Application Note

Ultra-Low Power Sensing Techniques for Building Automation Applications Using TI's Sensor Controller

Eyal Cohen, Ahmad Ibrahim, Svein Vetti

ABSTRACT

This application note explores the versatility of the Sensor controller, an ultra-low-power auxiliary processor featured in the CC13x2/4 and CC26x2/4 wireless devices, specifically in the context of building automation applications. The document sheds light on the Sensor Controller's power efficiency and robust processing capabilities, being designed for managing and interfacing with sensors in low-power continuous sensing tasks.

Table of Contents

Trademarks

1 Introduction

The following two sections cover the idea behind using the sensor controller engine especially in Building Automation applications, and also briefly presenting our wireless MCUs (Sub1GHz and/or 2.4GHz bands) that featured this engine.

1.1 Sensor Controller in Building Automation

The Sensor Controller is a programmable, autonomous ultra-low power CPU with fast wake-up capability and very low current sensor readings. Most building automation systems needs to wake up and execute a small task many times per second, the startup and shut down energy can easily be the dominating factor for the total energy spent for the application. A large high speed MCU system usually needs a lot of modules/routines that increases the energy consumption dramatically when changing state from standby to active modes. For example, a larger MCU system can require a much higher capable PRCM (Power and Clock Module) system. To solve this problem, TI introduced the Sensor Controller engine that can wake up from standby – execute a task and go back to standby – by using a little energy as possible. This application note explores the versatility of the Sensor controller, an ultra-low-power auxiliary processor featured in the CC13x2/4 and CC26x2/4 wireless devices with a focus on building automation applications.

1.2 TI Devices

1.2.1 CC13x4 Wireless MCUs

The SimpleLink™ CC13x4 device family is a low-power, Sub-1GHz (or multi-band Sub1-GHz and 2.4GHz with CC1354) wireless microcontroller (MCU) targeting applications needing enhanced security, on-chip over-the-air update capability, and support for advanced applications or large wireless protocols. The device supports IEEE 802.15.4, IPv6-enabled smart objects (6LoWPAN), Wireless M-Bus, Wi-SUN, Amazon Sidewalk, mioty, and proprietary systems, including the TI 15.4-Stack (Sub-1GHz) and 2.4GHz protocols such as BLE 5, Zigbee and Thread.

In addition to the software compatibility, within the Sub-1GHz wireless MCUs, there is pin-to-pin compatibility from 32kB of flash up to 1MB of flash in the 7mm × 7mm QFN package for maximum design scalability. For more information on TI's Sub-1GHz devices, *Sub-1GHz products*.

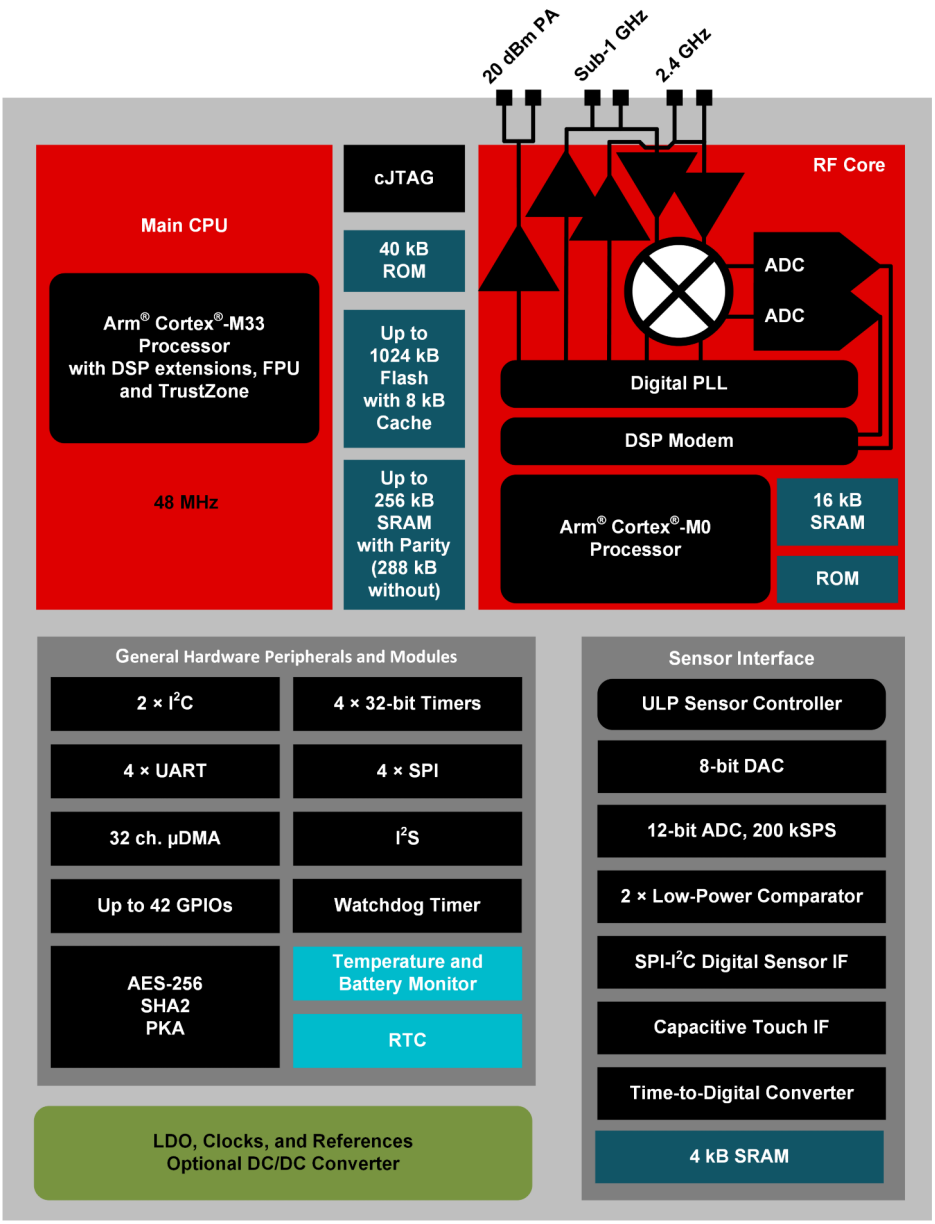

Figure 1-1. CC1354 Block Diagram

1.2.2 CC26xx Wireless MCUs

The SimpleLink™ CC26xx device family is a series of low-power multiprotocol 2.4GHz wireless microcontrollers (MCU) deigned for IoT applications. These devices support Thread, Zigbee, Bluetooth 5.3 Low Energy, IEEE 802.15.4, IPv6-enabled smart objects (6LoWPAN), proprietary systems, including the TI 15.4-Stack (2.4GHz), and concurrent multiprotocol through a Dynamic Multiprotocol Manager (DMM) driver. The device is powered by an ARM Cortex, which varies by model -M3, M4, with the latest devices featuring M33 processors. Memory configurations range from 128kB Flash/28kB RAM to 1MB Flash/296kBRAM).

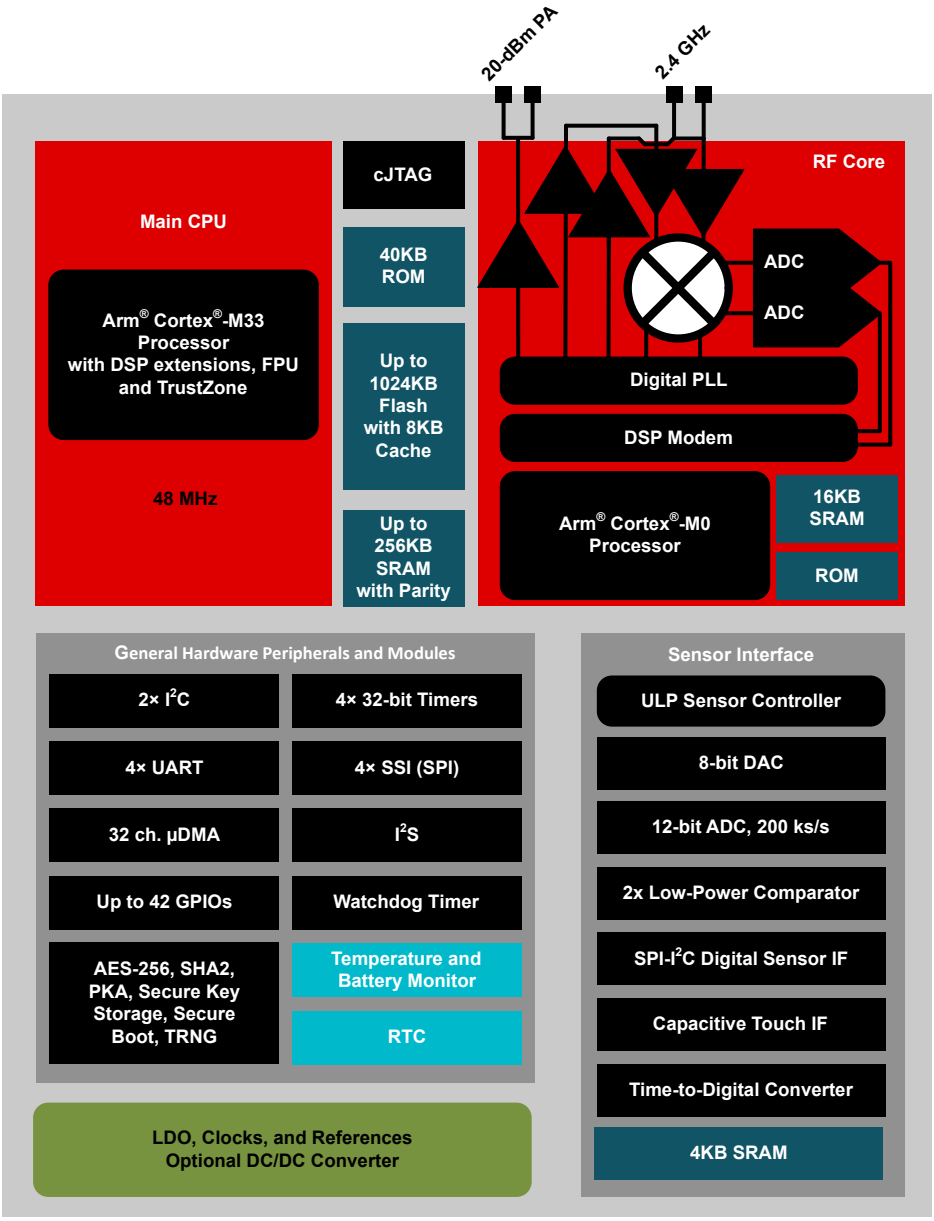

Figure 1-2. CC2674P10 Block Diagram

The CC13xx devices are part of the SimpleLink™ MCU platform, which consists of Wi-Fi®, Bluetooth® Low Energy, Thread, Zigbee®, Sub-1GHz MCUs, and host MCUs that all share a common, easy-to-use development environment with a single core software development kit (SDK) and rich tool set. A one-time integration of the SimpleLink™ platform enables you to add any combination of the portfolio's devices into your design, allowing 100 percent code reuse when your design requirements change. For more details, see the SimpleLink MCU platform.

2 Sensor Controller

The Sensor Controller was specifically designed with low power applications in mind – giving developers the ability to create smart sensors that run for years on a coin cell battery. The Sensor Controller is programmable and allows users to read and process data to make low level decisions while the rest of the system sleeps. The Sensor Controller can then wake up the system to perform more computationally-intensive tasks or transmit a message with the radio.

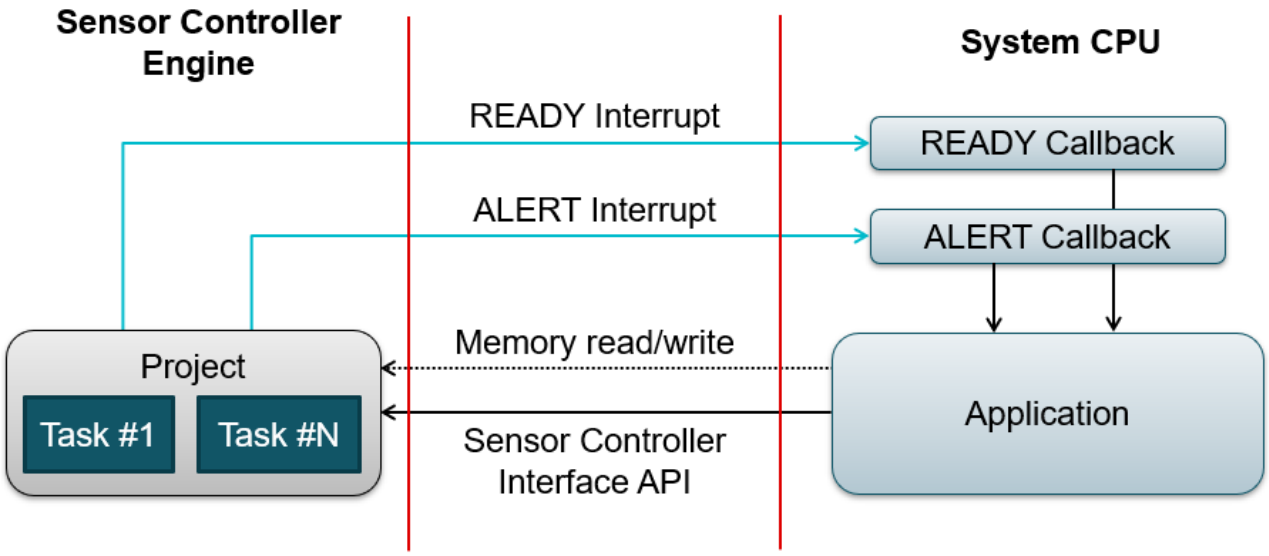

Figure 2-1. System CPU and Sensor Controller Interaction

The Sensor controller can implement tasks that are running in serial, while each task algorithm is divided into four types of code blocks:

- **Initialization Code**: Runs one time when the task is started through the task control interface
- **Execution Code**: Runs each time the task is scheduled for execution (based on periodic ticks from the real-time counter (RTC))
- **Event Handler Code**: Runs one time when the trigger that was set up occurs, (for example, an edge or level on an AUX I/O pin, or after a variable delay)
- **Termination Code**: Runs one time when the task is stopped through the task control interface

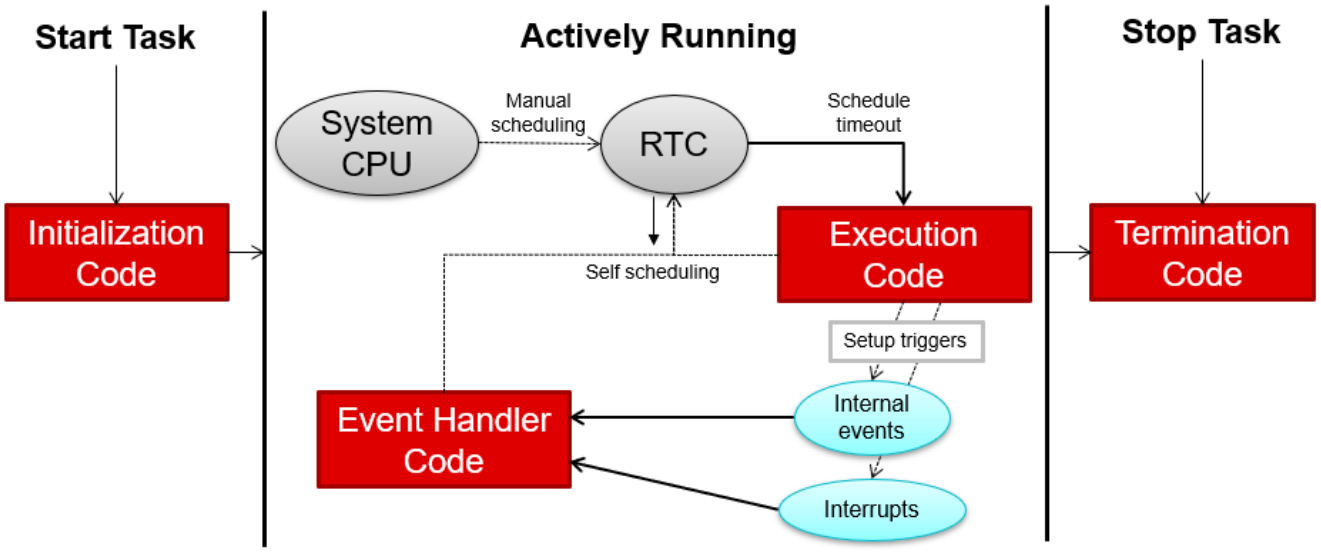

Figure 2-2. Task Execution Flow

2.1 Features

The Sensor Controller has access to both analog and digital peripherals which include:

- 12-bit ADC capable of sampling analog capable I/Os or internal chip voltages.
- Two comparators: one high speed continuous time comparator (CompA) and one low-power clocked comparator updated at 32kHz (CompB).
- 8-bit reference DAC capable of supporting the comparators with reference voltages.
- SPI master interface.
- Bit-banged serial interfaces including I2C master, UART and more.
- Time to digital converter capable of measuring the time between configurable start and stop triggers.
- Programmable current source capable of giving out currents between 0 and 20µA.
- Two simple 16-bit timers.
- Timer with PWM and four capture/compare channels allowing event setting and clearing without waking up the Sensor Controller.
- 16-bit asynchronous multipurpose timer with 4 capture or compare channels.
- Ultra low power 16-bit pulse counter capable of count rising edges on any digital input pins or comparator output.
- Access to all GPIO pins.

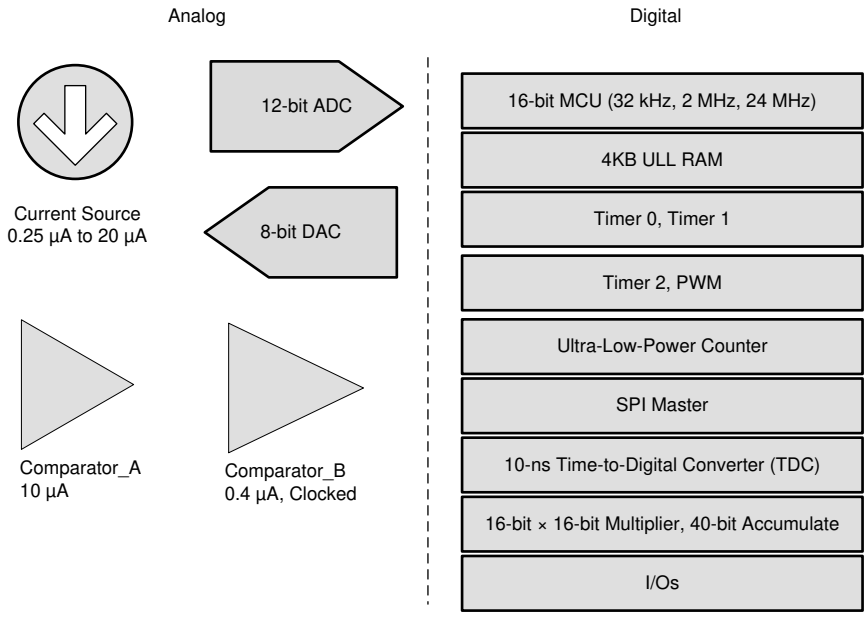

Figure 2-3. Sensor Controller peripherals

2.2 Sensor Controller Power Modes

The Sensor controller operates in three modes: an active (24MHz) mode, a low-power (2MHz) mode and a standby mode. The Sensor Controller is in active mode or low-power mode when running task code, and in standby mode otherwise.

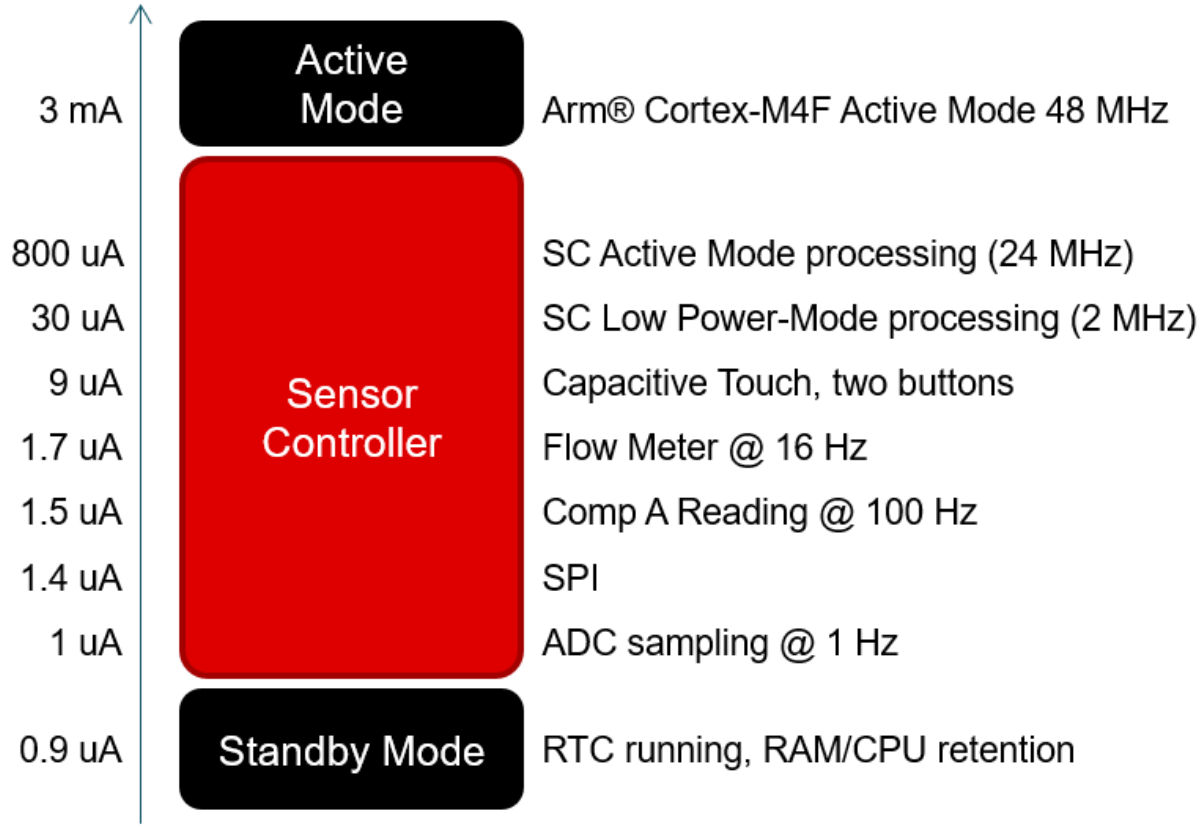

Figure 2-4. Sensor Controller Power Consumption Modes versus Arm Cortex M4F

The Sensor Controller can run from two different internal power sources – from the main power management system or from the power scheme that is used during standby.

The main power system is designed to deliver tens of milliamps and can power the complete IC, but has the drawback of using more power during the transition from and back to standby mode. In active (24MHz) mode, this system is needed for operation.

The standby power scheme running in standby, but can also power the sensor controller in the low-power mode. Since there is no need to change the mode of operation and that there is a dedicated 2MHz oscillator in the system (as opposed to a higher frequency divided down design) the low-power (2MHz) mode of operation is designed for ultra low power operations.

2.2.1 Active Mode

In active mode (24MHz), the Sensor controller operates at a maximum speed of 24MHz, enabling the controller to handle more computationally demanding tasks. While the power consumption and wake-up time is higher in this mode, this is relatively more efficient compared to using the main CPU for similar tasks. All peripheral modules are available in active mode, unless the sensor is used by the System CPU.

2.2.2 Low Power Mode

In low-power mode (2MHz), the Sensor Controller runs at a reduced clock speed of 2MHz. Peripheral modules that require active mode (SCLK_HF and/or other system functionality) are not available. Unavailable peripherals include:

- 12-bit ADC
- Programmable current source (0.25µA to 20µA)
- Time to digital converter

However, the low-power mode offers the benefit of lower power consumption and faster wake-up times while still maintaining functionality for simpler operations.

As a substitute design to the 12-bit ADC, a 8-bit, successive-approximation (SAR)-type, low-power, software ADC can be implemented using the Sensor Controller running in 2MHz mode. This design is discovered in details later in this document.

The Sensor controller can change power mode dynamically from task code and transition between active and low-power modes, creating balance between performance and power consumption.

For example, one can implement a code which allows entering Active Mode only during some wakeups:

```
// The task code block starts in low-power mode

// If some condition is met ...
if (...) {

// Enter active mode
pwrRequestAndWaitForActiveMode();

// Get one ADC sample
adcEnableSync(ADC_REF_FIXED, ADC_SAMPLE_TIME_2P7_US, ADC_TRIGGER_MANUAL);
adcGenManualTrigger();
adcReadFifo(output.adcValue);
adcDisable();

// Return to low-power mode
pwrRequestAndWaitForLowPowerMode();
}

// The task code block ends in low-power mode, so unless changed by the static configuration,
// the Sensor Controller can start in low-power mode at the next wake-up
```

The procedures used for the previous code are *pwrRequestAndWaitForActiveMode()* - which requests change to the active power mode, and waits for the change to take effect - and *pwrRequestAndWaitForLowPowerMode()* - which requests change to the low power mode, and waits for the change to take effect.

2.2.3 Standby Mode

In Standby mode, the Sensor Controller is essentially inactive, but preserves the state (RAM retention) while minimizing power.

The term *Standby* is used for the mode where the complete MCU system is *paused*, but with a real time clock running in addition to having memory retention for the complete MCU RAM. This means that the MCU system can continue to operate where it was before standby was entered. Note that some MCU manufacturers are stating their Standby consumption with almost no RAM retention and this can mean that much more time and energy is spent to start operating again.

The CC13xx family is using TI designed 65nm process technology that offers ultra low leakage transistors. These transistors are used in modules where ultra low power is important – for example in the sensor controller MCU system and for most of the RAM blocks. This dedicated process technology can also offer low power even at elevated temperatures.

A key power-saving technique using this feature is sensor sequencing which enables a sensor at a designated time period before the sensor is needed for sampling allowing the Sensor Controller to stay in the lowest power state while the sensor boots and activate only when the sensor is ready.

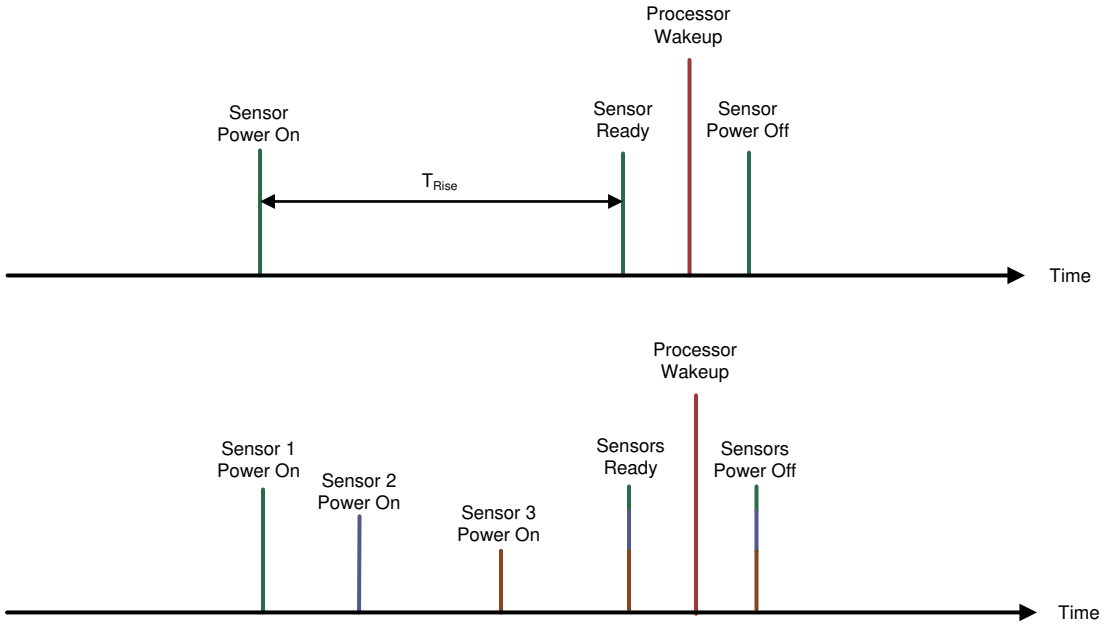

Figure 2-5. Sensor Sequencing with One and Multiple Sensors

For more information, please check *Sensor Sequencing Using the CC13x2 and CC26x2 Sensor Controller*.

2.2.4 Switching Between Power Modes

As mentioned in the beginning of this chapter, the sensor controller can operate at two speeds: **2MHz** (Low power mode) and **24MHz** (Active mode). The 2MHz mode can offer the lowest power in terms of startup efficiency, but has a also some limitation with respect to the number of peripherals available. When the sensor controller is in Standby mode (essentially inactive, but preserves the state (RAM retention) while minimizing power), the system can wake up either to the 2MHz mode or to the 24MHz mode. To measure the current that is drawn due to the wake up process (in other words, the *extra* energy that is not really contributing towards the application, but only due to the wake up process) we have chosen a very simple, but also measurable application: Wake up from standby -> toggle a pin and go back to standby – and repeat this 100 times per second (100Hz). Given the simplicity of the application, this does not not contribute much towards the total consumption, so more or less only measure the *extra* energy that needs to go from standby to an active mode (24MHz) and then return to standby.

2.2.4.1 24MHz - Startup From Standby and Return to Standby Energy

The below figure is zoomed in to one of the transitions (green shows the pin toggle, yellow is the current consumption and time is on the x-axis) The energy spent for the measured by the instrument to be 145nC. If using this to estimate the total consumption, the response is 100Hz×145nC = 14.5uA. The measurement from the instrument is showing an average current consumption of 13.5uA, so using the start up energy measurement per wake up gives a pretty good estimate in this case.

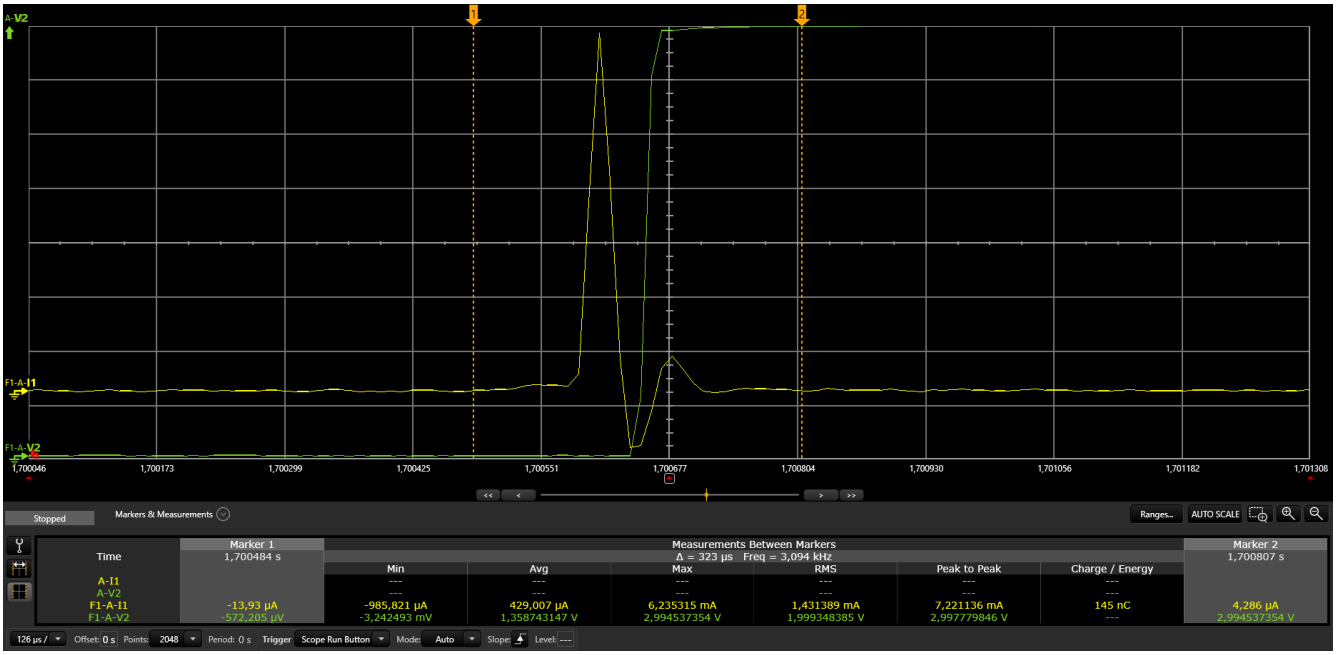

Figure 2-6. 100Hz Pin Wake Up Test Case in 24MHz Mode

2.2.4.2 2MHz - Startup From Standby and Return to Standby Energy

Figure 2-7 is zoomed in to one of the transitions (green shows the pin toggle, yellow is the current consumption and time is on the x-axis) The energy spent for the measured by the instrument to be 8nC. If we use this to estimate the total consumption this receives 100Hz×8nC = 0.8uA. The measurement from the instrument is showing an average current consumption of 867nA, so using the start up energy measurement per wakeup gives a very good estimate in this case.

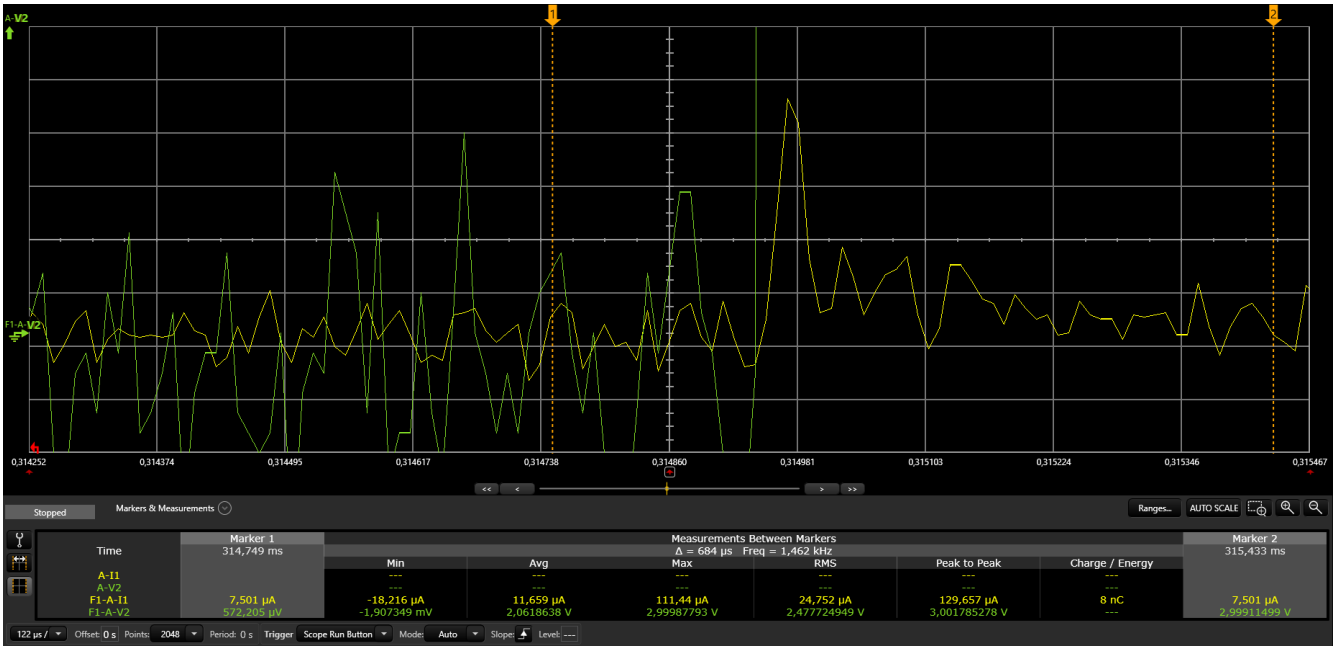

Figure 2-7. 100Hz Pin Wake Up Test Case in 2MHz Mode

2.3 Power Measurement Setup

All the measurements mentioned in this App note are performed using the LAUNCHXL-CC1352P Rev. B board equipped with the necessary Booster packs or boards needed, but applicable for all devices with Sensor Controller. Any additional equipment used is specified in the corresponding application.

2.3.1 EnergyTrace™ Software

For current measurements, traditionally power/current consumption measurements are performed using a power analyzer, which provides detailed insights but often involves expensive, bulky and not always available equipment. An example for how measurements can be done using a power analyzer on the LAUNCHXL-CC1312R1 are shown in Ultra-Low Power Sensing Applications With CC13x2/CC26x2.

Another design is the EnergyTrace™ software which is an energy-based code analysis tool included in Code Composer Studio™ (CSS) IDE version 6.0 and later. EnergyTrace™ simplifies the process significantly allowing developers to measure and analyze real-time energy consumption directly from the board during development. The EnergyTrace™ provides a live tracking of the energy and current signals and provides the mean values for power, voltage and current during the program run as well as a battery life estimation.

2.3.2 Software

The following software is used:

- Sensor Controller Studio 2.9.0.208
- Code Composer Studio 12.7.1
- SimpleLink CC13x2 / CC26x2 SDK Version 7.40.00.77

2.3.3 Current Consumption Measurements

For each of the measurements, the XDS110 debugger jumpers are initially kept on the LAUNCHXL-CC1352P board. The relevant application software is then flashed on the CC1352 using the CCS IDE. Once completed, the board is is disconnected from the PC device and all the jumpers on the Launchpad are removed except for the XDS110 *GND*, *3.3V*, *RXD<<* and *XDS110 Power* jumpers as these are crucial for the functionality of the EnergyTrace™. The board is then reconnected to the PC device using the USB micro and EnergyTrace™ is started.

Each reported current value is the average of 10 measurements on four boards, for a total of 40 measurements and is the average of a 30 sec time window run of the program specified. All measurements were done in a room temperature environment.

The application note uses the following software and hardware to find the current consumption of the Sensor Controller:

2.3.4 Hardware

The following hardware is used:

- *LAUNCHXL-CC1352P*, Kit Rev B
- *BOOSTXL-ULPSENSE*, Kit Rev A
- *Door and Window Sensor Evaluation Platform Introduction and Performance Overview*, application note

3 Building Automation Use-Cases and Techniques using Sensor Controller

The Sensor Controller provides a platform for developing energy-efficient building automation designs. The ultra-low-power operation and ability to handle real-time data collection makes the Sensor Controlled well-designed for key use cases such as PIR based motion detection, glass break detection, door and window and sensing:

- **Motion Detection - Analog, Digital PIR Sensor:**

The Sensor Controller is able to periodically sample data from a PIR sensor at frequencies designed for the application, apply digital processing methods to refine the received signal and use basic algorithms (threshold based as well as other methods) to detect motion. This allows the main CPU to remain in sleep mode until motion is detected.

- **Door/Window Sensors - Reed Switch and Magnet Sensors:**

The Sensor Controller is able to periodically check the reed switch's state, filter out noise or any false triggers (for example through debouncing the switch) making sure that only a legitimate event is processed. The Sensor Controller can wake up the System CPU when detection occurs while the switch operates in low-power mode during most of the routine.

- **Glass break detection - Microphone, Piezoelectric Sensor:**

For glass break detection, the Sensor Controller can monitor audio or vibration signals using a microphone, or measuring pressure an external piezoelectric pressure sensor. Detection can then occur based on analyzing incoming signals for specific frequency patterns indicative of breaking glass. Additional processing can be further handled by the main processor if needed.

The following chapters demonstrate the implementation of each of these applications using the Sensor Controller with a focus on low power consumption for longer battery life.

3.1 PIR Motion Detection

There are several different types of sensors that can be used to detect basic motion, but the most common design for the last decade has probably been using PIR sensors. PIR sensors are based on Wien’s displacement law, which states that black-body radiation curve for different temperature can peak at different wavelengths that are inversely proportional to the temperature. Basically, to monitor the infrared spectrum, objects of different temperature can radiate different levels of energy. Figure 3-1 what is seen in images taken with infrared camera:

Figure 3-1. Infrared Images

3.1.1 PIR Traditional Signal-Chain

In a traditional signal chain of PIR Motion detection, a signal from the PIR sensor is usually fed through a series of gain band-pass filters, that usually include DC blocking caps, and then fed to a set of comparators for low and high side waveform detection, acting as a window comparator.

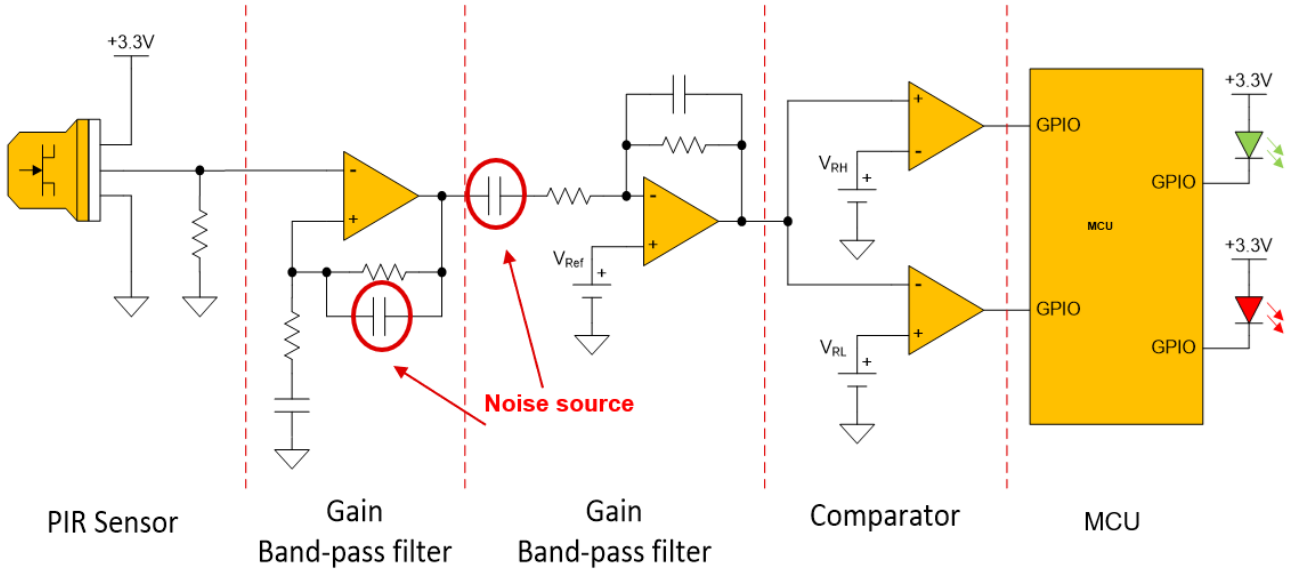

Figure 3-2. PIR Traditional Signal-Chain

One of the main disadvantages of using pure hardware components for the above, is that the signal chain is fixed and limited. For the band-pass filters, the gain and cutoff frequencies are configured specifically for a particular detection range and motion speed. Typical cutoff frequencies are around .7 – 30Hz and overall signal gain can be as high as 1000x. Another downside of this signal chain we found during our investigation, is that these DC filter caps actually end up being very large noise sources for the signal chain. Ceramic surface mount capacitors are usually made of barium titanate, which has a piezoelectric effect, meaning any noise or vibration actually generates small noise signals on the caps. Tantalum capacitors at such low frequencies can also introduce noise onto the signal. Coupling this noise with up to 1000x gain in the signal chain can return a very poor signal to noise ratio.

3.1.2 Capacitor-less Motion Detection Block Diagram

The block diagram below demonstrates the implementation of motion detection using the Sensor Controller engine. The block diagram features a straightforward signal chain for an analog PIR sensor, incorporating one operational amplifier (op-amp) for buffering the signal and another for amplifying the PIR signal. This capacitor-less design effectively eliminates the DC component of the PIR signal by utilizing the internal DAC of the Sensor Controller. The DAC stabilizes to the DC value of the PIR signal, updating every few seconds to make sure accurate detection and enhanced system performance.

The threshold detection and movement frequencies can be adjusted through software, allowing for flexible and customizable motion detection. When motion is detected, the Sensor Controller wakes up the main ARM CPU, which then activates the radio to transmit a signal over Sub-1GHz or 2.4GHz. In the absence of motion, the main CPU remains in deep sleep mode, making sure ultra-low power consumption and extended battery life. This design efficiently balances performance and energy efficiency, designed for low-power motion detection applications.

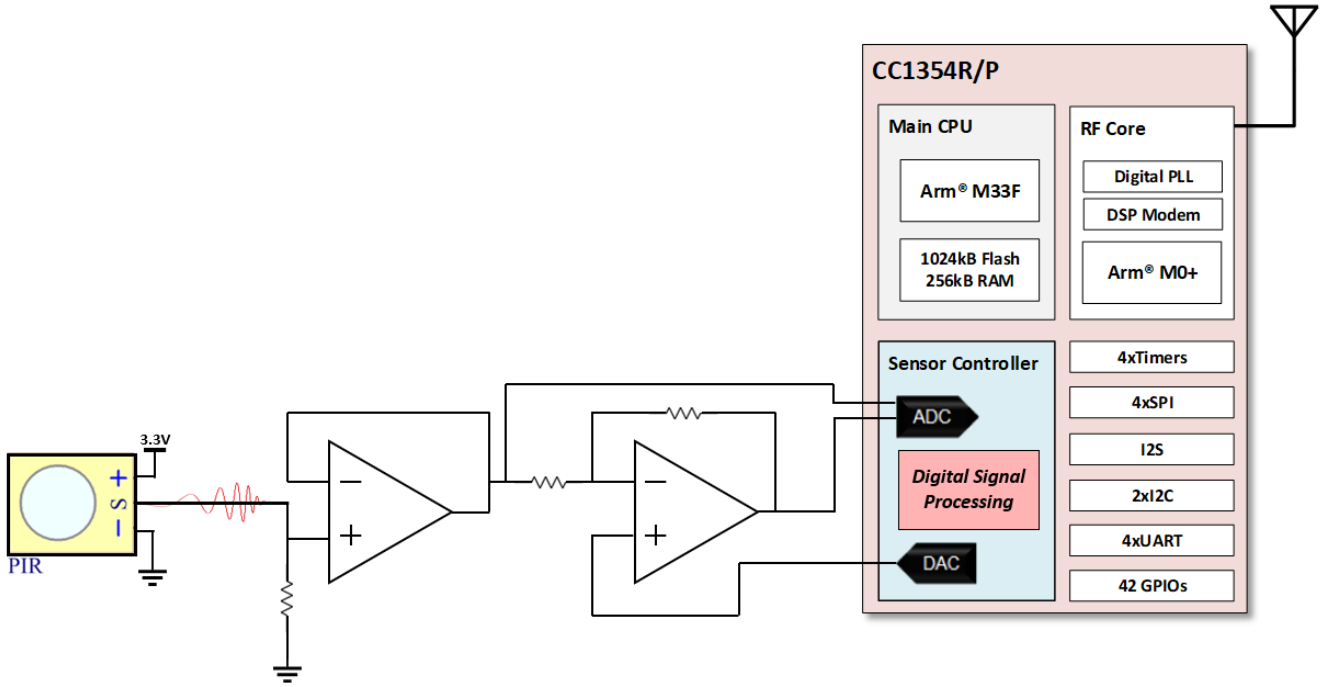

Figure 3-3. Capacitor-less PIR Motion Detection Block Diagram

3.1.3 Digital Signal Processing

The following section elaborates on the digital signal processing implemented for the PIR motion detection. This process is designed to optimize the sensitivity and accuracy of the system while minimizing noise and false triggers.

3.1.3.1 Hardware

- CC1352P Sensor Controller
- Murata IRA-S210ST01 Analog PIR Sensor
- Fresnel Lens IML-0685

The signal chain we used for this Motion Detection is mostly purely buffering and gain stages. To remove DC blocking capacitors in the traditional circuit that were adding noise to the system, the stages are being biased using DACs. Instead of using comparators, the signals are fed into an analog-to-digital converter (ADC) which enables a feedback loop for the DAC for tracking changes to environment and ambient temperature and allows digital filtering of the signal.

3.1.3.2 Digital Signal Processing

Figure 3-4 shows what the PIR signal looks like over time as the ambient temperature and the PIR sensor body temperature are fluctuating. There is no motion detect events here, just the signal drifting.

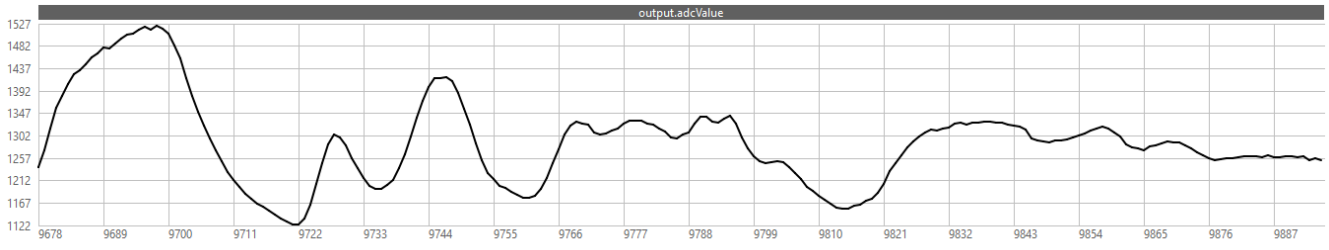

Figure 3-4. PIR Signal over time

The PIR raw signal can be somewhat noisy due to environmental changes, such as temperature fluctuations or background interference. As a result, avoid using simple threshold on the raw signal because this can fluctuate

up and down, leading to unreliable detections. Instead, we analyze the signal's first derivative to measure how quickly the signal rises over time. A rapid change in the signal results in a high first derivative, which we then threshold to detect movements more reliably. Before applying this approach, we oversample the raw signal and use a moving average filter to smooth out small spikes.

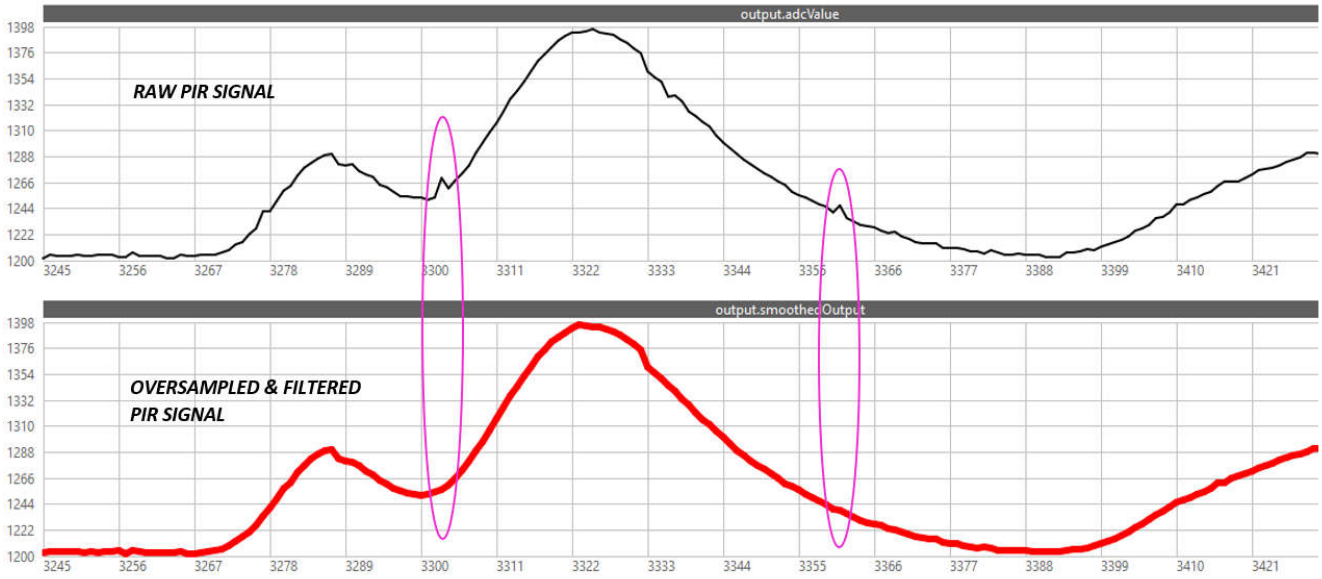

Figure 3-5. Smoothed PIR Signal Versus Raw Signal

After smoothing the raw PIR signal, calculate the absolute value of the first derivative. This step allows the focus on the magnitude of changes in the signal, regardless of the direction of the variation. To detect movement, we set a software-defined threshold on this absolute first derivative. If the magnitude of the derivative exceeds the threshold, this indicates a rapid change in the signal, which corresponds to motion. This method provides a robust way to detect movements while minimizing false triggers caused by gradual signal fluctuations or environmental noise.

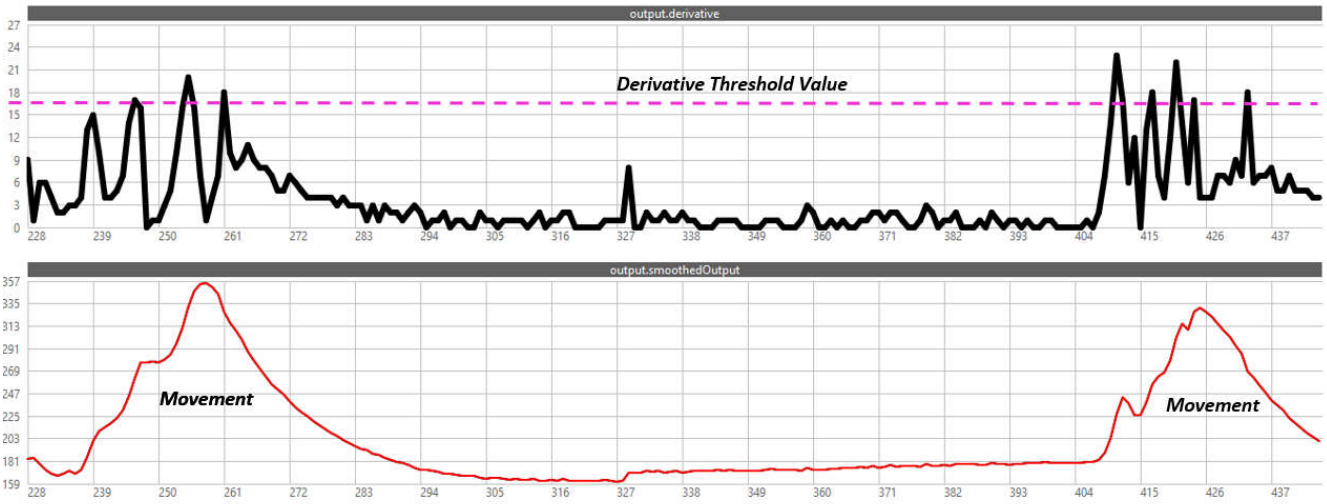

Figure 3-6. First Derivative of the Smoothed Signal

3.2 Glass Break Detection

For reliable glass break detection in building automation, various sensor-based methods can be used to identify the unique signals associated with glass breakage. By combining sound and vibration detection, or using advanced frequency filtering, these methods help make sure accurate, responsive monitoring with minimal

false alarms. Common approaches include using piezo sensors to detect impact vibrations, microphones for audio pattern recognition, and even accelerometers for multi-axis vibration sensing. Each method offers unique advantages and can be tailored to be designed for the specific security needs and power constraints of automated building systems.

3.2.1 Low-Powered and Low-Cost Glass Break Block Diagram

Figure 3-7 illustrates how the internal peripherals of the CC13xx are utilized to design a simplified, ultra-low-power glass break detector with an external piezoelectric sensor. For this use-case we used the internal Comparator B together with a reference DAC and ADC. The COMPB peripheral is a low-power clocked comparator that is updated at 32kHz. COMPB can be used to continuously monitor slow signals and wake up the Sensor Controller from standby mode. Monitored signals include but is not limited to power supply voltages or analog sensor outputs. In Sensor Controller Studio, the threshold voltage of the COMPB peripheral can be configured using the internal reference DAC. Once the piezoelectric sensor signal surpasses the configured threshold, the system samples the entire piezo signal over a dedicated time using the internal ADC. This sampled data is then used to calculate the signal's energy, allowing differentiation between a knock on the glass and an actual break.

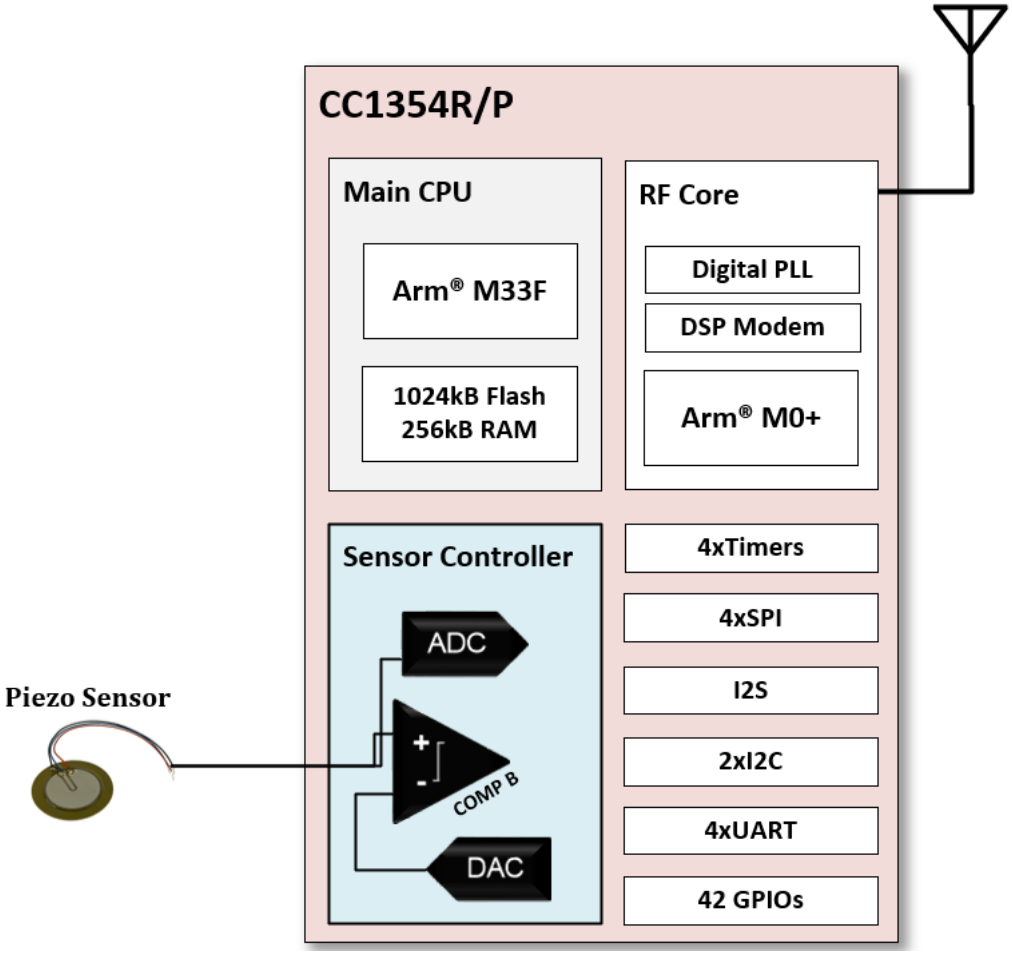

Figure 3-7. High-Level Glass Break Detection Block Diagram

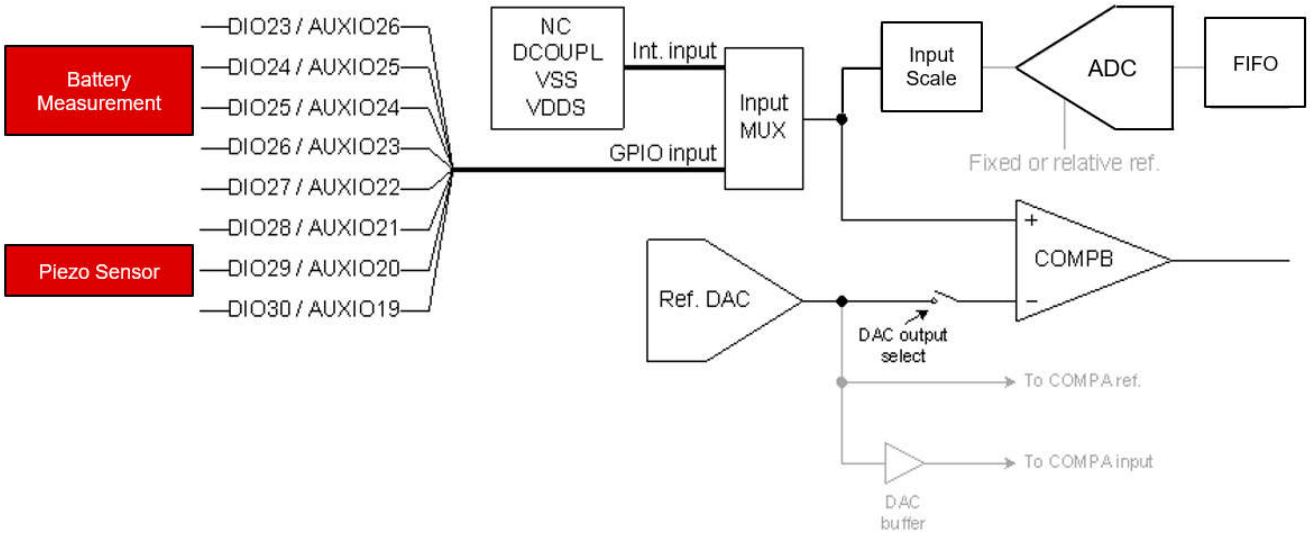

Figure 3-8. Internal Connection of Sensor Controller Peripherals

Figure 3-9 was generated by connecting a piezo sensor in the topology that is mentioned previously. By vibrating the board with the Sensor Controller + piezo, we can simply plot different voltage spikes that are reaching between 0.5V to 5V (depending on how large is the resistor you load the piezo sensor with). Then by analyzing the signals and the energy, we can understand what is considered as a *break* and was is considered as a normal knock or vibration.

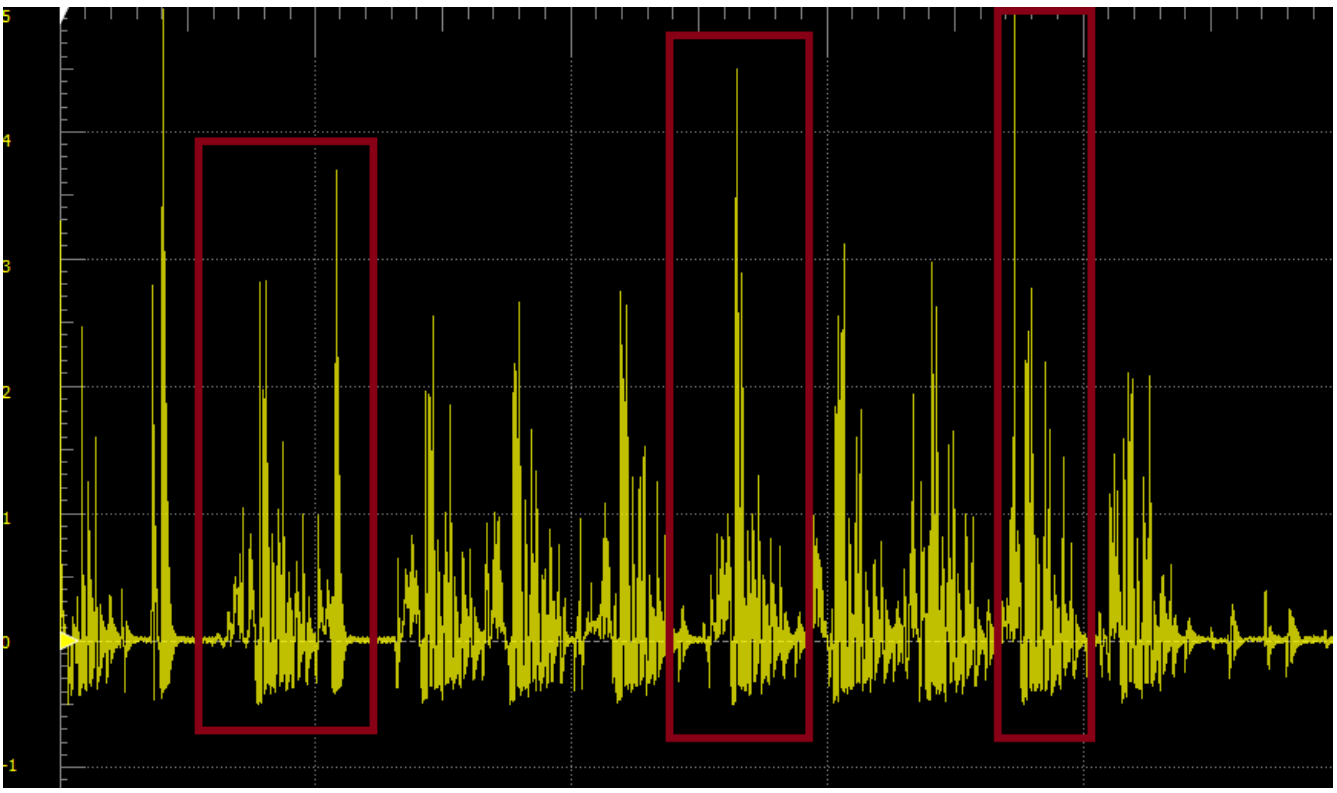

Figure 3-9. Piezo Sensor Signals

3.3 Door and Window Sensor

The Reed Switch is capable of detecting magnetic fields. Normally, the switch is open, but when a magnetic field is applied to the switch, the switch closes. The Reed Switch is connected in series with a pull-up resistor, thus, the REED signal can be active low. A reed switch can be used in numerous of applications; one example can be a door/window sensor. Figure 3-10 shows a detection of a door opening event.

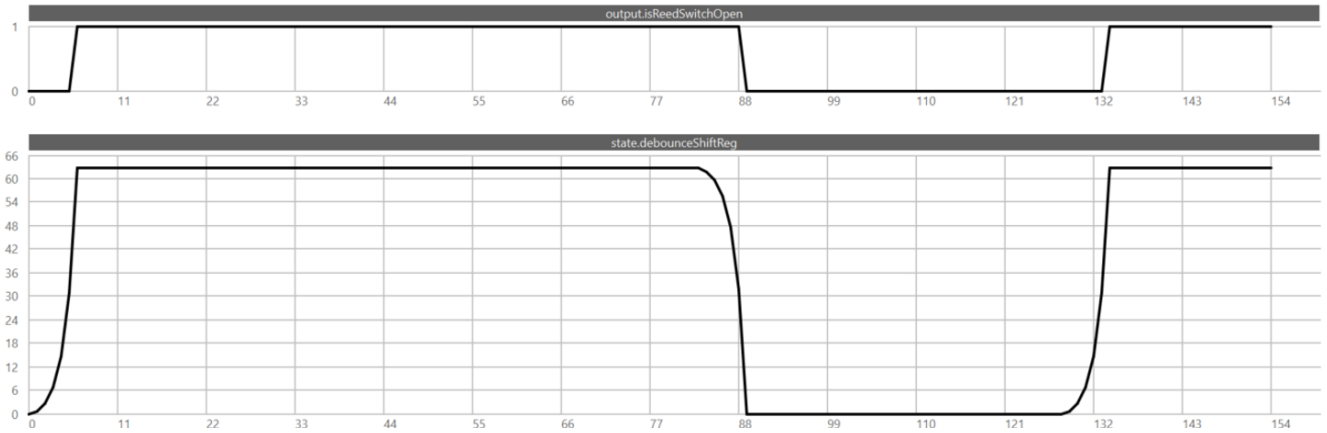

Figure 3-10. Detection of a Door Opening Event

The detection distance with the first magnet is maximum at the center at approximately 0.7 inches, with two lobes on either side of the center peak. The darker shaded red areas indicate detection overlap in the Z axis which are clearly displayed in the following 3D figure:

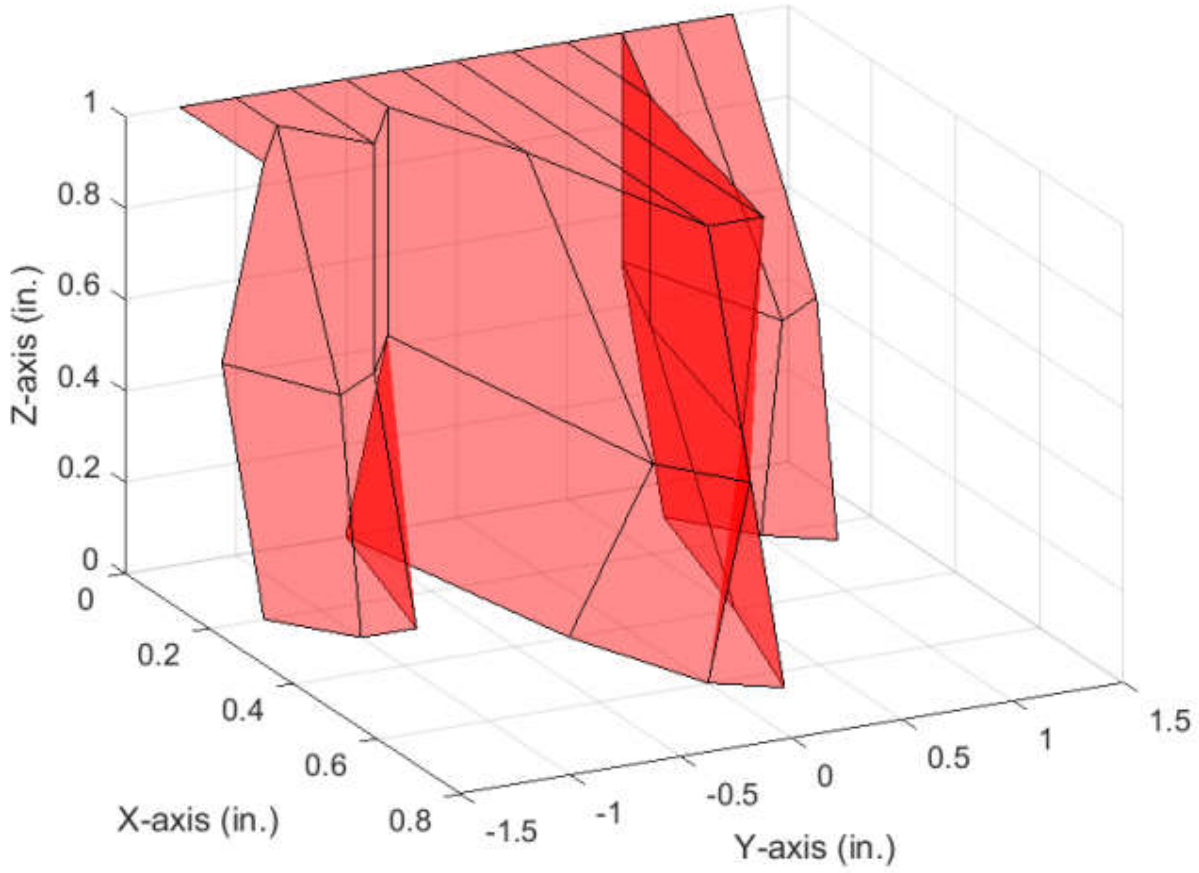

Figure 3-11. Reed Switch Magnet 1 Detection Field (3D)

3.4 Low-Power ADC

As mentioned previously, the 12bit, analog-to-digital converter (ADC) peripheral block included in CC13x2 and CC26x2 devices, which is typically used by the Sensor Controller, requires the SCLK_HF signal, and is therefore not available in low-power (2MHz) mode. The goal of this section is to highlight an alternative design creating an 8bit, successive-approximation (SAR)-type, low-power, ADC using the CC13x2/4 and CC26x2/4 Sensor Controller running in 2MHz mode. In this design, the comparator and digital-to-analog converter (DAC) peripherals are used which are available in low-power mode.

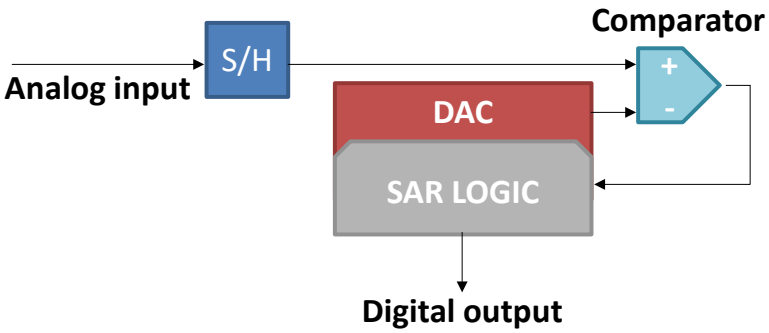

Figure 3-12. SAR ADC Block Diagram

The SAR ADC consists of four major blocks:

- **Sample and hold (S/H)** block locks the analog voltage level during sampling.
- **SAR logic** block is replicated using the Sensor Controller.
- **DAC** whose output is controlled by the SAR logic block.
- **Comparator** compares the analog signal to the DAC output.

3.4.1 Code Implementation in Sensor Controller Studio

The following describes ADC code implementation in the Sensor Controller Studio.

Start with configuring the needed data structures and task resources:

Data structures:

- cfg
  - resolution = 8
- output
  - value
- state
  - isLarger
  - nextVal

Task resources:

- Analog Pins
  - PIN_ANALOG
- COMPA
- Reference DAC
- RTC-Based Execution Scheduling
- Delay Insertion

C Code

Initialization Code:

```
//Select SAR ADC input pin compaSelectGpioInput(AUXIO_A_PIN_ANALOG); fwScheduleTask(1);
```

Execution Code:

```
//Enable comparator compaEnable(COMPA_PWRMODE_ANY);

state.nextVal = 0x0080; // 1/2 VDD

output.value = 0; //Set the reference DAC output to 1/2 VDD
refdacStartOutputOnCompaRef(state.nextVal); refdacEnable(REFDAC_PWRMODE_ANY,REFDAC_REF_VDDS);

U16 currBit = cfg.resolution;
U16 step = 64;

while(currBit > 0){ //Update DAC value refdacChangeOutputValue(state.nextVal); //Wait for DAC to
stabilize...
//An additional slight delay for DAC and comparator to stabilize...
refdacWaitForStableOutput();

fwDelayUs(10); //Read comparator output compaGetOutput(state.isLarger); if(state.isLarger == 1){
state.nextVal += step; //Increase DAC value for next iteration
}
else{
state.nextVal -= step; //Decrease DAC value for next iteration
}

step >>= 1; //Divide step by 2
currBit -= 1;

//Update output value
output.value |= (state.isLarger << currBit);
}
refdacStopOutput();

compaSelectIntRef(COMPA_REF_VSS); //Disable peripherals
refdacDisable();
compaDisable();
fwScheduleTask(1);
```

Termination Code:

```
//Disable peripherals
refdacDisable();
compaDisable();
```

3.4.2 Measurements

The following measurements were done with an external DC power supply, EnergyTrace + CCS + Sensor Controller.

Table 3-1. Different Voltages Measurements of Both Software and Hardware ADCs

Voltage (V)	Hardware ADC	Software ADC (8 -bit)
1.0	0.949	0.800
1.5	1.421	1.190
2.0	1.891	1.590
2.5	2.359	2.000
3.0	2.829	2.400

Table 3-2. Software and Hardware ADC Power Consumption

	Current consumption	Unit	Batterylife (CR123)
Hardware ADC (1Hz)	1.3	µA	13 years
Hardware ADC (100Hz)	16.5	µA	1year and 4 months
Software ADC (1Hz)	0.8	µA	28years and 6 months

Table 3-2. Software and Hardware ADC Power Consumption (continued)

	Current consumption	Unit	Batterylife (CR123)
Software ADC (100Hz)	4.8	µA	4years and 8 months

3.5 Different Sensor Readings with BOOSTXL-ULPSENSE

The ULP Sense BoosterPack is a kit designed to make this easy to demonstrate the ultra-low power features of the sensor controller included in the CC13x2/CC26x2 RF SoCs. Examples of applications that can greatly benefit from this are Automated door and window sensors (based on a reed switch), Capacitive touch buttons and ultra-low power handling of accelerometers (for example fall detection).The board also contains an analog light sensor and a potentiometer that is useful for ADC test cases.

3.5.1 Capacitive Touch

As smart buildings and IoT-driven infrastructures gain traction, CapTouch offers an intuitive interface for controlling a wide range of systems such as lighting, HVAC and access control. For instance, in **lighting control** applications, CapTouch panels allow users to adjust brightness or schedule lighting changes with simple taps. In **HVAC systems**, CapTouch interfaces replace traditional thermostats with touchscreens that provide control over temperature, humidity and air quality settings. They can also be found in **access control** applications integrated into touch-sensitive keypads and biometric devices utilized for secure entry.

The Capacitive touch example on the BOOSTXL-ULPSENSE uses a small circular copper area on the top layer and a hatched ground plane on the bottom layer to create a small capacitor. The Sensor Controller is able to measure the capacitance using the ISRC (current source) and time-to-digital converter (TDC) peripherals. The capacitance of the system can change when touched with a finger tip. The capacitive touch example is tested both without any input to the cap touch buttons, and with the buttons activated. When no touch is detected on the buttons, the Sensor Controller wakes up with a frequency of 32Hz. If a button press is detected, the Sensor Controller increases the wake-up interval to around 100Hz. The Sensor Controller can also wake up the System CPU and notify of the touch event.

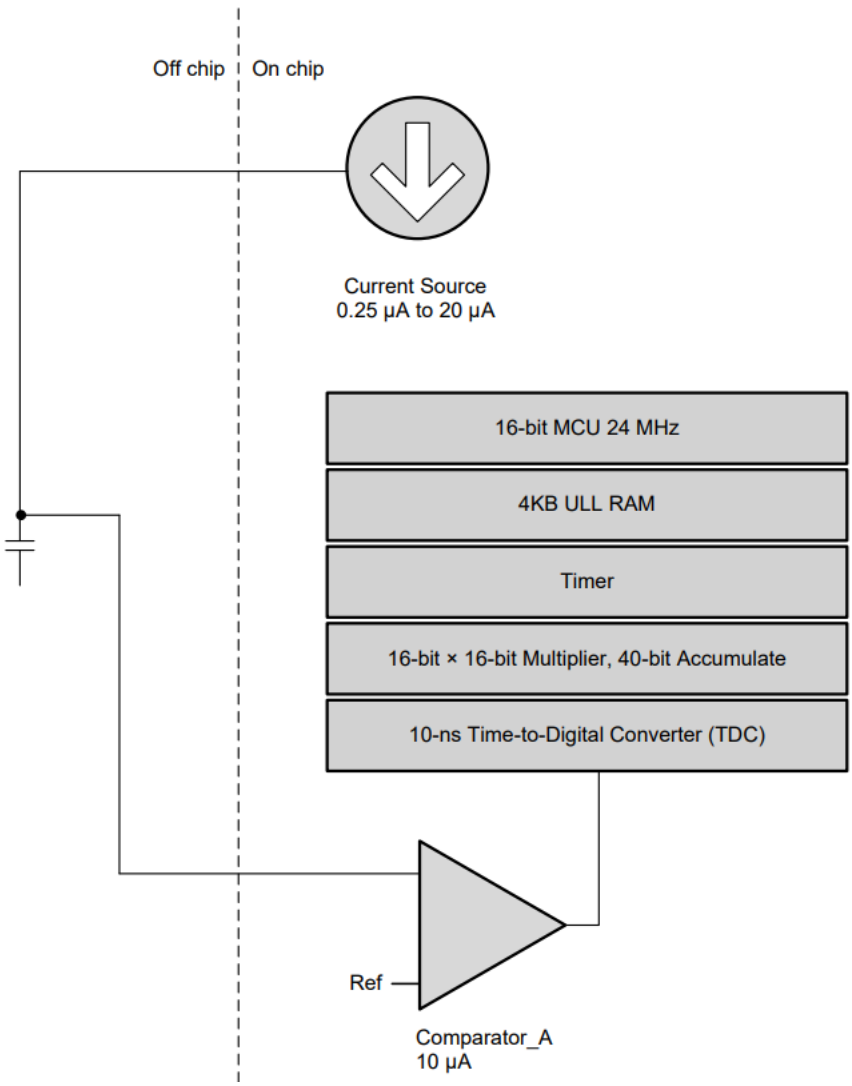

Figure 3-13. Capacitive Touch Principle (Simplified)

Table 3-3. Capacitive Touch Power Consumption

	Average Current Consumption	Unit	Battery life (CR123)
Without Touch (32Hz)	8.5	µA	2 years and 8 months
With Touch (approximately 100Hz)	84.1	µA	3 months

The following images show the current signal captured by EnergyTrace™ over the measurement period of 1 second as well as a single measurement.

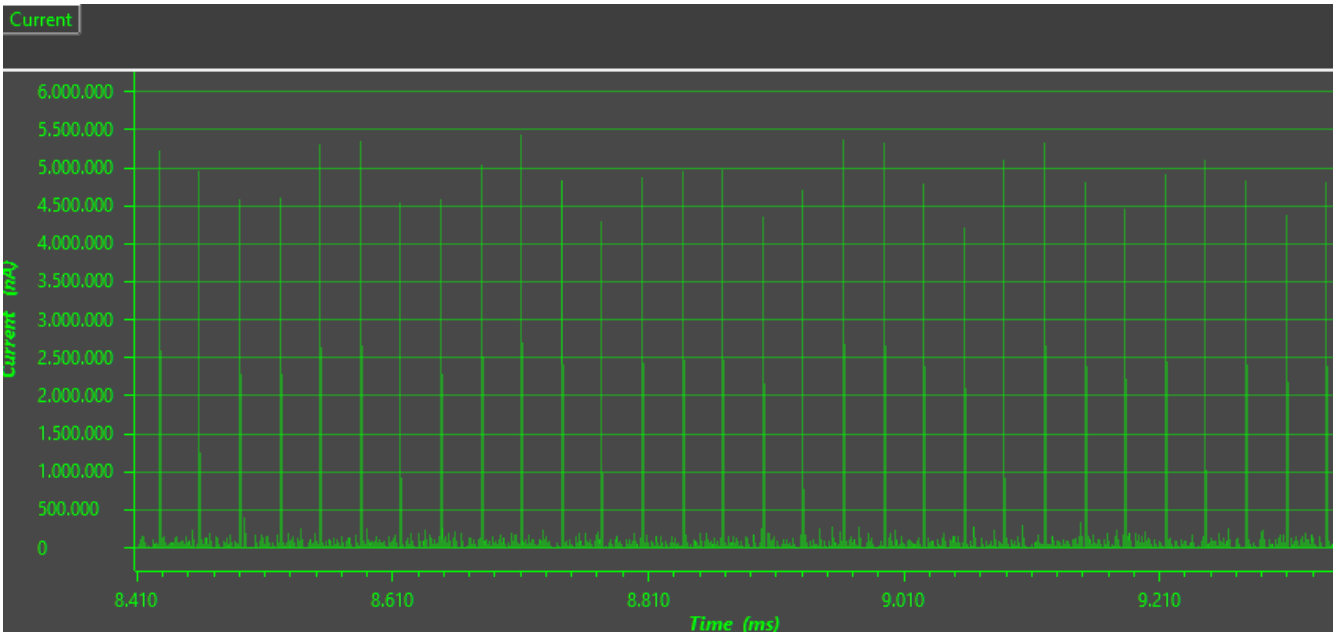

Figure 3-14. Capacitive Touch: Measurement Without Touch - 1 Second

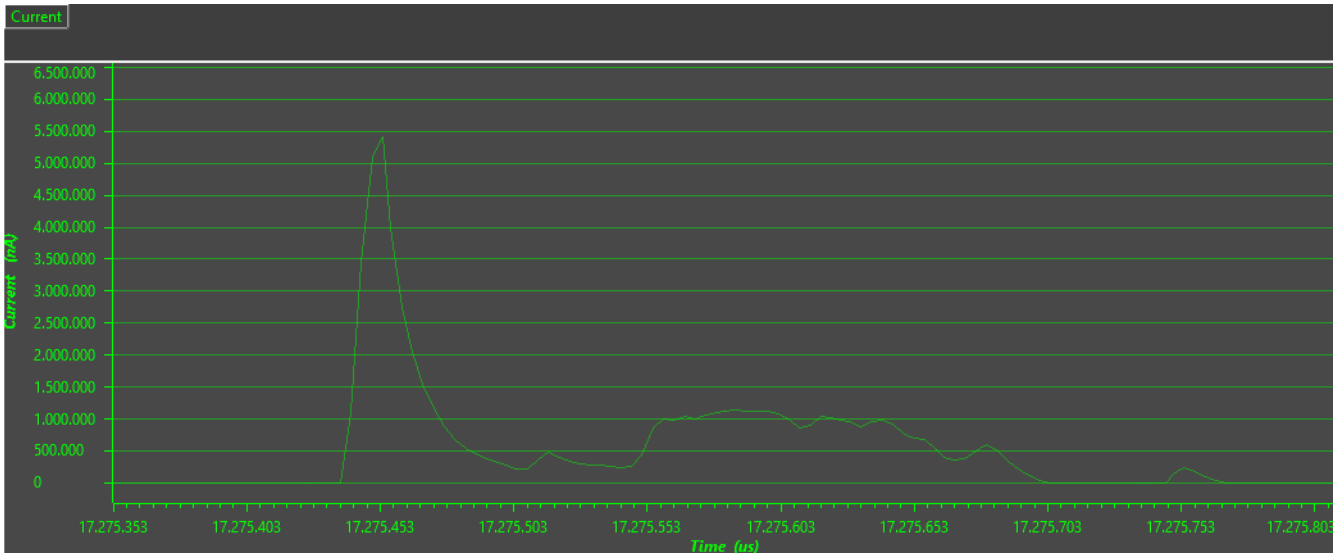

Figure 3-15. Capacitive Touch: Measurement Without Touch - One Measurement

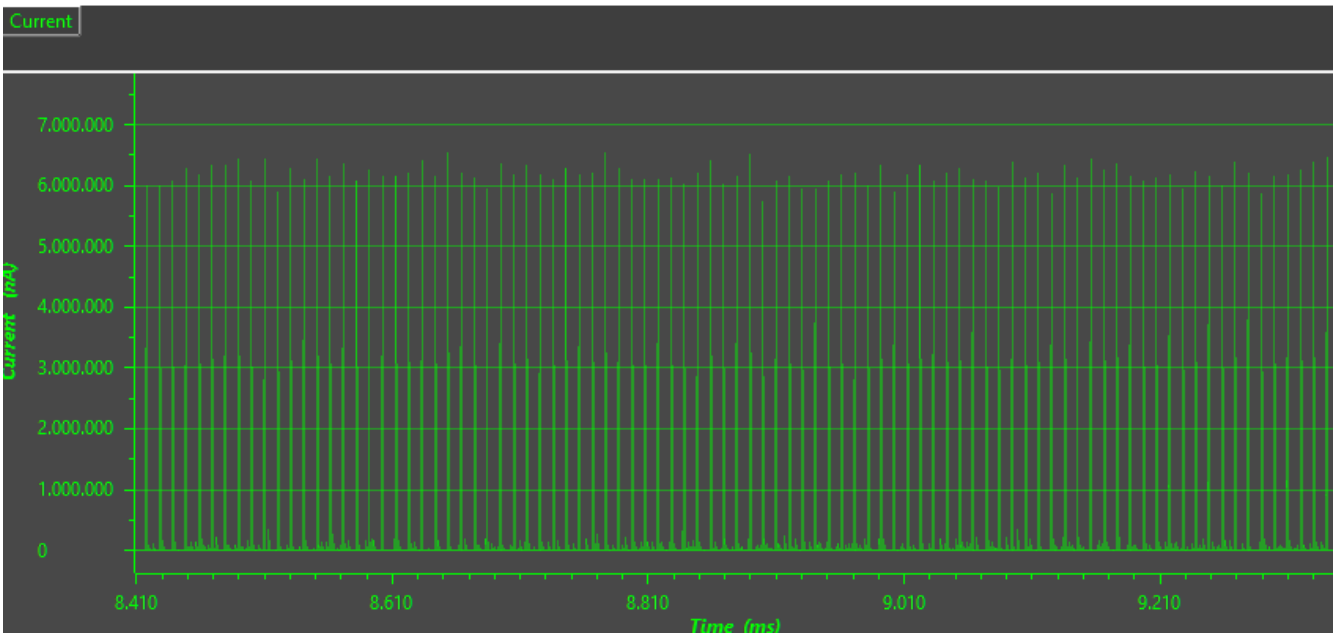

Figure 3-16. Capacitive Touch: Measurement With Touch - 1 Second

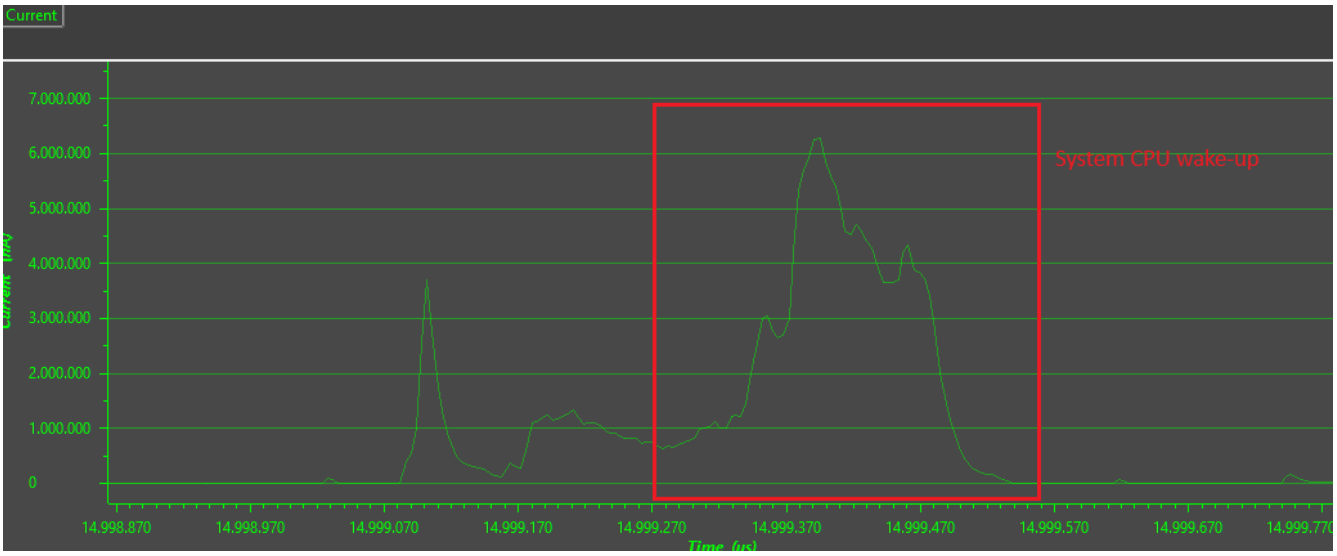

Figure 3-17. Capacitive Touch: Measurement With Touch - One Measurement

The 1 second measurement window shows the increased wake-up frequency described before with each spike (One measurement) representing a wake-up of the Sensor Controller.

3.5.2 Analog Light Sensor

Analog light sensors play a crucial role in building automation applications by providing real-time data on ambient light levels and thus enabling dynamic control of lighting systems and other environmental settings. One example can be in **lighting automation** where artificial light is adjusted based on the amount of natural light available (daylight harvesting). In **HVAC systems**, analog light sensors work alongside other sensors to detect sunlight intensity and adjust heating or cooling systems accordingly depending on the season and daytime.

The Analog Light Sensor outputs a current dependent on the amount of light the Sensor is exposed to. The current is fed into a resistor creating a voltage which can be captured by the ADC. The Sensor Controller uses the peripheral Timer 2 to enable power to the analog light sensor around 1ms before the Sensor Controller can wake up to take a measurement. If the light level is higher than a threshold, the Sensor Controller can wake up

the System CPU for processing of the measurement. This can repeat with a frequency of 10Hz. The example is tested with the same ambient light during the whole test period.

Table 3-4. Analog Light Sensor Power Consumption

	Average current consumption	Unit	Battery life (CR123)
Analog light sensor (10Hz)	9.1	µA	2years and 6 months

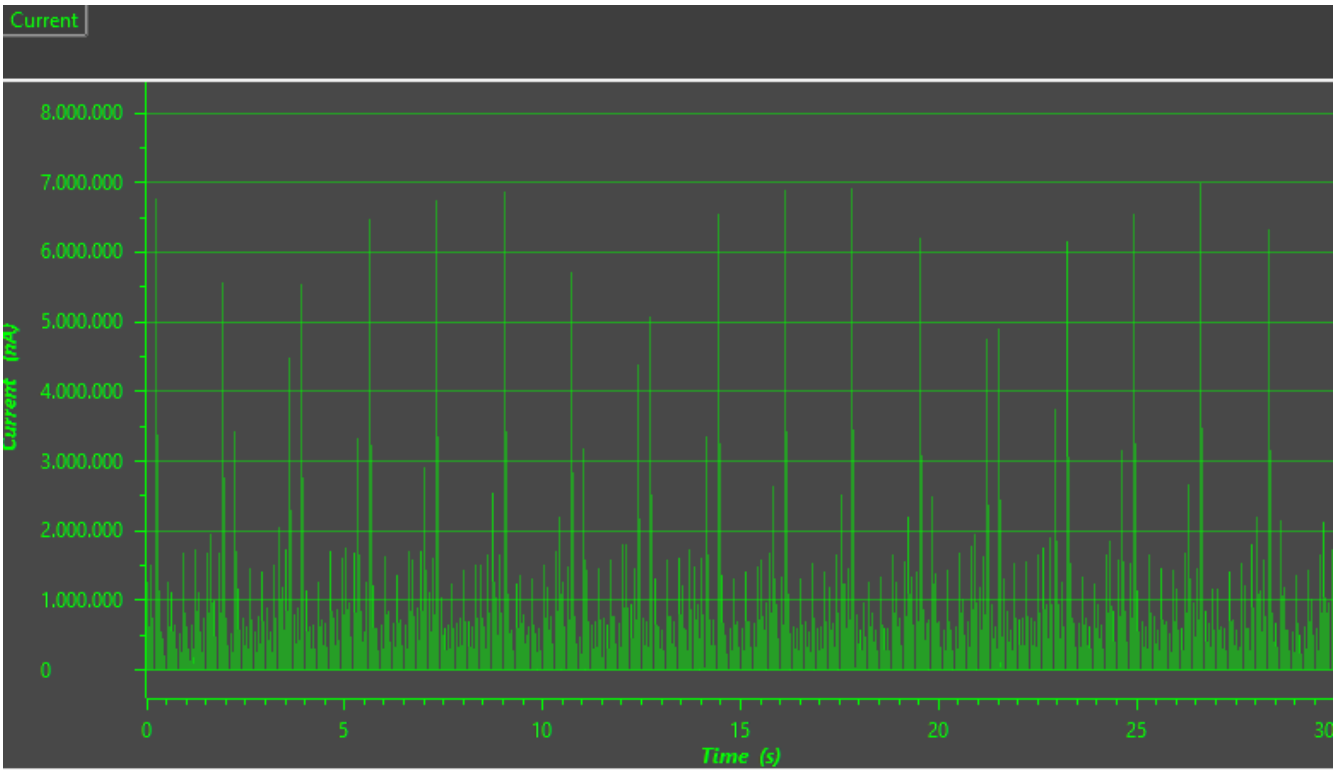

Figure 3-18. Analog Light Sensor: 30 Seconds

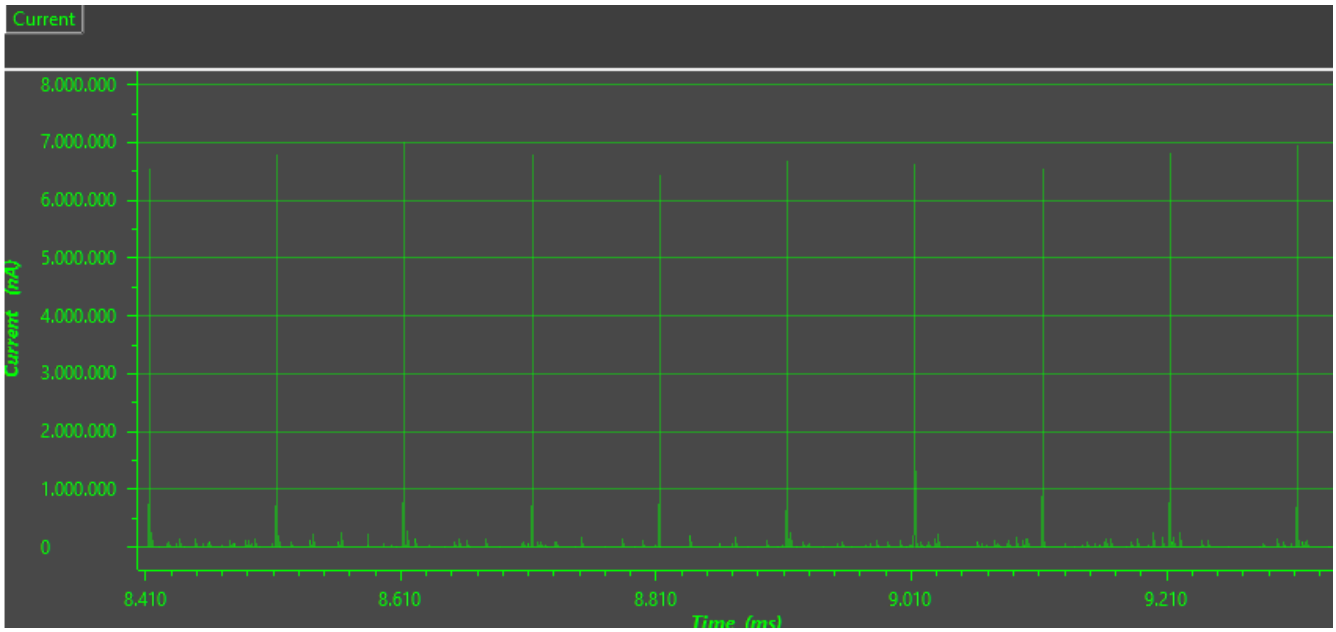

Figure 3-19. Analog Light Sensor: 2 Seconds

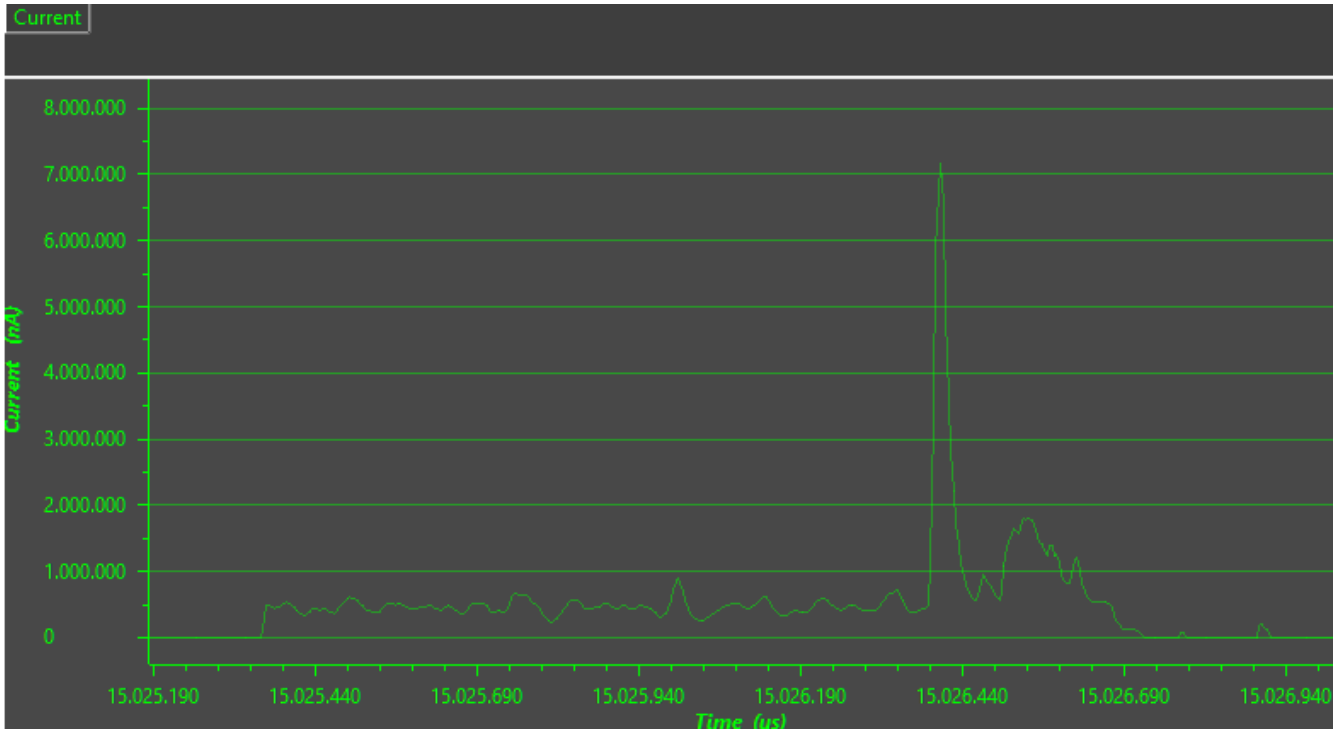

Figure 3-20. Analog Light Sensor: One Measurement

3.5.3 Potentiometer (0 to 200kΩ range)

A potentiometer consists of a resistive element and a movable wiper which adjusts the resistance moving along the element offering a variable voltage output. This makes potentiometers designed for dimmer switches for light control and able to regulate fan speeds, airflow and temperature settings by adjusting resistance in the control circuit to state some examples.

The potentiometer on the BOOSTXL-ULPSENSE is connected as a voltage divider. The example uses COMPA and the reference DAC to implement a SAR-ADC. The potentiometer example is tested with the potentiometer set to the middle position. The Sensor controller can wake up with a rate of 25Hz, and if a change is detected, the Sensor controller can wake up and notify the System CPU.

Table 3-5. Potentiometer Power Consumption

	Average current consumption	Unit	Battery life (CR123)
Potentiometer stationary (25Hz)	2.1	µA	10 years
Potentiometer moving	14.7	µA	1year and 6 months

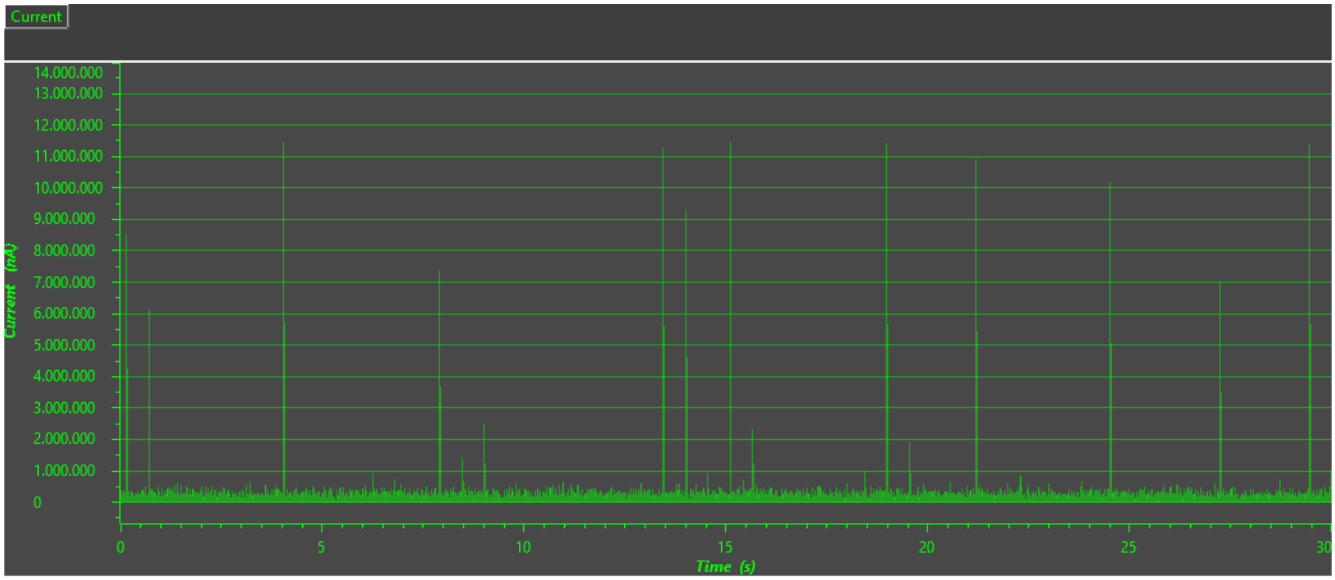

Figure 3-21. Potentiometer: Middle -30 seconds

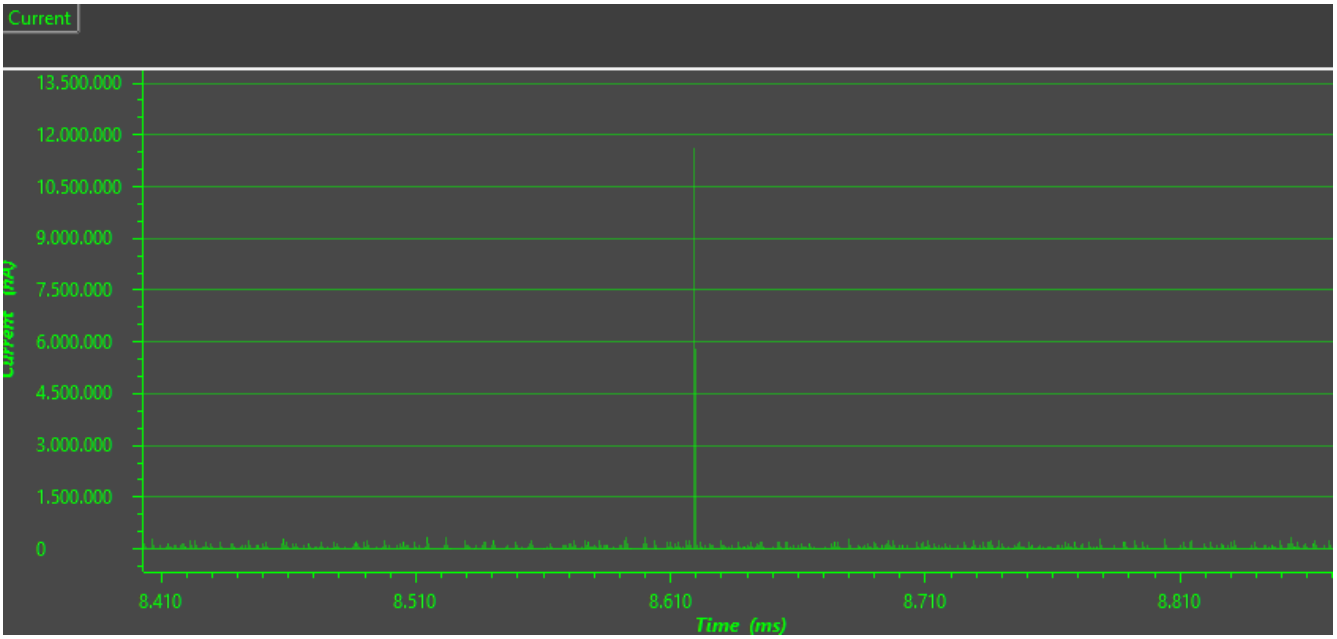

Figure 3-22. Potentiometer: Middle -1 seconds

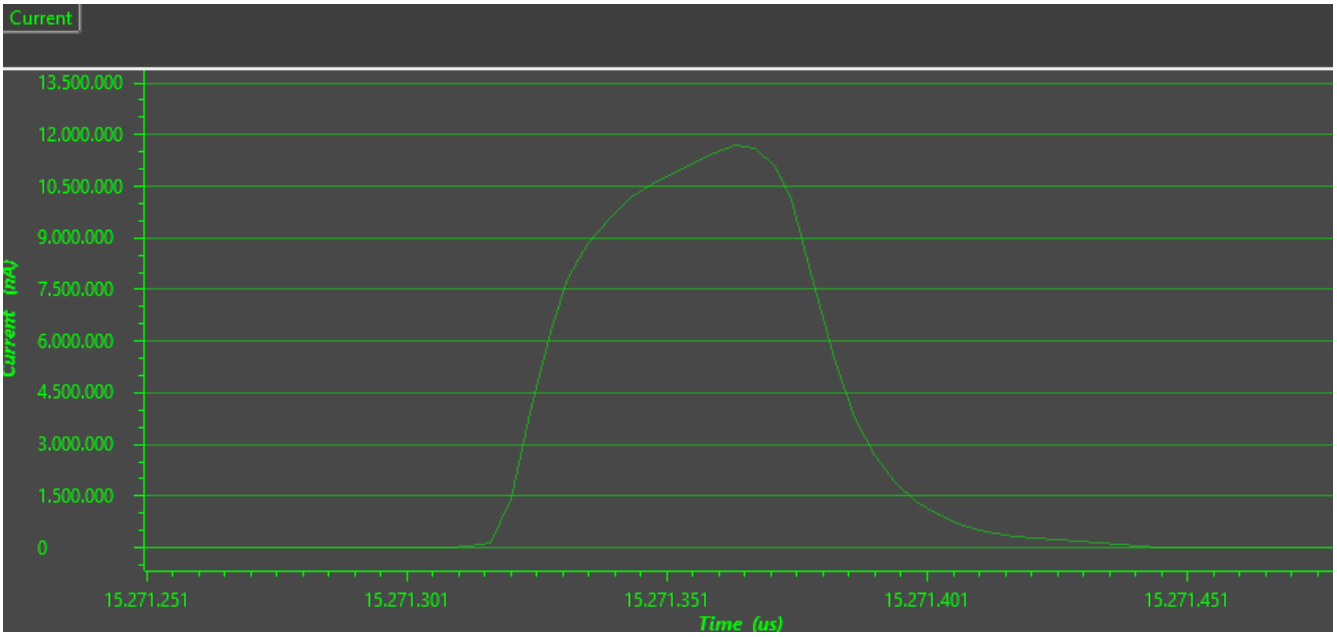

Figure 3-23. Potentiometer: Middle -One Measurement

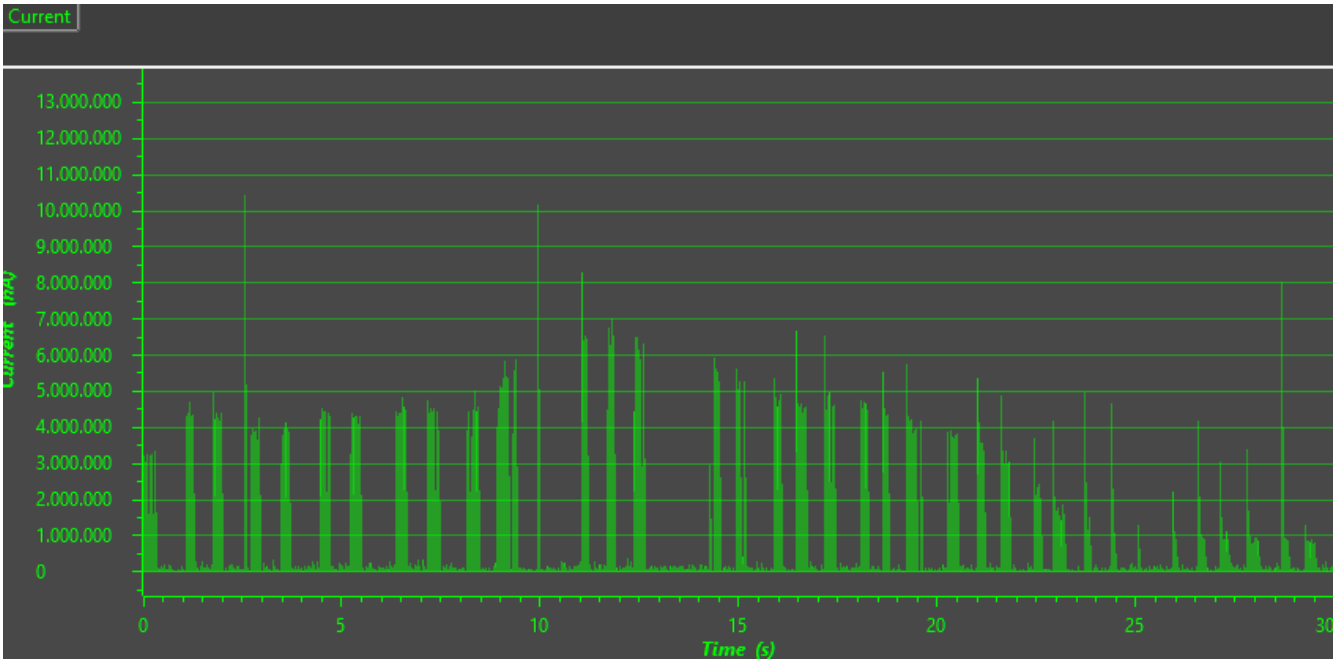

Figure 3-24. Potentiometer: Moving -30 seconds

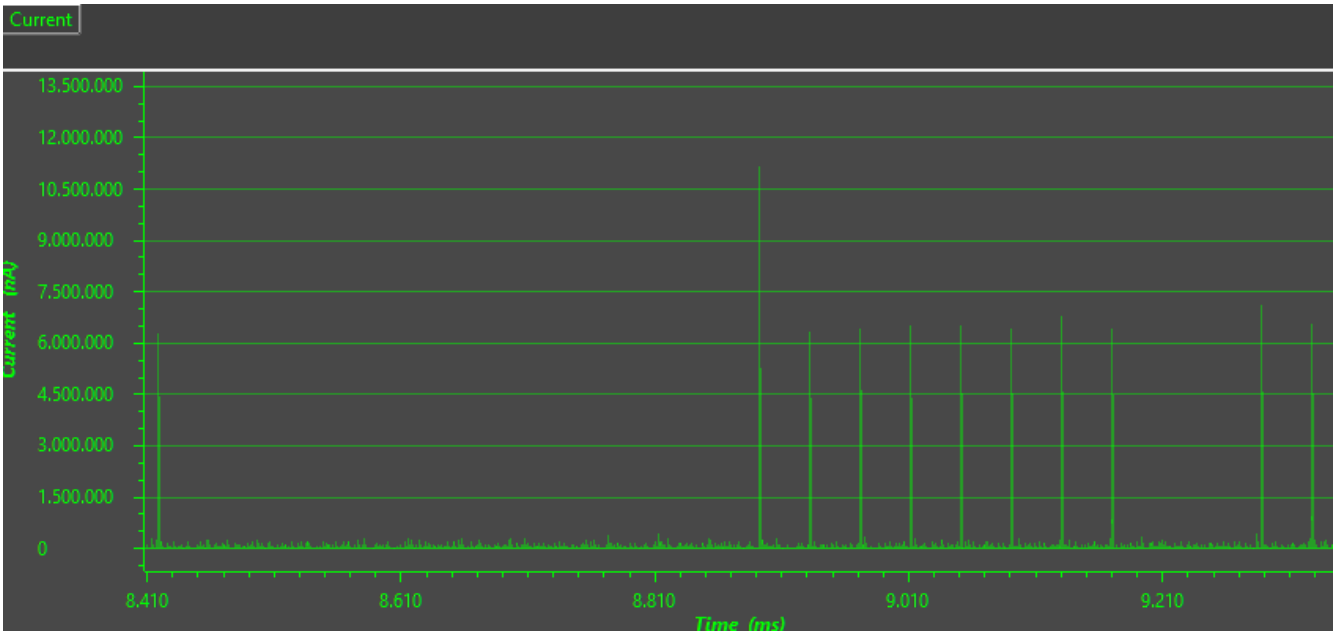

Figure 3-25. Potentiometer: Moving -1 second

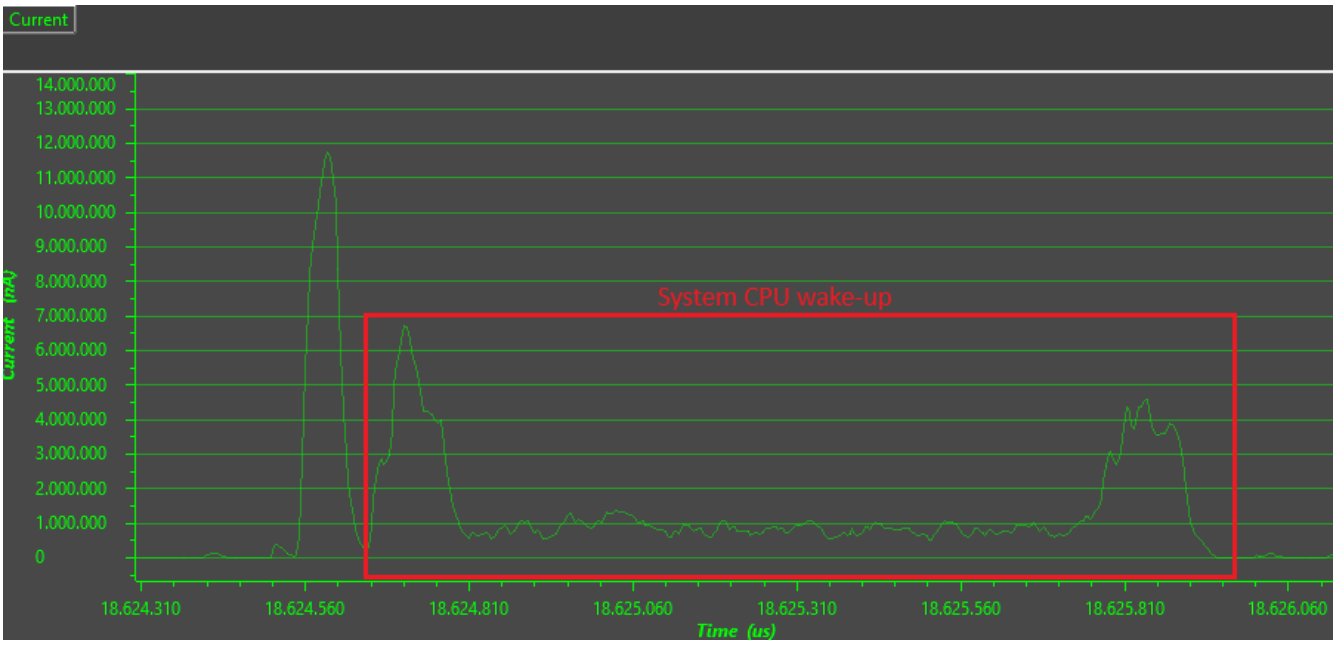

Figure 3-26. Potentiometer: Moving -One Measurement

3.5.4 Ultra-Low Power SPI Accelerometer

Ultra-low-power accelerometers are able to detect motion, orientation, vibration and even gestures with minimal energy consumption. These sensors enable a range of applications that improve efficiency, safety and convenience in modern smart buildings.

In **occupancy detection**, accelerometers can be used to sense motion or the presence of people in a room as accelerometers can be highly sensitive and are able to detect subtle vibrations such as footsteps. This becomes relevant for applications like conserving energy in lighting and HVAC systems and also for intrusion detection.

Systems where **monitoring and diagnostics** are crucial also make use of accelerometers. In elevator and HVAC systems, accelerometers provide data about vibration and movement patterns and thus can detect irregularities in operation and signal maintenance when needed.

As a side note, we also have a machine learning design for accelerometer data processing, leveraging the capabilities of the CC1352 platform to recognize gestures or movement patterns (for example up-down, side-to-side or circular motion). The full design can be easily evaluated with Edge Impulse platform, for additional information you can get started from here.

The BOOSTXL-UPLSENSE comes with a ultra-low power accelerometer. This sensor uses the serial peripheral interface (SPI) interface to communicate with the Sensor Controller. In the SPI accelerometer example, the device is kept at a steady state once . The accelerometer can report to the Sensor Controller with a frequency of 100Hz. If a change in the accelerometer over a certain threshold is detected, the Sensor Controller can wake up the system CPU and the System CPU can light up one of two LEDs.

Table 3-6. SPI Accelerometer Power Consumption

	Average current consumption	Unit	Battery life (CR123)
SPI Accelerometer stationary (100Hz)	5.1	µA	4 years and 5 months
SPI Accelerometer moving (100Hz)	8.5	µA	2 years and 8 months

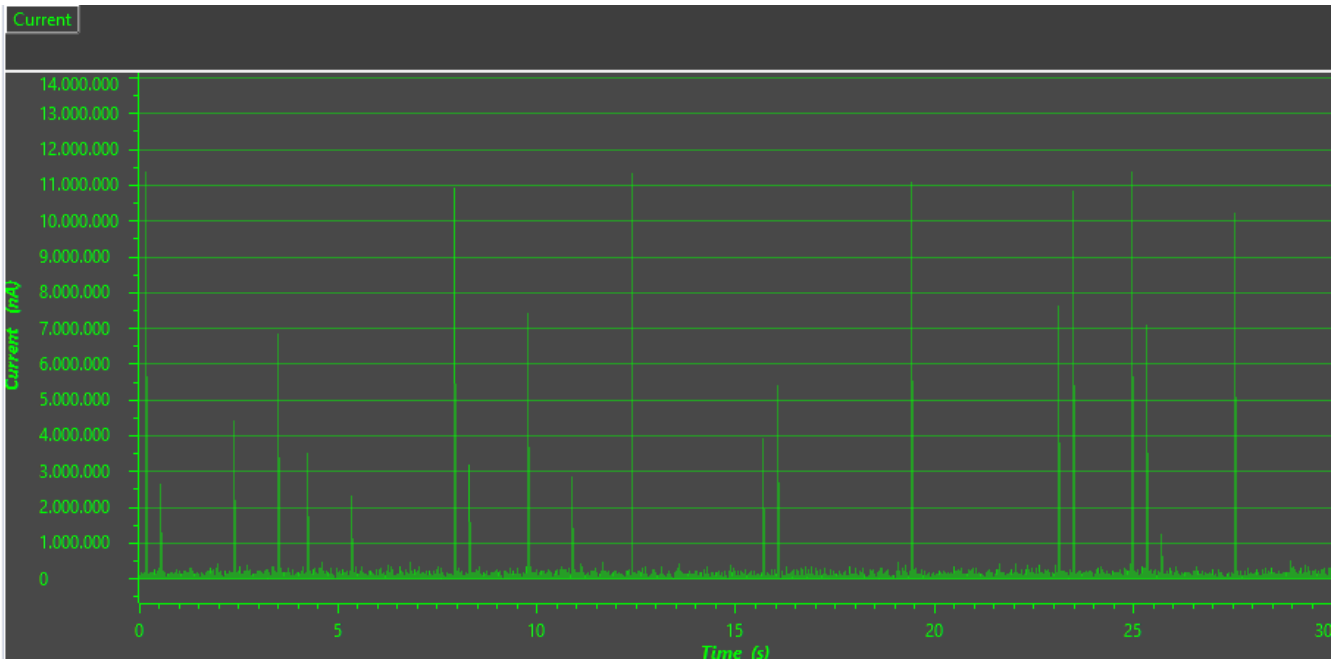

Figure 3-27. SPI Accelerometer: Steady - 30 seconds

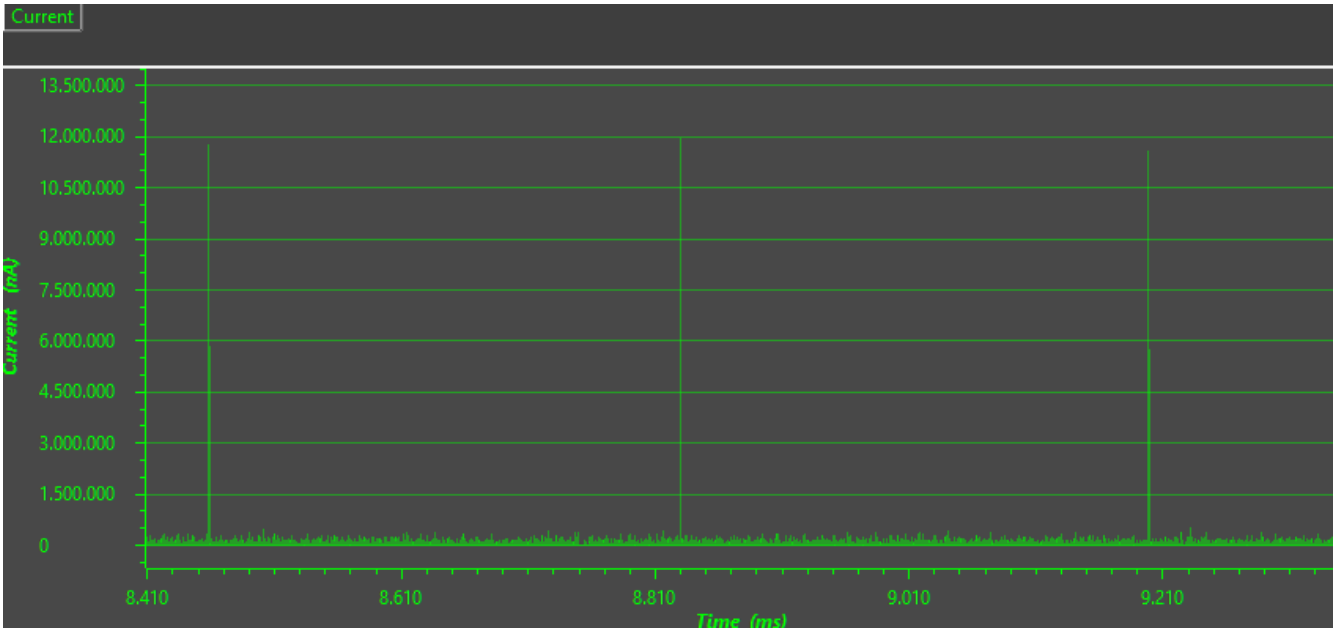

Figure 3-28. SPI Accelerometer: Steady - 1 second

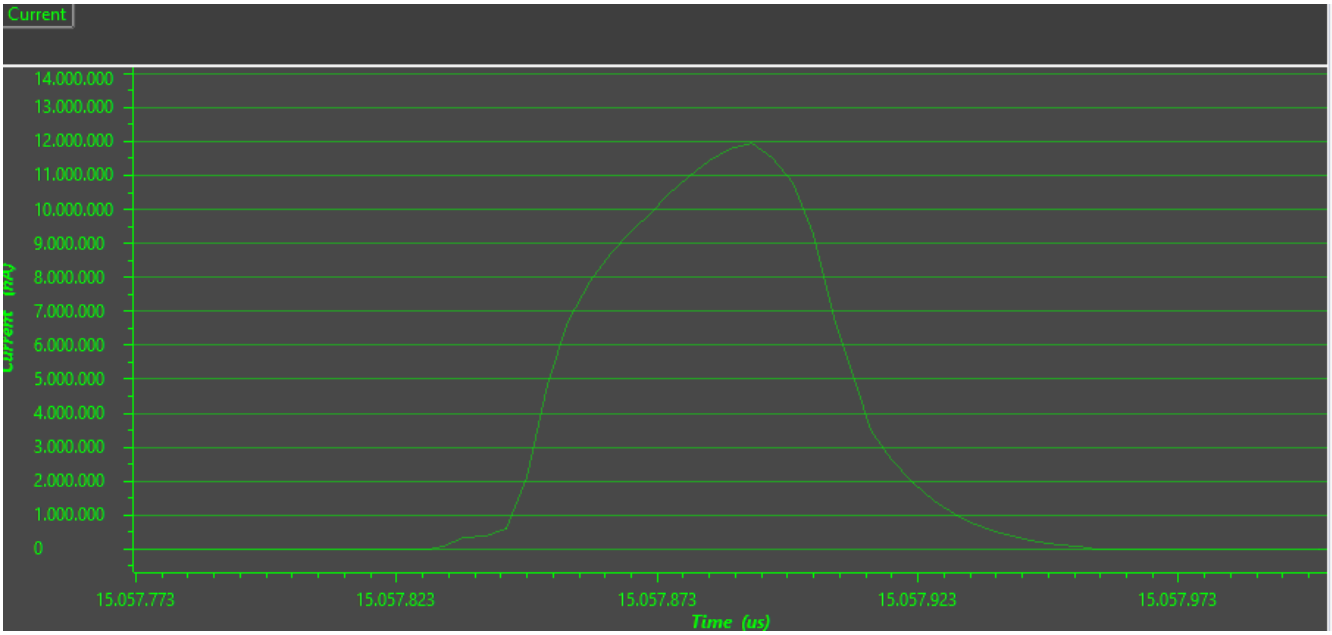

Figure 3-29. SPI Accelerometer: Steady - One Measurement

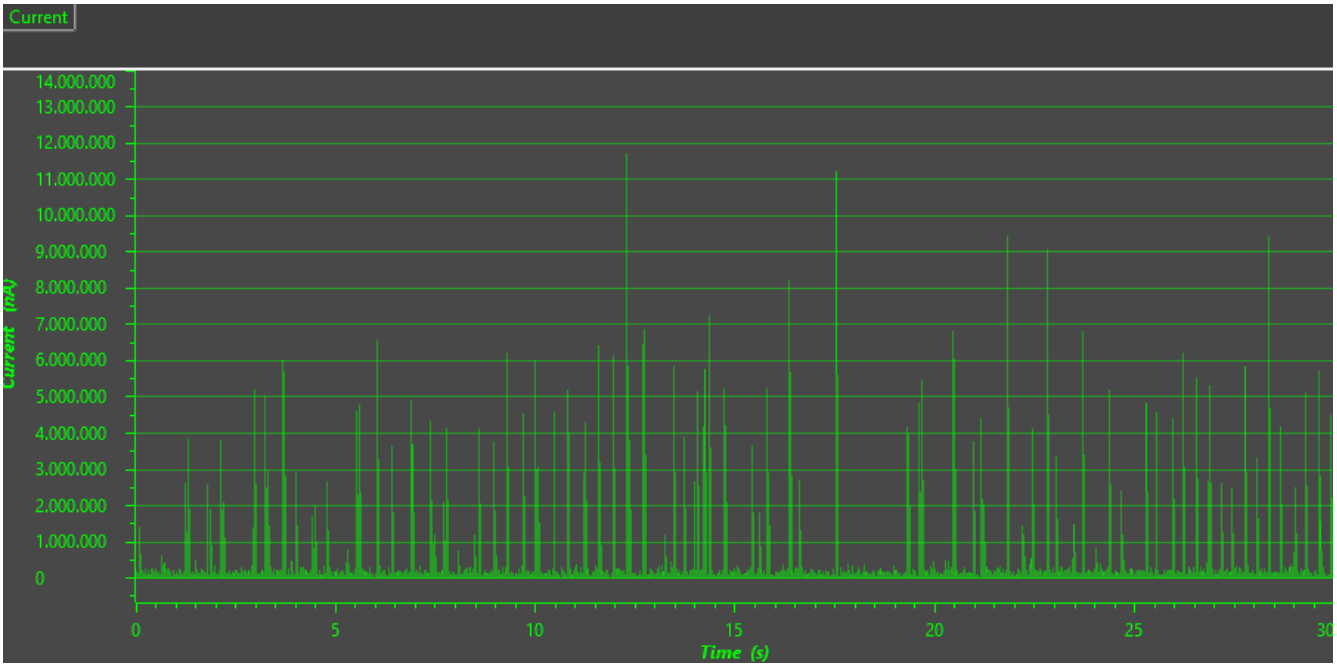

Figure 3-30. SPI Accelerometer: Moving - One Measurement

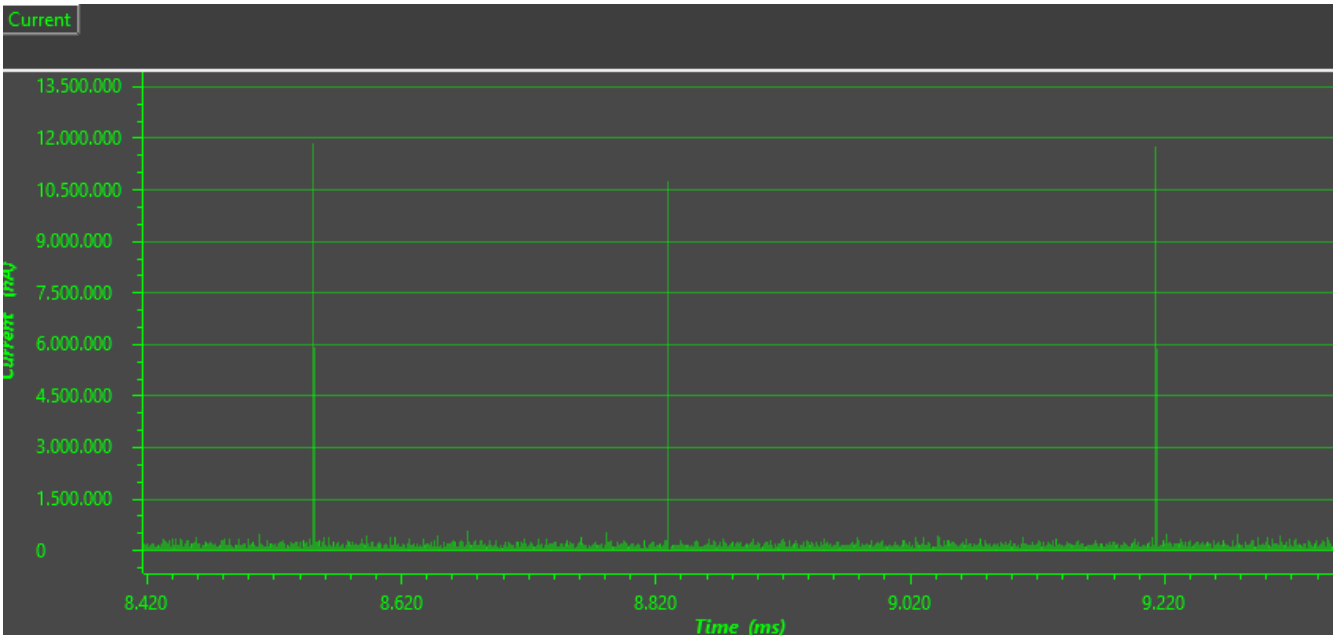

Figure 3-31. SPI Accelerometer: Moving - One Measurement

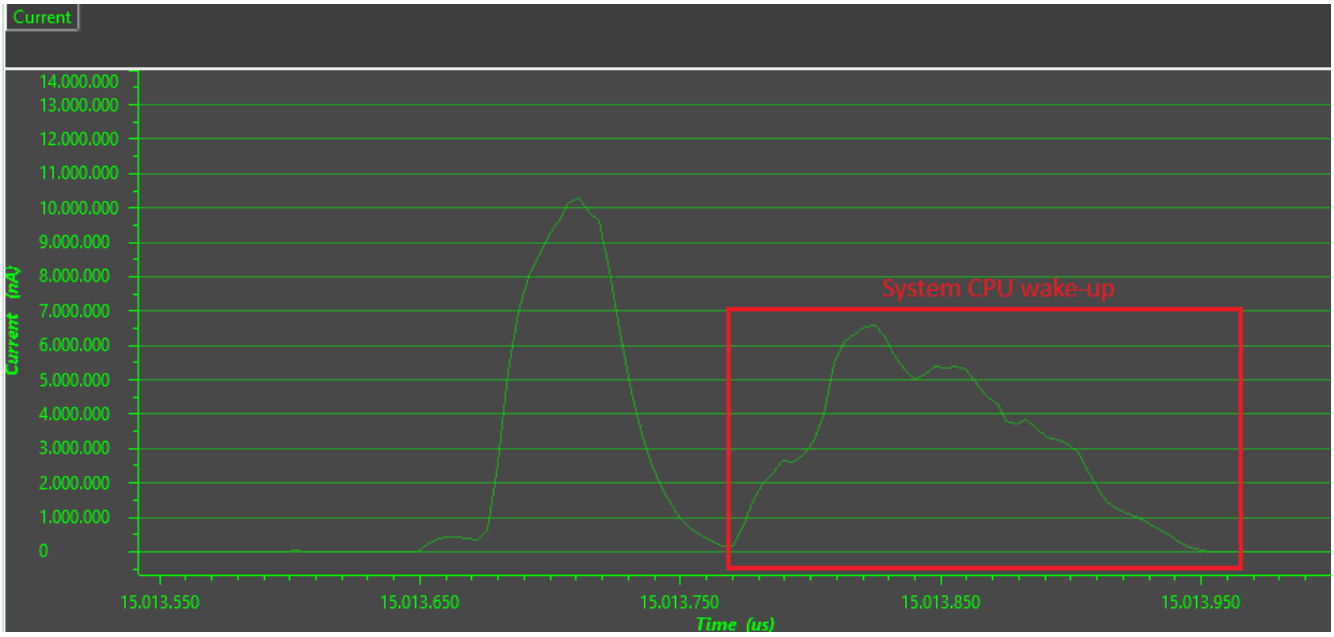

Figure 3-32. SPI Accelerometer: Moving - One Measurement

3.5.5 Reed Switch

The reed switch measurements are related to the Door & Window sensor that was covered in Section 3.3.

Table 3-7. Reed Switch Power

	Average current consumption	Unit	Battery life (CR123)
Without magnet present (64Hz)	1.7	µA	13 years
With magnet present (64Hz)	8.5	µA	2 years and 7 months

4 Summary

Table 4-1. Power Consumption and Battery Life Time

	Average current consumption (Sensor Controller)	Average current consumption (Cortex M4F)	Estimated Sensor Controller Battery Life Time (CR123)	Estimated ARM Cortex Battery Life Time (CR123)
PIR Motion Detection(20Hz)	7uA	approximately 2mA	5 years and 1 month	3 months
Glass Break Detection(Piezo sensor)	4uA	approximately1mA	6 years	5 months
Door/ Window -REED switch reading	8.5uA	approximately 1.5mA	5 years	3 months
SPIreading (4MHz in active mode)	3.7uA	243.8uA	6 years and 1 month	2 month and 3 days
SPIreading (1MHz in low power mode)	2.1uA	266.7uA	10 years and 9 months	2 month and 1 day
100Hz pin toggle (from Standby to active (24MHz) and back to Standby)	14.5uA	approximately 135uA	N/A - Use case is only for benchmarking power differences between modes	N/A- Use case is only for benchmarking power differences between modes
100Hzpin toggle (from Standby to low power (2MHz) and back to Standby)	0.8uA	approximately 135uA	N/A - Use case is only for benchmarking power differences between modes	N/A- Use case is only for benchmarking power differences between modes

When comparing the Sensor Controller to the System CPU, the Sensor Controller has significantly lower current consumption.

The different examples shown in this application note use many different techniques to reduce the power consumption. Some ways that low power consumption can be achieved are shown here:

- Low-power mode instead of active mode where possible
- Timer 2 in low-power mode instead of TDC, where possible
- SPI instead of I2C, where possible
- Timer 2 at 32kHz to power up the sensor and then wake up the Sensor Controller (when the sensor is ready)
- Timer 2 at 2MHz/32kHz as pulse width modulator (PWM) for status LEDs
- Pre-process the sensor data to detect relevant activity, and wake up the System CPU application only when needed.
- General-purpose input/output (GPIO) event handler for interrupt from digital sensor, with timeout for sensors that generate interrupt periodically.
- Minimized communication over I2C/SPI, reading multiple external device registers in one operation.
- Disable peripherals as soon as possible after measurement (before the data processing).
- Reduced sensor polling frequency when the sensor data indicate no activity.
- Reference DAC and COMPA in low-power mode can be used as low-precision ADC.

5 References

- Texas Instruments, *Sensor Sequencing Using the CC13x2 and CC26x2 Sensor Controller*, application note.
- Texas Instruments, *Low-Power ADC Solution for CC13x2 and CC26x2*, application note.
- Texas Instruments, *PIR Motion Detection With MSPM0*, application note.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265