Source Method

In this method, the sensor (R_{sensor}) shown in Figure 4 is excited using a constant current source. The voltage drop across the sensor is measured using an ADC. After the voltage is measured, the resistance of the sensor is calculated using Equation 1. In this method, the accuracy of the resistance measurement depends on the accuracy of the current source, and the offset and gain errors of the ADC. Note that the ADC gain error depends on the accuracy of reference voltage.

$$R_{sensor} = \frac{V_{sensor}}{I}$$ Equation 1

Figure 4. Resistance Measurement Using Constant Current Source



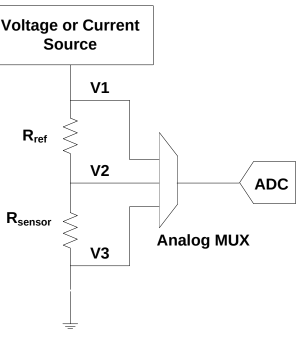
The following are the demerits of the Current Source method:

1. The current source must be very accurate; any current error causes an error in the measured resistance and hence an error in the measured physical entity.
2. The offset, gain, and integral nonlinearity (INL) error of the ADC caused by an inaccurate reference voltage can lead to inaccuracies in the measured resistance.
3. Nonlinearity of R_{sensor} will cause a nonlinearity of V_{sensor}, which will cause an error in the measured physical entity.

4.2 Ratiometric Resistor Divider Method

The method shown in Figure 5 is used to reduce some of the errors associated with the previous method. A reference resistor (R_{ref}) and the sensor (R_{sensor}) form a resistor divider, which is excited using a voltage or current source. The resistor divider can also be excited from the device supply voltage based on the requirement. R_{sensor} is calculated using the ratio of voltages developed across the reference resistance and the sensor.

Figure 5. Resistance Measurement Using Reference Resistor



Note: The location of the R_{ref} and R_{sensor} can be exchanged based on the measurement range of the physical quantity. For example, if temperature is the physical quantity being measured and if you use negative temperature coefficient element like thermistor, it is recommended to place the thermistor on the top and the reference resistor at the bottom. This arrangement ensures that the low resistance of thermistor at a high temperature does not cause the ADC input signal to go below the measurement range.

An ADC is used to measure V1, V2, and V3, and the sensor resistance is calculated from Equation 2.

$$R_{sensor} = \frac{(V2 - V3)}{(V1 - V2)} * R_{ref}$$ Equation 2

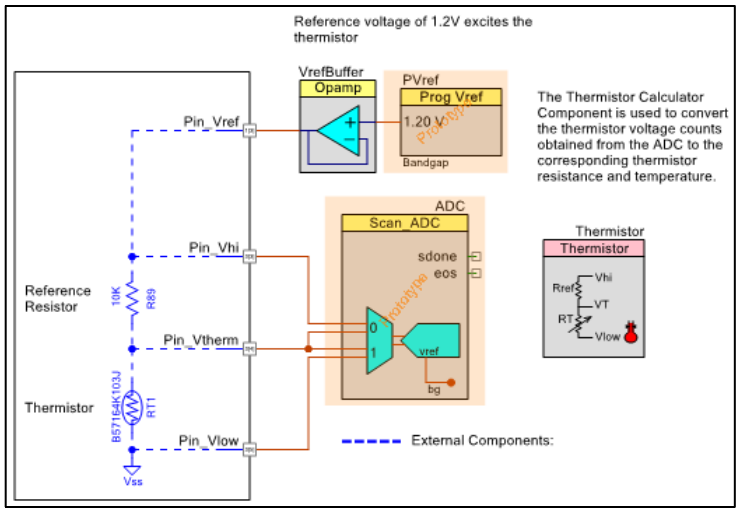
In this method, the accuracy of the sensor resistance measurement depends only on the accuracy of the reference resistor and the ADC linearity. Using precise reference resistor is the cheapest way to improve the precision of this measurement circuit. This method is independent of the errors in the excitation source as well as the ADC offset and gain errors. In Equation 2, the ADC's offset error gets cancelled in the subtraction operations in the numerator and denominator, and the ADC's gain error gets cancelled by the ratio of numerator and denominator.

4.2.1 Implementation for Resistor Divider Method

In this section, let us discuss how the potential divider method is implemented in PSoC Creator to measure temperature using a thermistor. In this example, a Negative Temperature Coefficient (NTC) thermistor (B57164K103J) is used. This section focuses only on the design and analysis of the AFE. For more information, see Code Example CE223621.

Figure 6 shows how the potential divider method is implemented in PSoC 4100PS.

Figure 6 PSoC Creator Schematic for Temperature Sensing


Reference voltage of 1.2V excites the thermistor
VrefBuffer
Opamp
PVref
Prog Vref
1.20 V
Bandgap
The Thermistor Calculator Component is used to convert the thermistor voltage counts obtained from the ADC to the corresponding thermistor resistance and temperature.
Pin_Vref
Pin_Vhi
Pin_Vtherm
Pin_Vlow
ADC
Scan_ADC
sdone
eos
vref
bg
Thermistor
Thermistor
Rref
RT
Vhi
VT
Vlow
Reference Resistor
10K
R89
Thermistor
B57164K103J
RT1
Vss
External Components:

A reference resistor R_1 is connected in series with the thermistor. The thermistor is biased using a buffered 1.2-V voltage from the programmable reference Component PVref. In the resource constraint design, the device supply voltage can be used to bias the thermistor with a reduced resolution of measurement in the given temperature range. Another advantage of using bias voltage from the programmable reference Component PVref is that it can be dynamically turned ON or OFF in a power-efficient design.

The three voltage signals from the potential divider are connected to the mux which is part of the Scan_ADC Component. The three voltage signals are connected to two differential channels. The Scan_ADC Component reference is set to enable the full-scale range of the ADC. Using the full-scale range of the ADC results in increased measurement resolution.

The ADC conversion output is passed to the `Thermistor_GetResistance()` API of Thermistor Calculator Component to calculate the resistance value of the thermistor. The resistance value is then passed to the `Thermistor_GetTemperature()` API to calculate the temperature based on the thermistor parameters entered by the user in the Thermistor Calculator Component. Temperature can be calculated by using either the Steinhart–Hart method or the lookup table method. The code example, CE223621 implements the code using the lookup table method.

The same technique can be used to measure resistance of other types of sensors such as RTDs, strain gauges, and LDRs.

4.3 Accuracy Analysis

Resistive sensor measurement accuracy depends on the accuracy of the resistance measurements. When measuring resistances, the following are sources of error:

1. Tolerance of reference resistor
2. ADC offset and gain
3. ADC nonlinearity

See the Error in Physical Quantity, for example, Temperature or Pressure section to calculate the error in the physical quantity based on the error in the resistance measurement.

Tolerance of Reference Resistor

Let us start with Equation 2, which is reproduced below:

$$R_{sensor} = \frac{(V2 - V3)}{(V1 - V2)} * R_{ref}$$

Now, add an error ΔR_{ref} to R_{ref}. This results in an error ΔR_{sensor} added to R_{sensor}.

$$R_{sensor} + \Delta R_{sensor} = \frac{(V2 - V3)}{(V1 - V2)} * (R_{ref} + \Delta R_{ref})$$ Equation 3

The change in $\frac{(V2-V3)}{(V1-V2)}$ is small, and is ignored to simplify the calculation. You get Equation 6 by dividing Equation 3 by Equation 2.

$$\frac{R_{se\ nsor} + \Delta R_{sensor}}{R_{sensor}} \cong \frac{R_{ref} + \Delta R_{ref}}{R_{ref}}$$ Equation 4

$$1 + \frac{\Delta R_{sensor}}{R_{sensor}} \cong 1 + \frac{\Delta R_{ref}}{R_{ref}}$$ Equation 5

$$\frac{\Delta R_{sensor}}{R_{sensor}} \cong \frac{\Delta R_{ref}}{R_{ref}}$$ Equation 6

As seen in the above expression, the % error introduced in R_{sensor} is approximately same as the % error in R_{ref}. The % error in R_{ref} is the tolerance of the resistor. For example, if the tolerance of R_{ref} is 0.1%, the maximum error introduced by R_{ref} in the resistance measurement is approximately 0.1%.

Table 1 shows the actual errors in measured resistance for a 10-KΩ reference resistance. The % error in the measured value of R_{sensor} is same for all values of R_{sensor}. A 0.1% error in R_{ref} introduces a 0.1% error in R_{sensor}. A 5% error in R_{ref} introduces a 4.76% error in R_{sensor}.

Table 1. Error in Resistance Measurement for Various Values of Error in Reference Resistance

R_{sensor}	**Error in R_{ref}**			
	0.10%	**1.00%**	**2%**	**5%**
10 KΩ	0.10%	0.99%	1.96%	4.76%

ADC Offset and Gain Errors

Let us now introduce offset error V_{os} and gain error x to the Equation 2.

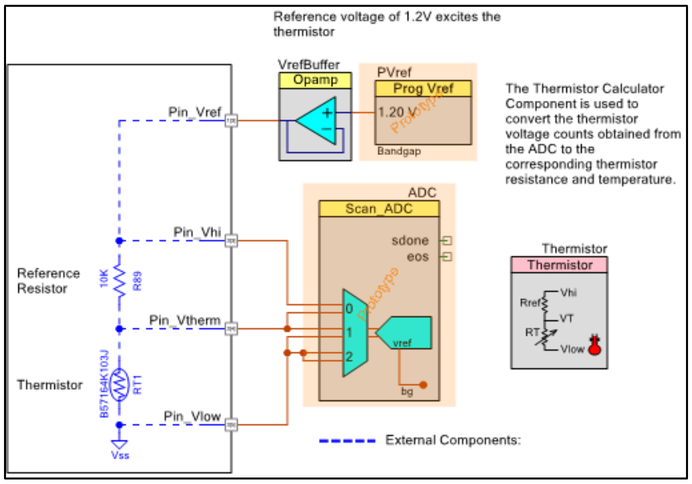
$$R_{sensor} = \frac{x(V2 + V_{os}) - x(V3 + V_{os})}{x(V1 + V_{os}) - x(V2 + V_{os})} * R_{ref} \qquad \text{Equation 7}$$

$$R_{sensor} = \frac{\cancel{x}(V2 + \cancel{V_{os}} - V3 - \cancel{V_{os}})}{\cancel{x}(V1 + \cancel{V_{os}} - V2 - \cancel{V_{os}})} * R_{ref} \qquad \text{Equation 8}$$

Equation 8 shows that both the offset and gain errors are cancelled in the potential divider method. Hence, ADC offset and gain errors do not affect the accuracy of measurement.

When using differential measurement as shown in Figure 6, the offset voltage of the ADC can be measured by adding one more differential channel and shorting both terminals to ground as shown in the channel#2 of Figure 7. The offset count can then be subtracted from the counts that correspond to channel#0 and channel#1 to remove the offset error.

Figure 7. Offset Compensation in Differential Modes of ADC



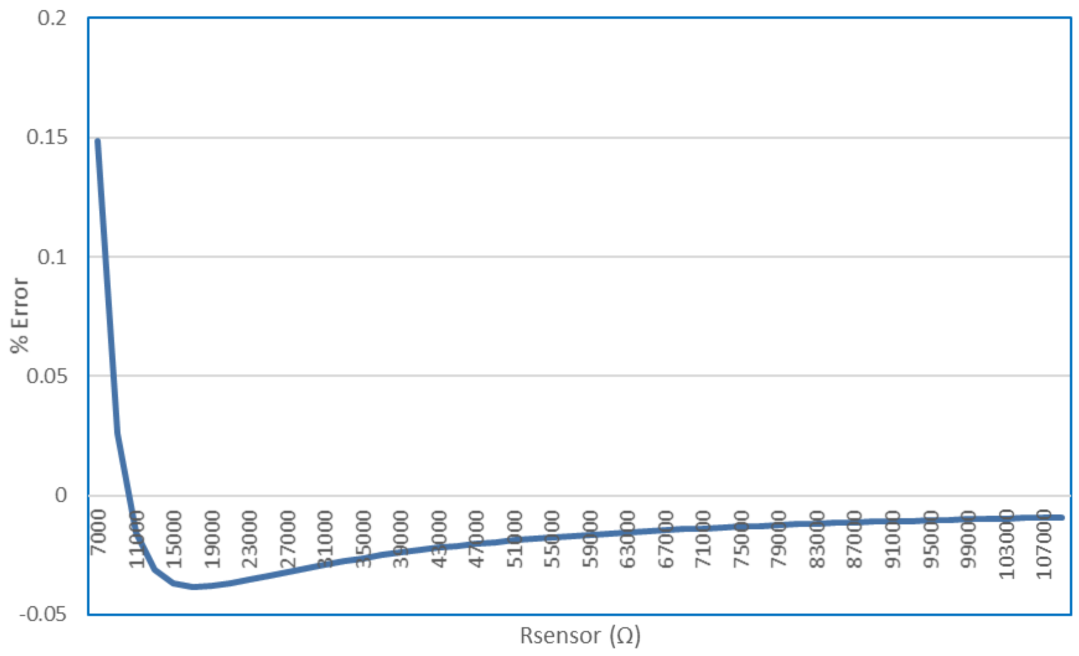
ADC Nonlinearity Error

The SAR ADC in PSoC 4100PS device has a maximum INL of ±2 LSB. For a 1.2 V range and 12-bit resolution, the INL error is approximately 1.2 mV; the worst-case error occurs when the INL voltage affects both numerator and denominator of Equation 2. Consider the following example, where:

R_{sensor} = 7 kΩ R_{ref} = 10 kΩ Excitation voltage = 1.2 V

The worst-case error in the measured resistance due to the INL error is 10 Ω, which is 0.14% error. The worst-case error in the measured resistance due to ADC INL error for various values of R_{sensor} is shown in Figure 8.

Figure 8. Worst case Error in Resistance Due to INL versus R_{sensor}



Error in Physical Quantity, for example, Temperature or Pressure

This section describes the exact steps to calculate the error in the measured physical quantity, for example ΔT in case of temperature, due to the error in the measured resistance value (ΔR_{sensor}).

1. Calculate the error in sensor resistance (ΔR_{sensor}) for a known set of R_{ref}, ΔR_{ref}, and R_{sensor} from Equation 6.
2. Calculate the absolute value of the deviated sensor resistance R'_{sensor} from Equation 9.

$$R'_{sensor} = R_{sensor} \pm \Delta R_{sensor} \qquad \text{Equation 9}$$

3. From the sensor manufacturer's datasheet, find the physical quantity corresponding to R_{sensor} and R'_{sensor}.
4. Take the difference between the physical quantity (like temperature or pressure) values calculated in steps 3 to calculate the error in physical quantity due to the variation in R_{ref}. This value corresponds to the error in physical quantity.

Reference Resistor Selection

To achieve the maximum resolution at either extreme of temperature, the reference resistance should be close to the resistance of the thermistor at the middle of its temperature range. If you are more interested in measuring the temperature at one extreme, then the reference resistance should match the resistance of the thermistor at the temperature extreme being measured.

Conversion Accuracy

The accuracy of converting resistance to physical quantity depends on the accuracy of curve fitting to match the nonlinear characteristics of the sensor. For example, in the case of NTC (Negative Temperature Coefficient) thermistors, the resistance of the thermistor decreases as the temperature rises. The variation of resistance with temperature is nonlinear. In the Thermistor Calculator Component, the conversion error across a temperature range of –40 °C to 125 °C is less than 0.01 °C.

5 AFE for Capacitive Sensor

In capacitive sensors, the capacitance between the sensor terminals varies with respect to the physical quantity being measured. Some examples of capacitive sensors are:

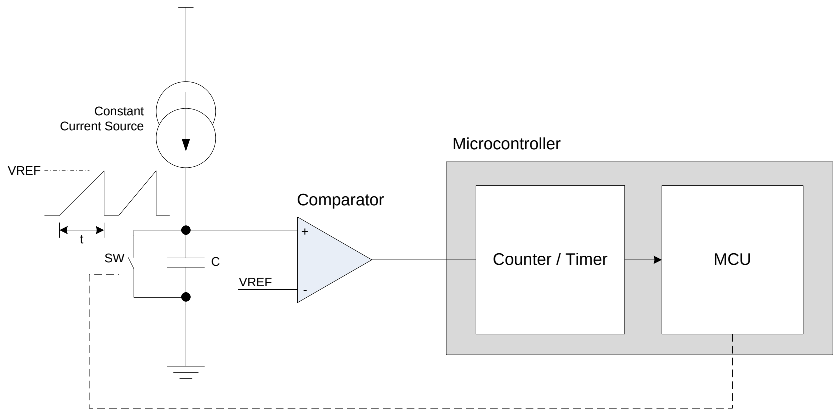
- Humidity sensors
- Pressure sensors
- Proximity sensors

This section describes two AFE designs for measuring a capacitive sensor. It also explains an implementation using PSoC 4100PS, with a code example. A performance analysis section explains the errors associated with one of the capacitance measurement methods, and tips to handle such errors.

5.1 Single Slope Method

In this method, a constant current source is used to charge the capacitor. The time taken to charge the capacitor to a known voltage can be used to measure capacitance.

Figure 9. Single Slope Method to Measure Capacitance



In the above circuit, the microcontroller first keeps the switch SW (could be a GPIO or an external FET) closed, thus holding the capacitor in a discharged state. Then the microcontroller releases the switch. The current source charges the capacitor, and the voltage across the capacitor ramps up linearly. When the voltage across the capacitor becomes greater than VREF, the comparator output goes high. The microcontroller uses the Timer/Counter to measure this time duration, and then calculates the capacitance C using Equation 10.

$$C = \frac{I\,t}{V} \qquad \text{Equation 10}$$

Where,

I = Current from the constant current source

V = Reference voltage of the comparator

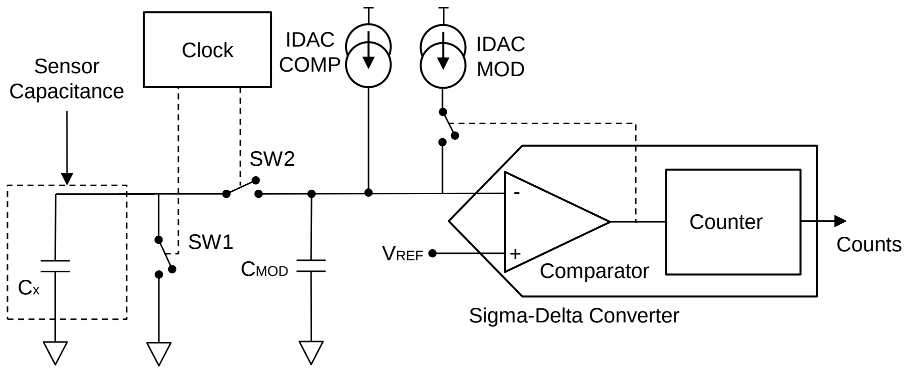
t = Time taken to charge the capacitor to VREF

The accuracy of this method depends on the accuracy of the constant current source, the comparator reference voltage and the timer.

5.2 Sigma Delta Modulator Method

In this method, a Sigma Delta modulator converts the capacitance into a digital bit stream, where the density of the bit stream is directly proportional to the capacitance value. Figure 10 shows the Sigma Delta Modulator.

Figure 10. Sigma Delta Modulator

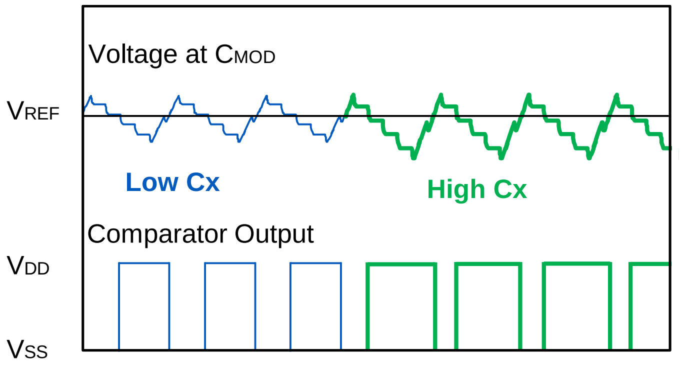


$IDAC_{MOD}$ is a current source that charges a modulation capacitor C_{MOD}. When the voltage across C_{MOD} goes above VREF, the output of the comparator goes low and turns off $IDAC_{MOD}$.

C_x is the sensor capacitance to be measured. SW1 and SW2 along with C_x form a switch capacitor cell. SW1 and SW2 are driven out of phase by a clock source. When SW2 is ON, charge is transferred from C_{MOD} to C_x. When SW1 is ON, C_x is discharged. Thus Cx, SW1, and SW2 together act as a resistor discharging C_{MOD}. As C_{MOD} is discharged, the voltage drops. When the voltage drops below V_{REF}, the output of the comparator goes high and turns on $IDAC_{MOD}$ which restarts the charging cycle. $IDAC_{COMP}$ is a fixed current source that can be used with $IDAC_{MOD}$ to increase the dynamic range of measurement.

Figure 11 shows the output of the comparator and the voltage across C_{MOD}.

Figure 11. Output of Comparator for Different C_X Values



A smaller C_x (signals shown in blue) results in less charge taken out of C_{MOD}, thus letting C_{MOD} charge faster to V_{REF}. This results in the comparator output staying HIGH for a shorter period of time. A larger C_x (signals shown in green) results in more charge taken out of C_{MOD}, thus increasing the time to charge C_{MOD}. This results in comparator output staying HIGH for longer period of time. The counter integrates the output of the comparator over a given period of time, and the output of the counter is directly proportional to the capacitance.

The following equation shows the relationship between the output of the counter, also called "raw count", and the capacitance.

$$\mathrm{raw\ count} = (2^{\mathrm{N}} - 1)\,\frac{(\mathrm{V_{DD}} - \mathrm{V_{REF}})\,\mathrm{F_{SW}}}{\mathrm{I_{MOD}}}\,\mathrm{C_S} - (2^{\mathrm{N}} - 1)\,\frac{\mathrm{I_{COMP}}}{\mathrm{I_{MOD}}}$$ Equation 11

Where,

C_S – Capacitance being measured

N – Resolution of the counter in bits

F_{SW} – Frequency at which SW1 and SW2 are operated

I_{MOD} – Current from $IDAC_{MOD}$

I_{COMP} – Current from $IDAC_{COMP}$

V_{REF} – Reference voltage of Comparator

As can be seen from the Equation 11, the accuracy of measurement depends on $IDAC_{MOD}$, $IDAC_{COMP}$, V_{DD} and V_{REF}. An error in any of these parameters introduces an error in measured capacitance.

To compensate the error in these parameters, a single reference capacitor with good tolerance and temperature stability is used. Both the sensor capacitor and the reference capacitor are measured with the Sigma Delta Modulator, and the value of the sensor capacitor can be found from the ratio of raw counts.

$$C_{sensor} = \frac{C_{ref}}{RawCounts_{Cref}} * RawCounts_{Csensor}$$ Equation 12

Though this method looks much complex than the single slope methods, all the hardware necessary to implement this method is present inside PSoC 4100PS in the CapSense® Sigma Delta (CSD) block. This makes it easy to use this method.

5.2.1 Parasitic Capacitance

The parasitic capacitance of the PCB trace connecting the sensor capacitance to the sensing hardware and pin capacitance are added to the result, introducing an offset error.

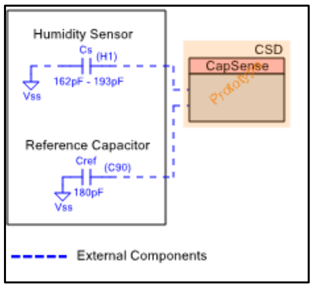
There are two ways to address this offset caused by PCB trace. The first is to keep the PCB trace as short as possible to reduce the parasitic capacitance. This method is used for the reference capacitor (C_{ref}). This is usually not practical for the sensor capacitor. In such cases, the parasitic capacitance of the system is measured by taking the initial measurements with the sensor disconnected from PSoC. The result can then be subtracted from the sensor capacitor measurements.

5.2.2 Implementation for Sigma Delta Method

In this section let us discuss how to use Sigma Delta Modulator method to measure Capacitance of a humidity sensor using PSoC 4100PS. For details of firmware design, see Code Example CE223620.

Figure 12 shows the PSoC Creator schematic for the humidity sensor project. The CapSense Component implements the sigma delta modulator hardware. The Component also has API functions to report the raw counts corresponding to the capacitance. Note that the CapSense Component is usually used for touch sensing applications like capacitive buttons, sliders, and trackpads. In this example, the CapSense Component is used for measuring the capacitance of the humidity sensor.

Figure 12. PSoC 4 Schematic for CSD



Using the CapSense component, PSoC 4100PS measures the raw counts (RawCount$_{cs}$) that correspond to the reference capacitor value (C$_{ref}$). Then the raw counts with the sensor disconnected is measured. These raw counts (RawCount$_{Cos}$) correspond to the parasitic capacitance (C$_{os}$). Since the reference capacitor (C$_{ref}$) is known, the value of the parasitic capacitance can be found using Equation 12. The raw counts with the sensor connected (RawCounts$_{CS}$) is measured. The value of the sensor capacitor is calculated using Equation 13.

$$C_s = \frac{C_{ref} * (RawCount_{Cs} - RawCount_{cos})}{RawCount_{Cref} - RawCount_{cos}} \qquad \text{Equation 13}$$

Where,

C$_{ref}$ – Reference capacitance value

C$_{s}$ – Sensor capacitance value

RawCount$_{Cs}$ – Raw count from measuring humidity sensor capacitance

RawCount$_{Cos}$ – Raw count value that corresponds to C$_{os}$ (caused by PCB trace and GPIO capacitance)

RawCount$_{Cref}$ – Raw count from measuring reference capacitance

Once sensor capacitance is calculated, humidity is calculated by using Equation 14.

$$\%RH = \frac{C_s - C_{nom}}{Sensitivity} + \%RH_{nom} \qquad \text{Equation 14}$$

Where,

%RH – Relative Humidity

C$_{s}$ – Measured capacitance of humidity sensor

C$_{nom}$ – Nominal capacitance of sensor (For the HS1100, this is 180pF)

%RH$_{nom}$ – Nominal humidity of the sensor (For the HS1100, this is 55%RH)

Sensitivity – pF/%RH value of the sensor (For the HS1100, this is 0.34pF/%RH)

5.3 Accuracy Analysis

The system accuracy of humidity measurement depends on the accuracy of capacitance measurement by CapSense Component and the sensor linearity.

As seen in the previous section, capacitance is calculated by using Equation 15.

$$C_s = \frac{C_{ref} * (RawCount_{Cs} - RawCount_{cos})}{RawCount_{Cref} - RawCount_{cos}}$$ Equation 15

From Equation 15, the accuracy of capacitance measurement depends on the accuracy of reference capacitor C_{ref}. The error in the measured Capacitance C due to the change in C_{ref} is provided by Equation 16.

$$Error = \Delta C = \frac{\pm C * \Delta C_{ref}}{C_{ref} \pm \Delta C_{ref}}$$ Equation 16

6 Summary

This application note explains the AFE for voltage, resistive and capacitive sensors. It provides the implementation details of each element in the AFE using PSoC Creator Components.

7 Related Documents

Application Notes	
AN79953 - Getting Started with PSoC® 4	Introduces you to PSoC 4, helps you explore the PSoC 4 architecture and development tools, and shows you how to create your first project using PSoC Creator
AN86233 - PSoC® 4 Low-Power Modes and Power Reduction Techniques	Discusses how to use the PSoC 4 low-power modes and features to operate at very low power levels
AN88619 - PSoC 4 Hardware Design Considerations	Shows you how to design a hardware system around a PSoC 4 device
AN73854 - PSoC® 3, PSoC 4, and PSoC 5LP Introduction to Bootloaders	Briefly introduces bootloader theory and technology
AN89056 - PSoC® 4 – IEC 60730 Class B and IEC 61508 SIL Safety Software Library	Describes the PSoC 4 IEC 60730 Class B and IEC 61508 safety integrity level (SIL) Safety Software Library and example projects
AN60590 - PSoC® 3, PSoC 4, and PSoC 5LP - Temperature Measurement with a Diode	Explains diode-based temperature measurement using PSoC 3, PSoC 4, and PSoC 5LP
AN70698 - PSoC® 3, PSoC 4, and PSoC 5LP – Temperature Measurement with an RTD	Explains the theory of temperature measurement using an RTD and includes projects to interface RTD with PSoC
AN66477 - PSoC® 3, PSoC 4, and PSoC 5LP – Temperature Measurement with a Thermistor	Describes how to measure temperature with a thermistor using PSoC 3, PSoC 4, or PSoC 5LP
AN89610 - PSoC® 4 and PSoC 5LP Arm® Cortex® Code Optimization	Shows how to optimize C and assembler code for the Arm Cortex CPUs in PSoC 4 and PSoC 5LP
AN90799 - PSoC® 4 Interrupts	Explains the interrupt architecture in PSoC 4 and its configuration in PSoC Creator
AN86439 - PSoC 4 - Using GPIO Pins	Explains how to effectively use PSoC 4 GPIO pins with various use case examples to demonstrate their features
AN85951 - PSoC 4 CapSense® Design Guide	Shows how to design capacitive touch sensing applications with PSoC 4
AN92239 - Proximity Sensing with CapSense®	Demonstrates PSoC 4 proximity sensing using CapSense
AN86526 - PSoC 4 I2C Bootloader	Describes an I2C-based bootloader for PSoC 4
AN68272 - PSoC® 3, PSoC 4, and PSoC 5LP UART Bootloader	Describes a UART-based bootloader for PSoC 3, PSoC 4, PSoC 5LP, and PSoC Analog Coprocessor

AN87391 - PSoC® 4 Segment LCD Drive	Demonstrates how easy it is to drive segment LCD glass using the integrated LCD driver in PSoC 4
AN84858 - PSoC® 4 Programming Using an External Microcontroller (HSSP)	Shows how to implement PSoC 4 device programming with an external microcontroller by using modular C code
Code Examples	
CE223618 - Interfacing PSoC 4 with PIR Motion Sensor	Demonstrates how to interface PSoC 4 with a PIR motion sensor
CE223621 - Interfacing PSoC 4 with Temperature Sensor	Demonstrates how to interface PSoC 4 with a thermistor
CE223620 - Interfacing PSoC 4 with Humidity Sensor	Demonstrates how to interface PSoC 4 with a humidity sensor

About the Author

Name:	Dineshbabu Mani
Title:	Staff Applications Engineer
Background:	Dineshbabu has Masters in Microelectronics from BITS, Pilani, India. His areas of interest include Mixed signal system design, Control Systems, and Signal Processing.

Document History

Document Title: AN223616 - AFE Implementation Using PSoC 4

Document Number: 002-23616

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6140998	DIMA	04/16/2018	New application note

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709