Application Report

SWRA585–March 2018

Low-Power ADC Solution for CC13x2 and CC26x2

Mandus Börjesson

ABSTRACT

The 12-bit, 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 (2 MHz) mode. The goal of this application note is to help users create an 8-bit, successive-approximation (SAR)-type, low-power, ADC using the CC13x2 and CC26x2 Sensor Controller running in 2-MHz mode. The comparator and digital-to-analog converter (DAC) peripherals are used to achieve a low-power ADC solution.

Contents

List of Figures

List of Tables

1 Introduction

Figure 1 shows the four major blocks of the SAR ADC. The sample and hold (S/H) block *locks* the analog voltage level during sampling. The comparator compares the analog signal to the DAC output, which is controlled by the SAR logic block. The function of the SAR logic is replicated using the Sensor Controller.

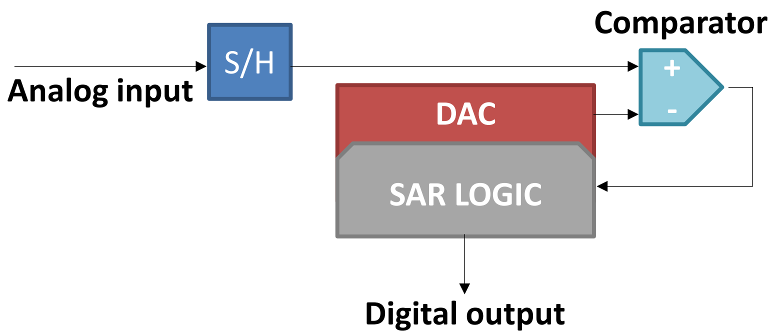

Figure 1. SAR ADC Block Diagram

As the name suggests, SAR ADCs try to predict the analog voltage using a comparator and a DAC. Figure 2 shows the workflow of a single, 5-bit, SAR, ADC sample. The DAC output is used as reference for the comparator and the analog voltage is the comparator input. Therefore, the comparator output tells if the input voltage is smaller or larger than the DAC output. Depending on the comparator output, the DAC output is increased or decreased by a step size that is half between iterations.

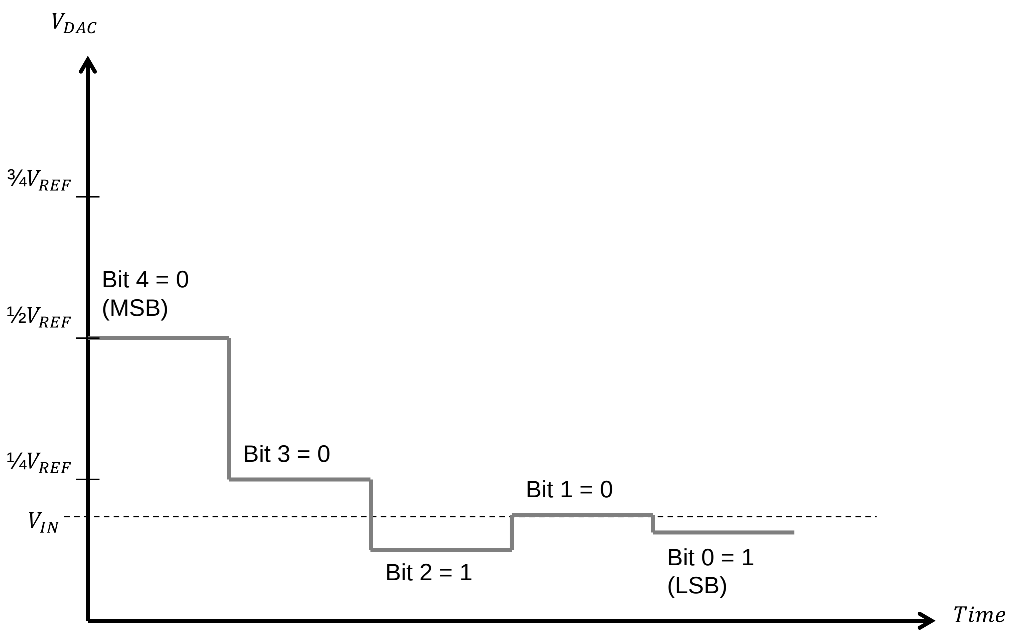

Figure 2. SAR Operating Principle

2 Code Implementation in Sensor Controller Studio

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

Data members:

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

Task resources:

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

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...
refdacWaitForStableOutput();

//An additional slight delay for DAC and comparator to stabilize...
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 Performance

The software-implemented ADC was tested by sampling a 100-Hz, 3-V, peak-to-peak, sine wave and calculating the effective number of bits (see Figure 3). One drawback of the software-implemented ADC is the lack of a sample/hold circuit, which results in a slight change of the signal during sampling. This results in a slight error in the lesser bits of the value. Table 1 lists the measured performance of the ADC. Figure 4 shows the digital output spectrum performance of the ADC implementation.

Table 1. Software-Implemented SAR ADC Performance

Current Consumption			
Sample frequency	1 Hz	20 Hz	100 Hz
Low-power, 2-MHz mode	0.9611 µA	1.33147 µA	2.9951 µA
Active, 24-MHz mode	1.4509 µA	4.9836 µA	20.1072 µA
Performance			
Maximum sampling rate	8.8 kHz		
Effective number of bits	6.3 bits		

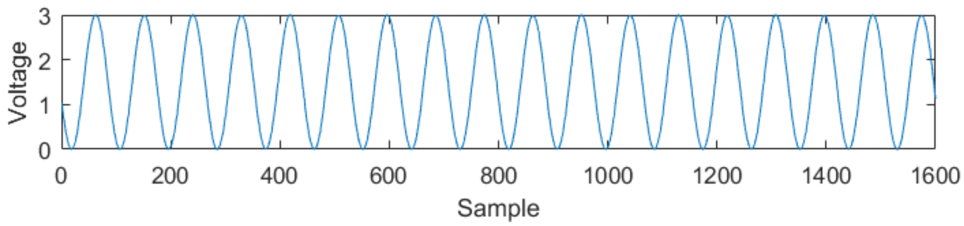

Figure 3. 1600 Samples of 100-Hz Sine Wave Sampled With Software-Implemented ADC

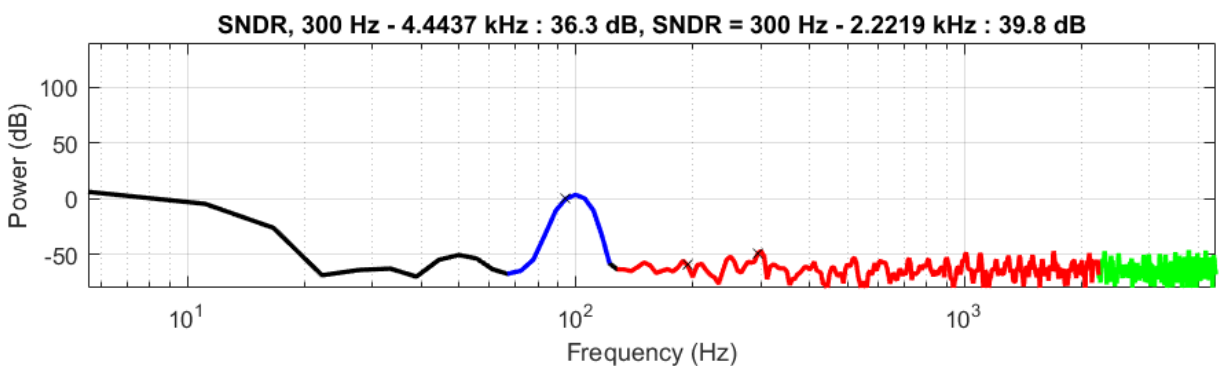

Figure 4. Digital Output Spectrum

Voltage Tracker

A.1 Overview

Because the software-implemented ADC was best suited for slow moving signals, due to the lack of a sample/hold circuit and the fairly low sampling rate, a tracking algorithm was attempted instead of the SAR logic, because it should consume less power and is still sufficient for sampling such signals.

A tracking ADC only reads the comparator output once upon waking up and tweaks the DAC output depending on the comparator output. If the DAC value is too low, the value is slightly increased and if it is too high, the value is slightly decreased. The stepping size is increased until the value has been found. The benefit of this algorithm is that the current consumption can be reduced, as compared to the SAR ADC, because the number of iterations each wakeup is reduced. The main drawback is that quick changes on the input signal may cause the algorithm to oscillate slightly before settling, as shown in Figure 5.

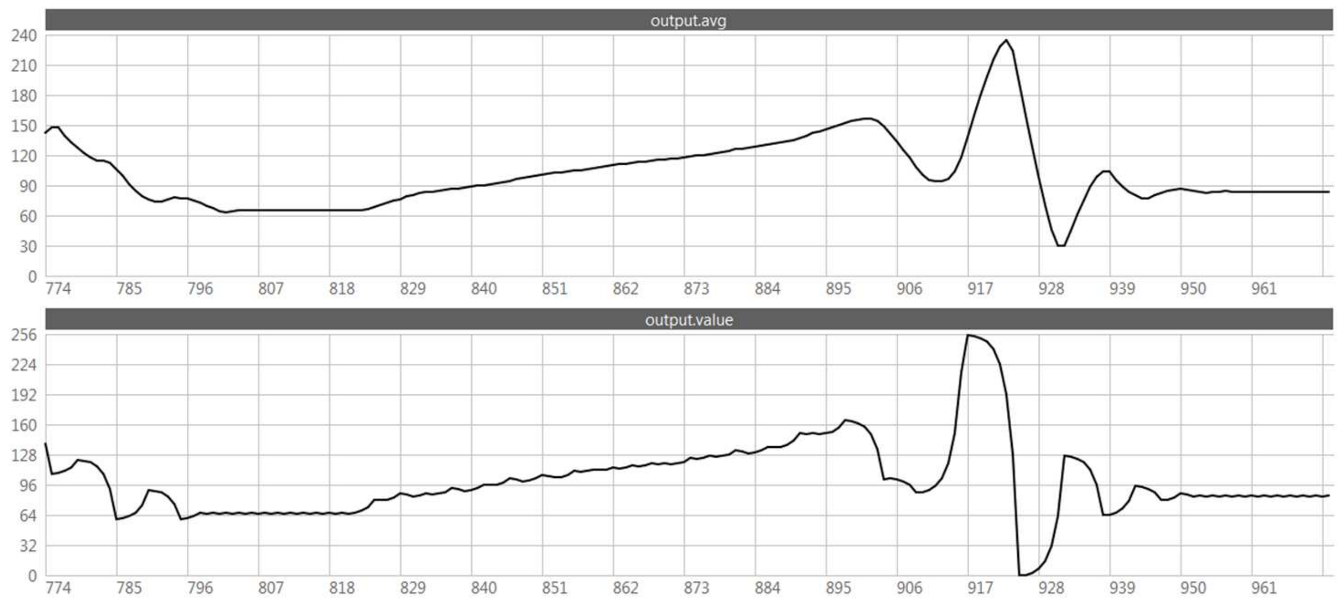

Figure 5. Voltage Follower Output With (Top) and Without (Bottom) 8-Value Averaging

Table 2 lists the voltage follower current consumption.

Table 2. Voltage Follower Current Consumption

Current Consumption			
Sample frequency	1 Hz	20 Hz	100 Hz
Low-power, 2-MHz mode	0.9362 µA	1.106 µA	1.8208 µA
Active, 24-MHz mode	1.4211 µA	4.4244 µA	17.6248 µA
Low-power, 2-MHz mode with averaging	0.9539 µA	1.1559 µA	2.0091 µA
Active, 24-MHz mode with averaging	1.4231 µA	4.4847 µA	18.0558 µA

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

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