Designing Sensors into Battery-Powered IoT Nodes

The Internet of Things (IoT) is turning real-world “analog” events into networked actions and reactions. Connected IoT nodes monitor analog events and, when events occur that need to be reported, translate them over the Internet to the cloud for an application to do something with it.

One of the most prominent classes of IoT applications uses battery-powered sensors, placed in a zone with no electrical wires to monitor events and communicate wirelessly to the IoT network. In most cases, these products are always-on, battery-operated, wireless sensors that support a wireless protocol, an MCU, and at least one analog sensor.

The challenge is to maximize the time the product can adequately sense the environment on a single battery or charge.

This challenge breaks down as follows:

- Adequately sensing the environment as required by the application;
- Completing any required sensor measurements using as little energy as possible;
- Keeping the “periodically required” MCU peripherals and CPU core asleep as much as possible.

Many typical MCUs in this type of application wake up the MCU core and various peripherals to do sensor measurements (Figure 1). When there is an event to report, such as a door opening, the MCU reports it and then returns to its duty-cycle process. This takes a lot of energy and does not maximize battery life because the “whole MCU” is operating including many peripherals and unneeded core processing power.

In fact, this approach will most likely result in a poor customer experience: the customer puts the device into their environment, sets it up on the network, and enables it, only to have it die a few months later because it does a poor job of managing its battery power.

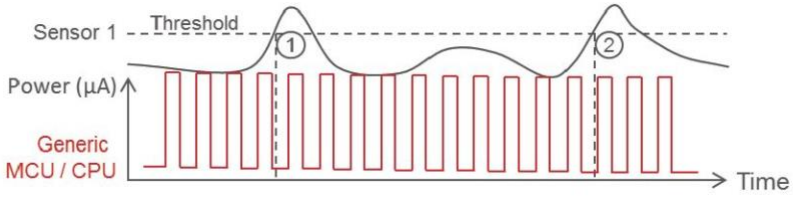

Figure 1: High energy consumption with CPU polling and active during every measurement.

The Ideal Battery-Powered, Wireless Sensor Node Solution for the IoT

The ideal solution will address every point in the challenge statement above... it will **maximize the time** the product can **adequately sense** the environment on a **single battery charge**.

With this in mind, a battery-powered IoT sensor device would offer:

- Autonomous, energy-efficient systems for sensor management and measurement;
- Individually configurable sensor inputs/outputs, thresholds, and configurations for each sensor;
- A low-power, configurable logic engine that wakes up the MCU only when it's absolutely required;
- Low-power memory to buffer multiple measurments and lengthen times between CPU wake-ups;
- Low-power wireless.

Silicon Labs Gecko Low Energy Sensor Interface (LESENSE)

Years ago, Silicon Labs anticipated the importance of battery-operated wireless sensor applications. We have been investing aggressively in energy-efficient wireless, MCU, and sensor technology ever since.

Our Gecko MCUs are architected with energy-efficiency in mind and offer several key systems that allow them to operate more efficiently and longer than other MCUs.

The Gecko and Wireless Gecko (together, “Gecko MCUs”) use Low Energy Sensor Interface (LESENSE) and the Peripheral Reflex System (PRS) as well as other energy-efficient technology to operate at very low power levels, while leaving the core and majority of the MCU in Deep Sleep modes.

This paper provides a high-level overview of LESENSE. The Peripheral Reflex System (PRS) allows various peripherals to be duty-cycled without core intervention. PRS is similarly important for power-saving advantages, but is addressed in other resources referenced at the end. Covering low-power wireless is also outside the scope of this paper.

Each requirement above combines with the others to conserve the most energy.

Solution Requirement	Requirement Explanation
1. Autonomous, energy-efficient sensor systems	By using an autonomous sensor system, the power-hungry core and other unnecessary peripherals can remain in Deep Sleep mode.
2. Individually configurable sensor inputs/outputs, thresholds, and configurations for each sensor	Because the inputs/outputs are each individually configurable for their assigned sensor, even various aspects of the sensor system itself remain asleep while others take measurements.
3. Low-power, configurable logic engine that wakes up the MCU only when it's absolutely required.	With low-power, dedicated logic, almost endless variations in sensor thresholds and events can be handled without waking the core until it's required.
4. Low-power memory to lengthen times between CPU wake-ups	Dedicated storage complements the low-power logic and allows multiple events to occur without waking the core or the rest of the chip. The same memory can improve sensor recalibration when necessary.
5. Low-power wireless	Wireless Geckos offer some of the most energy-efficient Bluetooth® Low Energy (BLE), ZigBee®, Thread, and proprietary wireless in the market.

Requirements for Battery-Powered IoT Sensor Systems

Gecko LESENSE Details

LESENSE is a highly configurable sensor interface and system that autonomously and continuously manages and monitors up to 16 resistive, capacitive, or inductive sensors while the overall chip remains in Deep Sleep mode and the core (CPU) remains off.

LESENSE consists of a sequencer, a count-and-compare block, a configurable decoder, and a RAM block for configuration settings and measurement results storage.

1. The **sequencer** operates the low-frequency oscillator and handles interaction with other peripherals through PRS, as well as timing the sensor duty-cycles and measurements.
2. The **count-and-compare block** counts pulses from the sequencer and compares the information with configurable thresholds for high-order measurements.
3. The **decoder/state machine** receives the sensor measurements and takes action based on up to 16 configurable states and associated actions.

LESENSE Configurable Sensor Thresholds

Waking up the CPU when an external event passes a sensor threshold is not a revolutionary concept. In essence, it removes the constant MCU duty-cycle from Figure 1 to a single event; when an analog event crosses a given threshold, the MCU wakes up and performs various actions.

However, what is unique in LESENSE is a complete sensor system managing and monitoring sensors and associated peripherals with no CPU involvement and minimal MCU involvement. This is the basic LESENSE idea, but the additional features take the idea further.

LESENSE also buffers a configurable number of threshold events without waking the CPU. This allows the system to monitor external stimuli over a longer period of time. LESENSE does this by autonomously duty-cycling any required peripheral blocks, such as analog comparators, low-frequency oscillators, and the sensor itself to complete sensor measurements while the CPU remains in Deep Sleep mode.

In the conceptual figure below, LESENSE is configured to allow Sensor 1 to exceed its configurable threshold twice before waking up the CPU.

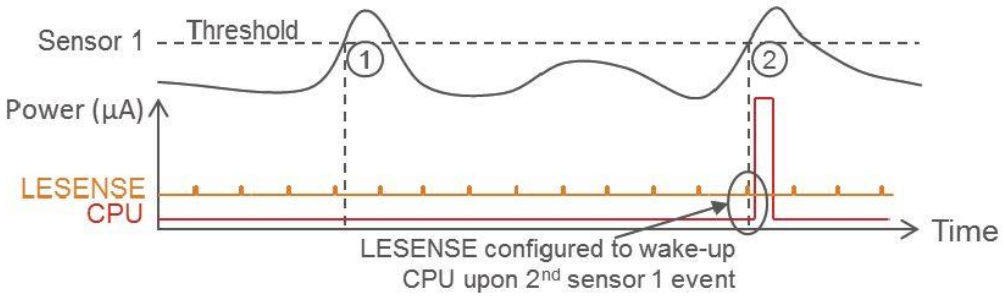

Figure 2: Each LESENSE-enabled sensor input/output is independent and configurable.

LESENSE also provides the added ability to manage and monitor up to 16 different sensors with unique thresholds. With the built-in, low-power state machine (decoder), LESENSE can evaluate several events before issuing an interrupt to wake up the CPU.

In Figure 3, LESENSE buffers measurement information for Sensor 2's events 1, 2, and 3 and combines them with measurement data for Sensor 1's events 1 and 2 before waking up the core. This simple use case employs LESENSE's individually configured sensors, low-power memory, and the low-power state machine.

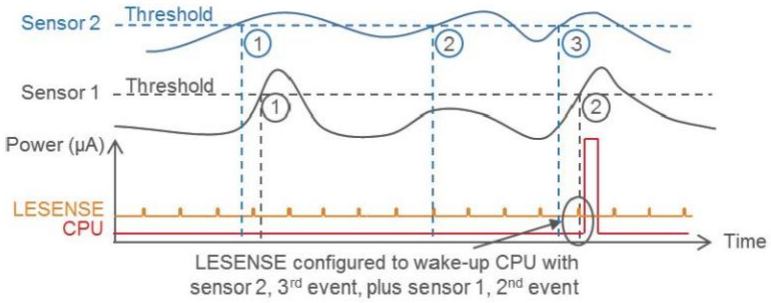

Figure 3: Multiple sensors and unique configurations support multiple events prior to CPU interrupts.

Sensor Node Recalibration from LESENSE Buffered Measurements

Since many sensor systems are implemented in a wide variety of environmental conditions, they must be able to operate reliably while parameters such as temperature, humidity, supply voltage, permeability, and conductivity change constantly.

With LESENSE's buffering capability, the CPU can recalibrate itself to multiple readings when it awakens. This avoids multiple, repeated calibration events as conditions change, further saving energy and providing a larger sample set for system calibration.

Summary

LESENSE enables the Gecko MCU and Wireless MCUs to monitor resistive, capacitive, inductive, (and IR) sensors while keeping the power-hungry core and most of the MCU in Deep Sleep mode. LESENSE can monitor up to 16 sensors using less than a µA, and offers configurable thresholds, available RAM for buffering multiple events, and a state-machine for configurable wake-up interrupts.