Application Report
SWRA578A – October 2017 – Revised May 2019

Getting Started With the CC13xx and CC26xx Sensor Controller

Mandus Börjesson

ABSTRACT

This application report is intended to be an introduction to the low-power Sensor Controller Engine in CC13xx and CC26xx devices. It explains the benefits of using the Sensor Controller in various use cases. The document also describes how to get started using the Sensor Controller and directs users to the required tools and collaterals.

Contents

List of Figures

List of Tables

Trademarks

LaunchPad, SimpleLink, Code Composer Studio, MSP432 are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

1 Introduction

This application report explains what the Sensor Controller (SC) is, how to get started using it, and how users can measure the current consumption of an application running on it. The focus of the application report is a comparison between the CC13x2/CC26x2 Sensor Controller and the Sensor Controller in the CC13x0 and CC26x0 MCUs. To highlight the advantages of using the Sensor Controller instead of using the main CPU, a comparison of the current consumption, where the main CPU performs the same tasks as the SC, is also included.

The measurements are performed on a CC1310 LaunchPad™, representing all CC13x0 and CC26x0 devices, and a CC1312 LaunchPad™ representing all CC1312, CC1352, CC1352P, and CC26x2 devices. In addition, the SimpleLink™ MSP432P401R LaunchPad is used for comparison.

Figure 1 shows the building blocks of the CC26x0 highlighting the Sensor Controller.

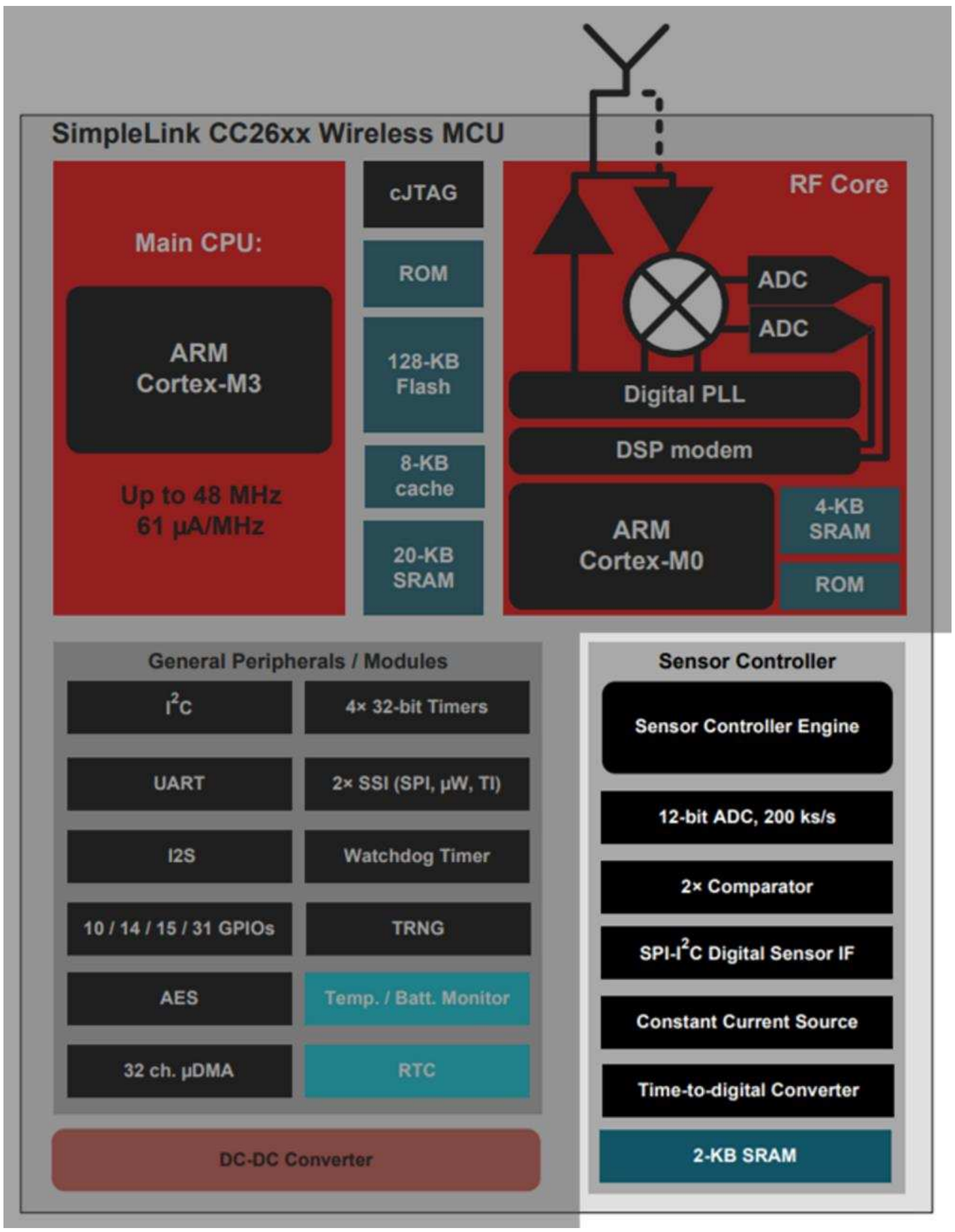

Figure 1. Block Diagram Highlighting CC26xx Sensor Controller

Table 1 lists the acronyms used.

Table 1. Abbreviations

Acronym	Definition
ADC	Analog-to-digital converter
CCS	Code Composer Studio™
CPU	Central processing unit
DMA	Direct memory access
MCU	Microcontroller unit
RTC	Real-time clock
RTOS	Real-time operating system
SC	Sensor Controller
SCE	Sensor Controller Engine
SCIF	Sensor Controller Interface Driver
SCS	Sensor Controller Studio
SPI	Serial peripheral interface

1.1 Background

The Sensor Controller is a small CPU core that is highly optimized for low power consumption and efficient peripheral operation. The Sensor Controller is in the CC26xx/CC13xx auxiliary (AUX) power/clock domain, and can perform simple background tasks autonomously and independent of the System CPU and MCU domain power state. The CC13x2/ CC26x2 Sensor Controller boasts some additional features over its predecessor, the CC13x0/CC26x0 Sensor Controller. Those features include: additional timers, access to more I/O pins, and more low-power modes. Table 2 compares the SC features between the CC13x0/CC26x0 and the CC13x2/CC26x2 devices.

Table 2. CC13x0/CC26x0 SC and CC26x2/CC13x2 SC Feature Comparison

CC13x0 and CC26x0 SC	CC26x2 and CC13x2 SC
2kB memory	4kB memory
32-kHz, 24-MHz system clock	32-kHz, 2-MHz, 24-MHz system clock
ADC	ADC
2x comparators	2x comparators
Configurable current source	Configurable current source
Time-to-digital converter	Time-to-digital converter
1x timer	3x timers with additional capabilities (match, compare, PWM)
Bit-banged SPI data transfer	Hardware SPI data transfer
Control of up to 8 analog/digital and 8 digital IOs	Control of up to 8 analog/digital and 22 digital IOs (all DIOs)
	Reference DAC
	Dynamic power control (allows switching between active and low-power mode during code execution)
	16 × 16-bit hardware multiplier with accumulator
	Signal observation

2 Device Overview

This chapter covers the software, hardware, and recommended reading required to replicate the results in this application report.

2.1 Software

The following software is used to create and work with Sensor Controller projects (revision numbers reflect version used for performance testing in this document. The latest version of the tools should always be used).

- Sensor Controller Studio (SCS) v2.3.0.684
- Code Composer Studio 8.2.0.00007
  - SimpleLink CC26x2 SDK using CoreSDK 3.80
  - SimpleLink SDK for MSP432 1.35

In addition, the software used to collect data on current consumption and logic signals from the DC power analyzer and logic can be found at the following web sites:

- Saleae Logic 1.2.14
- Keysight 14585A Control and Analysis Software

2.2 Hardware

For the measurements, the following three pieces of hardware were used:

- MSP432P401R LaunchPad
- CC1310 LaunchPad (rev. 1.0)
- CC1312 LauchPad (rev)

For measuring the current and logic signals, the Keysight N6705 DC Power Analyzer and Saleae Logic16 were used.

2.3 Recommended Reading

A video introduction to the Sensor Controller can be seen at training.ti.com.

For developping code for the Sensor Controller it's recommended to use Sensor Controller Studio. Documentation for the tool and the resources available in the Sensor Controller could be found either be pressing 'F1' in Sensor Controller Studio or online.

Educational, hands-on training modules are available in Resource Explorer under SimpleLink Academy. The TI-RTOS training and Sensor Controller training are particularly useful when getting started with Sensor Controller development. The Project from Scratch' SimpleLink Academy training covers in detail how to integrate the SC code with the rest of your project.

When selecting 'Code Generator' in Sensor Controller Studio the following file is generated in the output directory: scif_how_to_use.html. This file describes how the SCIF driver files should be integrated in the rest of the code.

3 Performance Comparison

To show the efficiency of using the Sensor Controller, a number of scenarios were tested on a subset of development boards. Some of the scenarios were not tested on every piece of hardware, see Section 4 for a complete list of scenarios and development boards used. The different scenarios are explained in greater detail in their respective sections. For the MSP432P401R, *baremetal* refers to the task executing with the use of timers instead of task scheduling, for example without the use of a RTOS. The MSP432™ current consumption was measured both with and without direct memory access (DMA).

Section 4 lists the scenario comparison. A description of each scenario is given along with the measurement results in Section 4.

Table 3. Scenarios

	CC2642R1			CC1310		MSP432P401R		
Scenario	**TI-RTOS**	**SC 2 MHz**	**SC 24 MHz**	**TI-RTOS**	**SC 24 MHz**	**TI-RTOS**	**Baremetal 48 MHz With DMA**	**Baremetal 12 MHz, 48 MHz Without DMA**
SPI	✓	✓	✓	✓	✓	✓	✓	✓
ADC	✓		✓	✓	✓			
Comparator		✓	✓		✓			
Wake up and go back to sleep		✓	✓		✓			
Wake-up/shutdown	✓		✓					
Pin change interrupt	✓	✓	✓	✓	✓			

3.1 Scenario Descriptions

- **SPI – 18-Byte transmissions:** Transmit 18 bytes of data over SPI at the maximum data rate and go to sleep between transfers. This was done 1, 20, and 100 times per second. The purpose of this scenario was to highlight the new hardware SPI module that was added to the Sensor Controller in the CC26x2, as well as compare it to something outside of the CC13xx/CC26xx family. The MSP432P401R was chosen due to its low-power operation.
- **ADC – wake up and perform an ADC read operation:** Wake up and do a single read of the analog-to-digital converter (ADC) at 1, 20, and 100 times per second. This was not done with the 2-MHz oscillator, because the ADC is not enabled at that frequency.
- **Comparator reading:** Wake up and perform a comparator read 1, 20, and 100 times per second. The purpose of this scenario was to compare the difference in current consumption when using the comparator instead of the ADC for simple threshold operations.
- **Waking up and going back to sleep immediately:** The purpose of this was to determine how much the overhead affected the average current.
- **Wake-up/shutdown:** Wake up from standby and set a pin high, do operations for about 1 ms to allow the current consumption to stabilize, and then set the pin low again and go back to sleep. The purpose of this scenario was to measure the time and energy from when the processor wakes up until it starts executing user code and from the last user operation until the processor enters sleep. In order to determine when a wake-up started, the OBS_PWR_DWN signal was tied to a pin using signal observation on the Sensor Controller. This signal goes low when the Sensor Controller is active in 24-MHz mode, during a recharge and when the main processor is active. This is useful for measuring the wake-up and shutdown time, but it does not work for 2 MHz (see Figure 2).

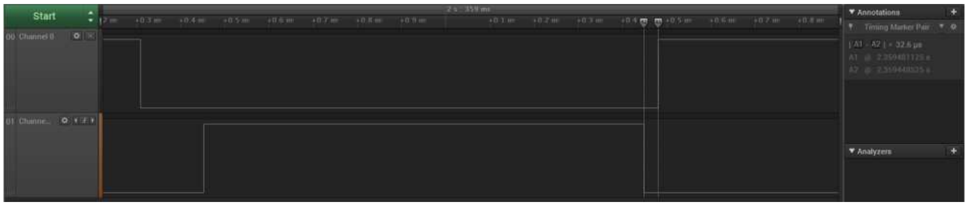

Figure 2. OBS_PWR_DWN (Top), Output Pin (Bottom)

- **Pin change interrupt:** Wake up from a falling edge on a pin and set an output pin high. Wait for 1 ms and then set the pin low again, go back to sleep and wait for next interrupt. The wake-up time was measured as the time it took from when the input signal went low until the output went high.

3.1.1 Measurement Conditions

A set of conditions are assumed in order to achieve consistent and reliable measurements that are replicable. To suppress external interferences, all jumpers were removed from the LaunchPads and the SmartRF06 evaluation board, and short wires were used when connecting to the DC power analyzer. Additionally, the logic analyzer was completely removed when performing current measurements. On the CC1310 board there is a flash memory IC which was also physically removed from the board prior to the measurements. The flash can be disabled through software and this should render results similar to when it was completely removed.

The current averages were measured using the Keysight 14585A Control and Analysis Software. The scope mode was used in single trigger mode, and markers were set from the start of one wake-up until the start of the next. Ten values were collected and averaged. Ensure to set the Range option to auto because the default range, 3A, results in poor resolution and inaccurate results (see Figure 3).

You can read more about power measurements in Measuring CC13xx and CC26xx Current Consumption.

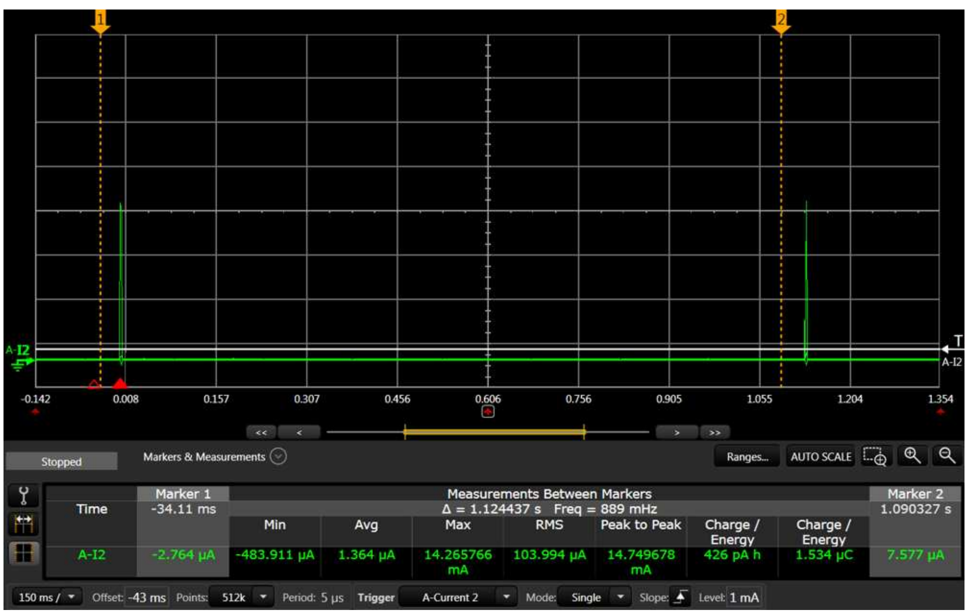

Figure 3. First Marker Before Application, Second Marker Before Next Application

4 Results

The following sections present the results of the comparative measurements between the Sensor Controller Engines and the Cortex CPUs.

4.1 SPI – 18-Byte Transmission

Table 4 lists the average current consumption for 18 byte SPI transmission. As expected, the CC2642R1 Sensor Controller has a lower current consumption. The difference is most striking when waking up 100 times per second.

Table 4. Average Current Consumption for 18-Byte SPI Transmission: 1, 20, and 100 Times Per Second

Processor Configuration	SPI Clock Frequency [MHz]	Wake-Ups Per Second			Units
		1 Time	20 Times	100 Times	
CC2642R1 SC, 2 MHz	1	1.0	1.4	3.0	µA
CC2642R1 SC, 24 MHz	12	1.5	4.0	15.6	µA
CC2642R1 Cortex-M4 running TI-RTOS	12	2.4	25.4	119.2	µA
CC1310 SC, 24 MHz	2	1.1	5.3	22.9	µA
CC1310 Cortex-M3 running TI-RTOS	12	2.6	32.8	158.3	µA
MSP432 running TI-RTOS	12	2.3	21.5	108.3	µA
MSP432 baremetal 48 MHz With DMA	16	1.2	5.8	26.2	µA
MSP432 baremetal 12 MHz, 48 MHz Without DMA	16	1.3	7.4	32.8	µA

Figure 4 shows the current consumption when performing SPI communication.

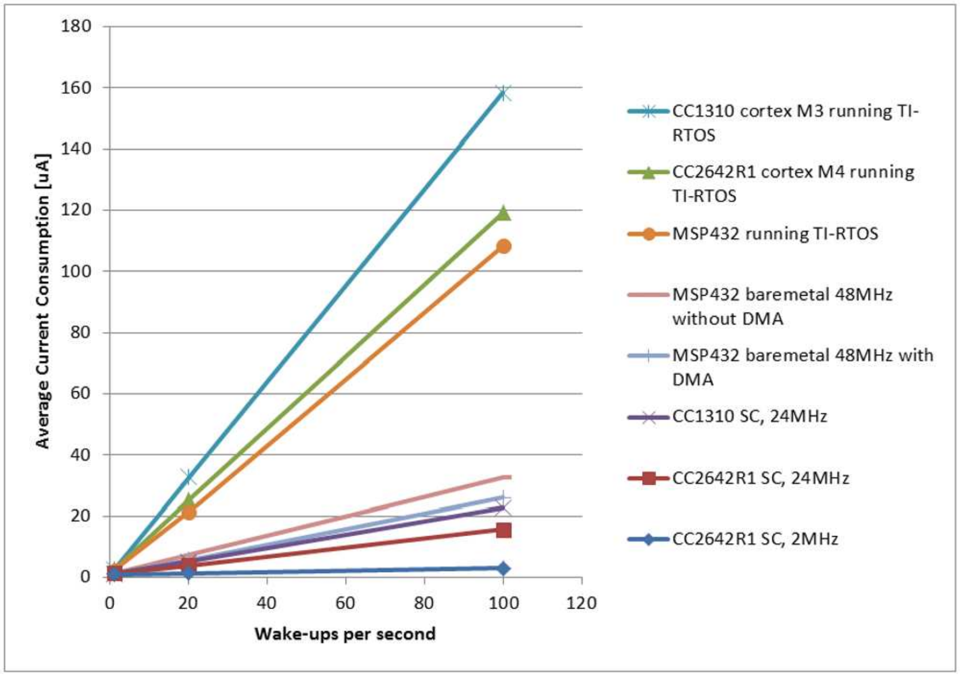

Figure 4. SPI Current Consumption

Figure 5 shows the current profile for a single SPI transaction.

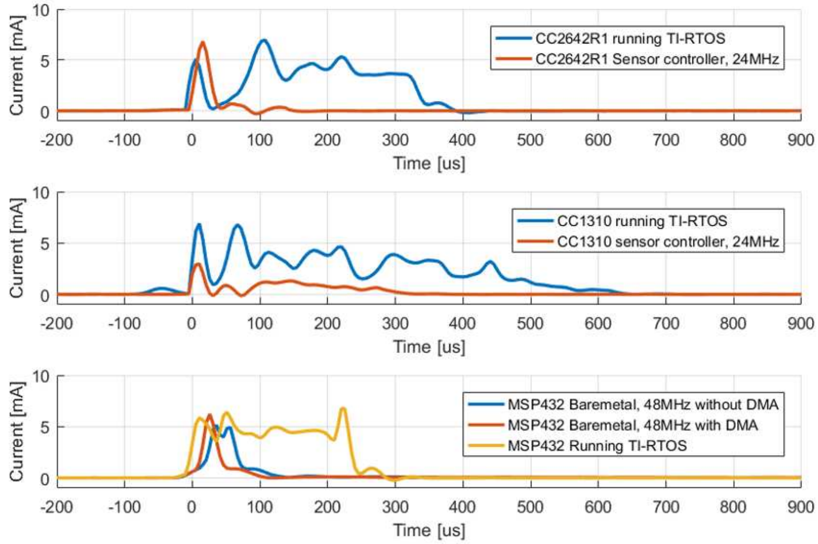

Figure 5. Current Traces for One Wake-Up of an SPI Application Running on Different Hardware

4.2 ADC – Wake Up and Perform an ADC Read Operation

Table 5 lists the average current for the ADC read operation.

Table 5. Average Current for ADC Read Operation: 1, 20, and 100 Times Per Second

Processor Configuration	Wake-Ups Per Second			Unit
	1 Time	20 Times	100 Times	
CC2642R1 SC, 24 MHz	1.4	4.4	17.4	µA
CC2642R1 Cortex-M4	2.6	28.7	118.3	µA
CC1310 SC, 24 MHz	1.1	4.6	19.2	µA
CC1310 Cortex-M3	2.4	28.1	137.8	µA

Figure 6 shows the current consumption when performing ADC conversions.

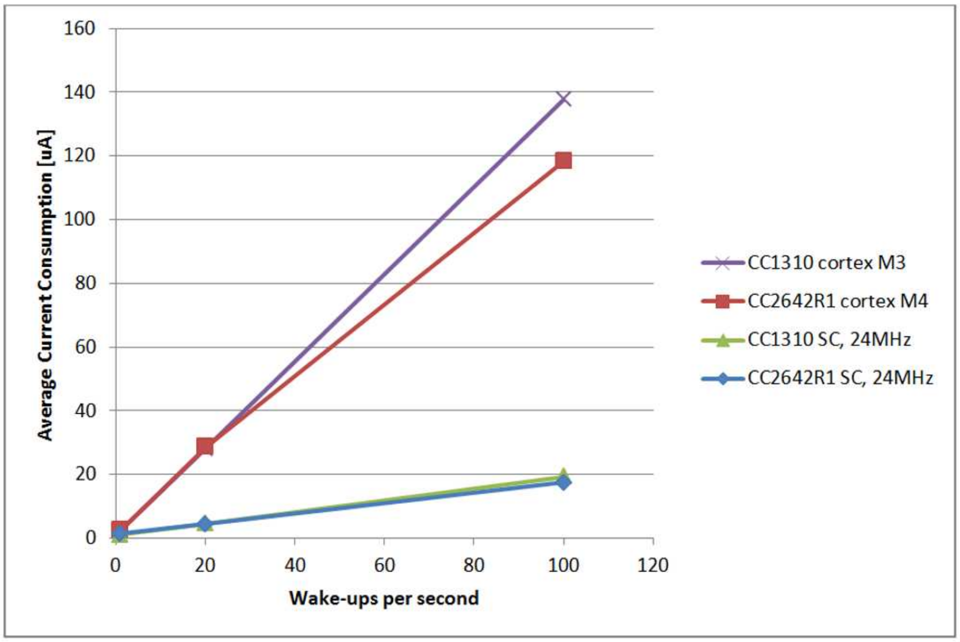

Figure 6. ADC Current Consumption

4.3 Comparator Reading

Table 6 lists the average current consumption for a comparator read.

Table 6. Average Current Consumption for Comparator Read: 1, 20, and 100 Times Per Second

Processor Configuration	Wake-Ups Per Second			Units
	1 Time	20 Times	100 Times	
CC2642R1 SC, 2 MHz	0.9	1.0	1.5	µA
CC2642R1 SC, 24 MHz	1.4	3.6	13.6	µA
CC1310 SC, 24 MHz	2.1	4.0	15.5	µA

Figure 7 shows the current consumption when running the comparator.

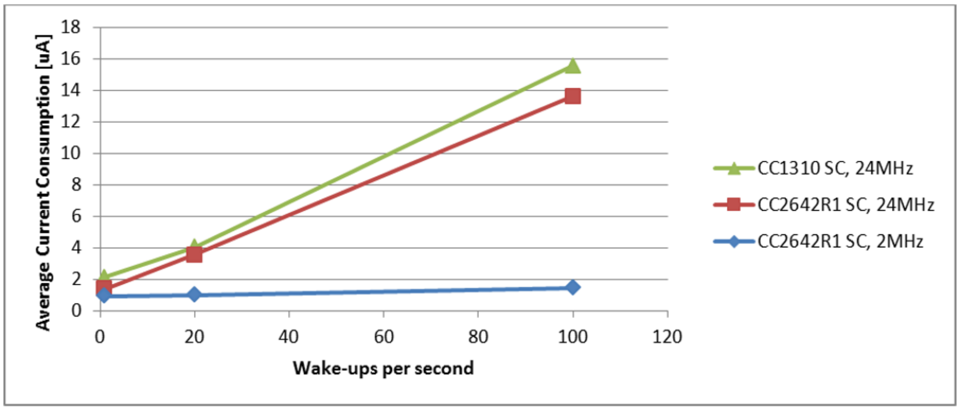

Figure 7. Comparator Current Consumption

4.4 Waking Up and Returning to Sleep Immediately

Table 7 lists the average current for waking up 1, 20, and 100 times per second.

Table 7. Average Current for Waking Up: 1, 20, and 100 Times Per Second

Processor Configuration	Wake-Ups Per Second			Units
	1 Time	20 Times	100 Times	
CC2642R1 SC, 2 MHz	1.0	1.0	1.2	µA
CC2642R1 SC, 24 MHz	1.4	3.4	13.0	µA
CC1310 SC, 24 MHz	2.2	4.1	15.7	µA

Figure 8 shows the current consumption for SC wake-ups only.

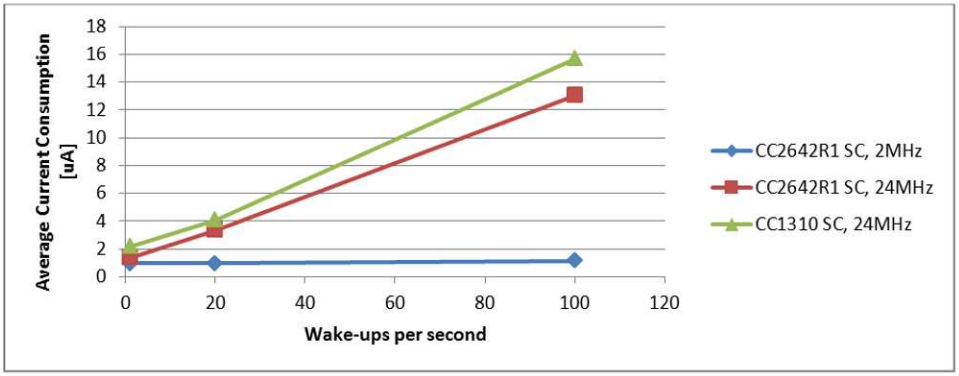

Figure 8. Wake-Up Current Consumption

4.5 Wake-Up and Shutdown

Table 8 lists the wake-up and shutdown time and energy for the CC2642R1 M4 and SC.

Table 8. Wake-Up and Shutdown Time and Energy for CC2642R1 M4 and SC

Parameter	CC2642R1 Sensor Controller, 24 MHz	CC2642R1 M4 Running TI-RTOS
Wake-up time	143 µs	282.3 µs
Wake-up energy	82.5 nC	628.1 nC
Shutdown time	32.6 µs	100.1 µs
Shutdown energy	14.4 nC	335.1 nC

Figure 9 shows a typical wake-up and shutdown.

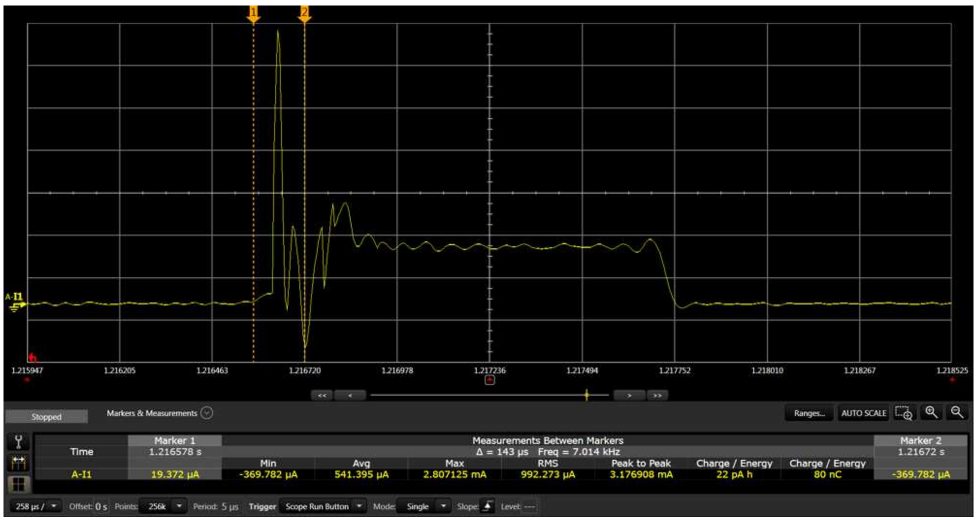

Figure 9. Typical Waveform for One Wake-Up and Shutdown, SC (1-ms Active Time)

Figure 10 shows the wake-up and shutdown times and energies.

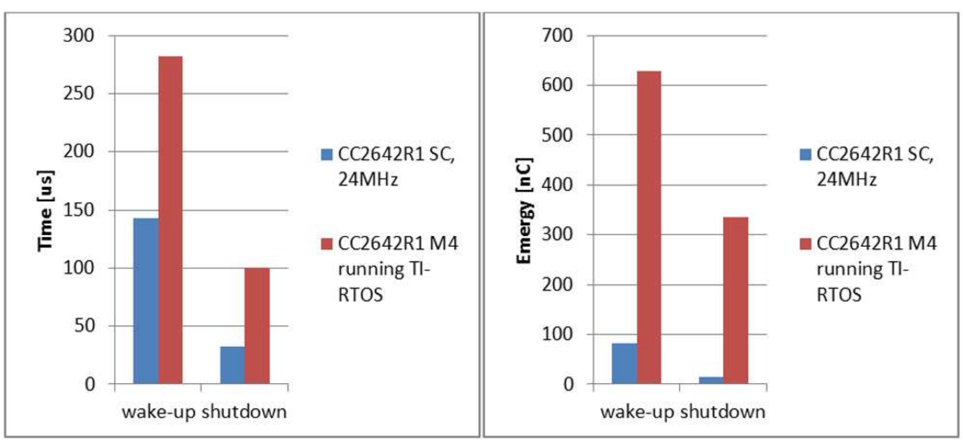

Figure 10. Wake-Up and Shutdown Time and Energy for SC and Cortex-M4 CPU

4.6 Pin Change Interrupt

Figure 11 shows the wake-up time when reacting to an interrupt for the SC and the Cortex CPUs.

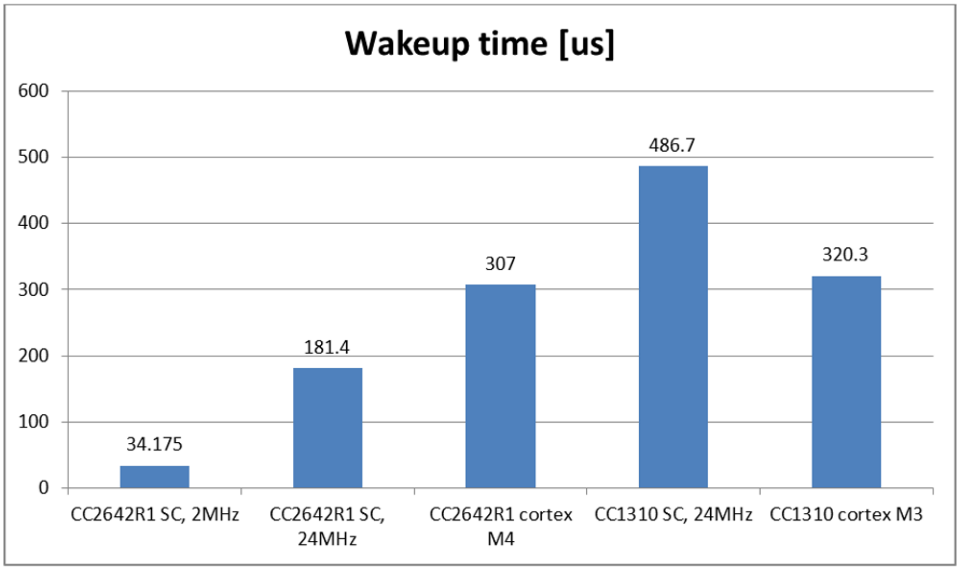

Figure 11. SC and Cortex CPU Interrupt Wake-Up Times

5 Conclusion

Judging from the results, the Sensor Controller has significantly lower power consumption compared to the main CPU in the CC2642R1, and CC1310 and MSP432 when performing data collection from sensors. However, there are some areas where the main CPU is the better option. Specifically, when performing complex mathematical operations and calculations, because the M3 and M4 are better suited for such tasks.

There are some interesting use cases for the Sensor Controller that have yet to be discussed in this application report. The first case is a method for further reducing power consumption of a system using the Sensor Controller. Usually, sensors do not need to be active constantly since they are usually read periodically. One way of reducing the power consumption of such an application is to only enable the sensors so that they are stable when the Sensor Controller wakes up to read them. In the CC2642R1 Sensor Controller, this can be done using Timer 2 without having to wake up to set the enable pins high. This is especially useful for applications that only poll sensors rarely, because the sensors themselves contribute to the overall power consumption.

Another use case for the Sensor Controller is to collect data and only wake the main processor for data processing. This use of the Sensor Controller offers two main benefits over using the main CPU exclusively. The first use being that since the Sensor Controller uses less power than the main CPU, the power used for collecting data is reduced significantly. When the Sensor Controller is done collecting data, it can generate an alert to wake the main CPU for processing the data. The second benefit that may not be as evident as the first, is that by using the SC to collect the data the main CPU is offloaded and free to do other tasks while the data is being collected. For example, the Sensor Controller can sample an ADC channel continuously and only wake the main processor when the signal passes a threshold.

The 2-MHz low-power mode in the CC2642R1 Sensor Controller enables even lower power consumption as well as quicker reaction to pin change interrupts. The trade-off is that some peripherals are not available in low-power mode and that the processing speed is slower.

Getting started using the Sensor Controller is easy, all that is required is an appropriate device SDK, as well as Sensor Controller Studio. For more details on how the Sensor Controller works, see the Sensor Controller Studio examples, help documentation, and user's guide, as well as the SimpleLink Academy training for the Sensor Controller.

6 References

1. Texas Instruments, SimpleLink Academy: Introduction to Sensor Controller Presentation
2. Texas Instruments, TI Resource Explorer
3. Texas Instruments, *Measuring Bluetooth Low Energy Power Consumption*
4. Texas Instruments, *CC13x0, CC26x0 SimpleLink™ Wireless MCU Technical Reference Manual*

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Original (October 2017) to A Revision **Page**