

# Designing Low Power Systems with Digital MEMS Sensors

Petr Stukjunger  
STMicroelectronics, Pobrezni 620/3, Prague, Czech Republic,  
petr.stukjunger@st.com

## Summary:

Design of low power systems has been significantly improved by recent innovations in the field of digital MEMS sensors. System engineers can take advantage of sensors' embedded features and parameters that enhance and simplify power consumption optimization at both sensor and system level.

**Keywords:** low power, system design, MEMS sensor, application, machine learning

## Introduction

More and more features and functionalities are expected from modern embedded systems. At the same time, it is required to minimize power consumption. These requirements are contradictory and often difficult to fulfill.

Applications that are utilizing digital MEMS sensors to measure environmental, motion and other types of data, can take great benefits by utilizing sophisticated features of these sensors.

In this paper we will discuss the features of digital MEMS sensors and system level techniques that designers may exploit to minimize power consumption of their systems.

## Low power sensor and features

Wide popularity of battery-operated nodes motivated manufactures of MEMS sensors to develop devices with ultra-low current consumption. The newest accelerometers can measure movements while consuming less than one microamp. Even larger improvements have been made in the design of gyroscopes, where we have seen more than 10times drop in their current consumption just over last couple of years.

Digital MEMS sensors are very flexible in their configuration offering variety of operating modes with associated output resolution and wide range of applicable data rates. Designers can therefore select the most suitable configuration for each application case. Some sensors are even capable to switch their operating mode and data rate autonomously based on an external motion event. Even though these improvements are of substantial help in low power system design, the newest MEMS sensors bring even more features that can help.

Motion detection features like wake-up (for system activation based on a motion), free fall (detection of device falling down), orientation detection (used daily in our mobile phones), single & double tap (for enhanced user interface) became almost industry standard. More and more common are also features like step counter or pedometer, tilt detection etc.

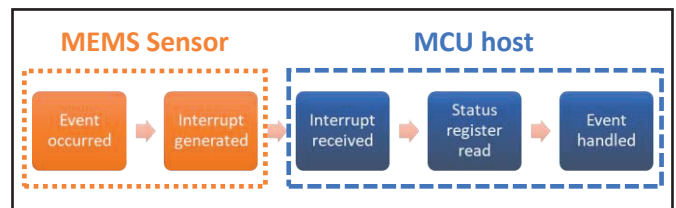


Fig. 1. Utilizing interrupt of digital MEMS sensor.

The above-mentioned features allow to offload microcontroller from continuous acquisition and evaluation of sensor data bringing a substantial decrease in system power consumption. It is simply achieved by the utilization of interrupt signals routed from the sensor to the microcontroller. The microcontroller is not anymore involved in nonstop data acquisition, instead it is solely waiting for a signal from the sensor raised just in the moment when there was a new event to be handled by the system.

## Advanced Embedded Features

The latest six axis inertial measurement units (IMUs) from STMicroelectronics bring system power reduction to a next level as discussed in [1]. Thanks to Finite State Machine and Machine Learning Core blocks it is possible to move medium complex algorithms from microcontroller inside the sensor itself and consequently reduce not just microcontroller's current consumption and load, but also traffic on communication bus.

### Finite State Machine

Finite State Machine (FSM) block provides to system designers possibility of creating their own state programs, where in each state two conditions may be evaluated, or a command executed. The conditions evaluate sensor output data with respect to a user-defined threshold or time-related execution based on internal timers. The FSM is well suited for implementation of gesture recognition algorithms or enhanced control mechanisms based on the sensor data.

### Machine Learning Core with AI

Machine Learning Core (MLC) is a hardware implementation of decision trees inside digital MEMS sensor. The sensor can run motion classification algorithms by its own and take the advantage of machine learning techniques well known from the field of artificial intelligence.

The utilization of the MLC follows common machine learning process. It starts by data collection and labeling followed by identification and extraction of the features that characterize the movements to be recognized. Then any conventional machine learning tool is executed to generate decision tree. Finally, the decision tree is converted into set of values, which are then loaded into sensor's configuration registers to run the decision tree algorithm.

The overall system current consumption is dramatically reduced as can be seen from the following example. We took human activity recognition algorithm and compared its implementation as microcontroller library vs. decision tree inside the MLC. In this comparison we have used STM32L476, ultra-low-power 32bit Cortex-M4 MCU, and LSM6DSOX, high-performance 6axis IMU with embedded MLC core, from STMicroelectronics.

As shown in table Tab 1. in both cases the sensor needs 15  $\mu\text{A}$  to sample data. Running the algorithm inside the MLC increases sensor's current consumption just by a few extra micro amps. When the microcontroller is running activity recognition library, it needs to collect sensor data at certain rate (16Hz in this case) and run the classification algorithm. On the other hand, when the MLC is running the activity recognition, the microcontroller can be left in a very low power mode for most of the time and wakes-up only upon notification from the sensor that a new motion class has been detected. In our example we considered that the class (user activity in this case) will change on average every 30 seconds. We can see that the overall system current consumption is greatly reduced by offloading the microcontroller and prolonging its stand-by time.

Tab. 1: Current consumption of an embedded system when running activity recognition algorithm. Comparing implementation inside STM32L476 32bit MCU vs. MLC of LSM6DSOX 6axis IMU.

	MCU library implementation	MLC implementation
Sensor data chain	15 $\mu\text{A}$	15 $\mu\text{A}$
Sensor MLC block	0 $\mu\text{A}$	4 $\mu\text{A}$
MCU stand-by time	63 ms (1/16Hz)	30 s
MCU consumption	51 $\mu\text{A}$	0.7uA
<b>Total</b>	<b>66 <math>\mu\text{A}</math></b>	<b>19.7 <math>\mu\text{A}</math></b>

### From Low Power Sensor to Low Power System

We have seen that sensors offer many options to optimize overall system current consumption. There are several methods how to save current also at the system level as discussed in [2]. The output data shall be read from the sensor using so-called data ready interrupt instead of continuously polling a status register to check whether new data has been sampled. Communication on the serial bus between the sensor and the microcontroller is another contributor to the overall system power consumption. SPI interface is therefore preferred over I<sup>2</sup>C. Brand new sensors are equipped with MIPI I3C<sup>SM</sup> bus that combines benefits of SPI, i.e. speed, and I<sup>2</sup>C, i.e. number of wires. Power supply level shall be as low as possible because it decreases the current consumed by the sensor itself.

### Conclusion

The ultra-low current consumption of modern MEMS sensors and their embedded features like FSM and MLC are handy tools for designers of low power systems. By employing optimizations also at system level, the overall power consumption budget is reduced multiple times. These mechanisms open new approaches to design of embedded systems including edge computing, in case of the MLC brought really to the very source of the sensor data.

### References

- [1] M. Castellano, R. Bassoli, M. Bianco, A. Cagidaco, C. Crippa, M. Ferraina, M. Leo, S.P. Rivolta, Embedded Algorithms for Motion Detection and Processing, proceedings of embedded world 2018 exhibition&conference
- [2] Vladimir Janousek, Petr Stukjunger, Low-power application design with ST's MEMS accelerometers, design tip, Rev 1, February 2019