EFM32
... the world's most energy friendly microcontrollers

Current Digital to Analog Converter

AN0064 - Application Note

Introduction

This application note describes how to use the EFM32 Current Digital to Analog Converter (IDAC), a peripheral that can source or sink a configurable constant current.

The general operation of the peripheral is explained and software examples include using PRS to control the IDAC, a calibration routine for a given current output and a demonstration on how to use the module to measure absolute capacitance.

This application note includes:

- This PDF document
- Source files (zip)
  - Example C-code for the Zero Gecko STK
  - Multiple IDE projects

ARM ZERO TINY GECKO LEOPARD GIANT WONDER

SILICON LABS

1 Current Digital to Analog Converter

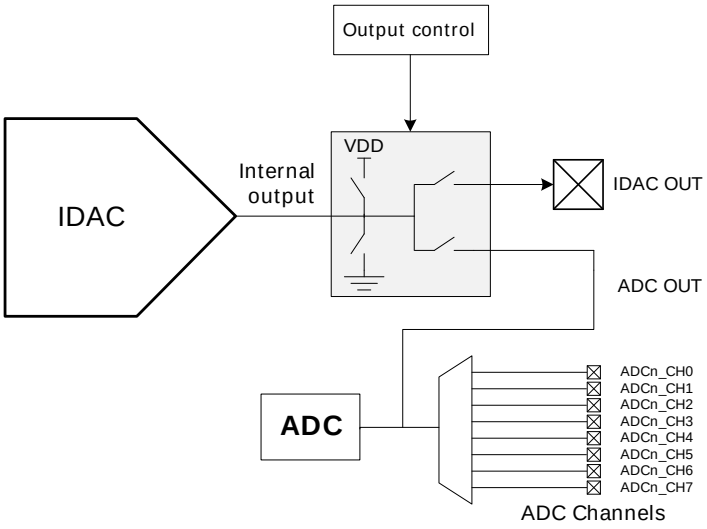
1.1 Introduction

The EFM32 IDAC can source or sink a configurable constant current from 50 nA to 64 µA. This can be either through the IDAC pin or an ADC channel pin.

1.2 Overview

An overview of the IDAC module is shown in Figure 1.1 (p. 2). The IDAC is designed to source or sink a programmable current. Controlling the IDAC output can be done through software or using PRS. The output from the IDAC can be sinked or sourced through the IDAC pin or the ADC. When using the ADC the current is sinked or sourced using the pin to the ADC channel that is currently in use.

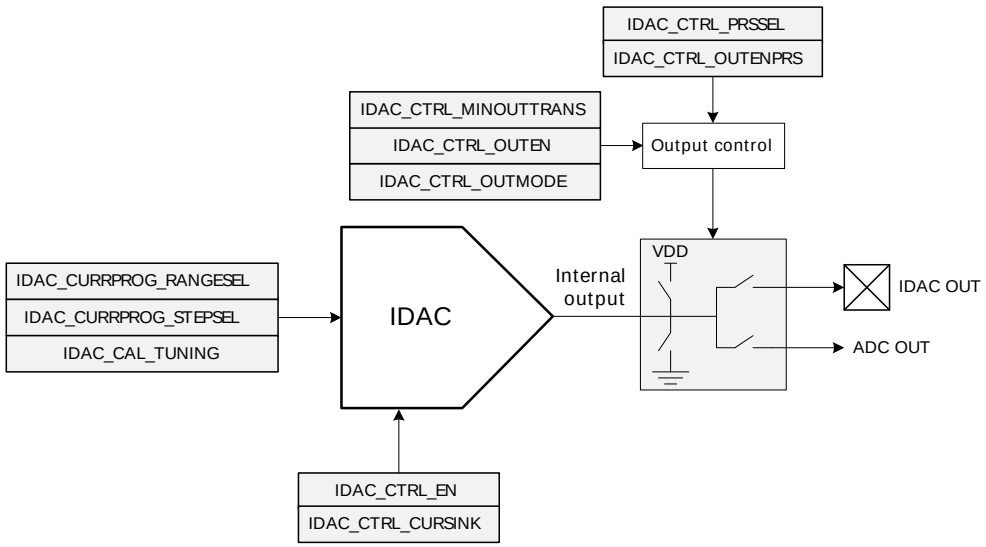
Figure 1.1. IDAC Overview



2 General Operation

A register overview of the IDAC module is shown in Figure 2.1 (p. 3) .

Figure 2.1. IDAC Register Overview



2.1 Current Selection

The 4 different current ranges can be selected by configuring the RANGESEL bitfield in IDAC_CURRPROG. Each range covers a different current interval and is linearly divided in 32 steps, which is configured by the STEPSEL bitfield in IDAC_CURRPROG. These current ranges with their step sizes are shown in Table 2.1 (p. 3) .

Table 2.1. Range Selection

Range Select	Range Value [µA]	Step Size [nA]	Step Counts
0	0.05 - 1.6	50	32
1	1.6 - 4.7	100	32
2	0.5 - 16	500	32
3	2 - 64	2000	32

2.2 Turning the IDAC on/off

The IDAC output can be controlled either by software or PRS. After enabling the IDAC module the current output can be turned on by setting the OUTEN in IDAC_CTRL. The IDAC output pin must be disabled in the appropriate GPIO_MODE register for the IDAC to work.

2.3 Output through IDAC pin

When the OUTMODE in the IDAC CTRL register is set to PIN the current output will go through the IDAC PIN when the current is turned on. First the IDAC must be enabled through the EN bit in the CTRL register, then the current can be turned on and off with the OUTEN bit in the same register.

2.4 Output through ADC channel pin

The output can also be set to go through one of the ADC channel pins, this is done by setting the OUTMODE to ADC. When the ADC starts a scan the IDAC output will go out on the selected channel.

The IDAC output will stop after the scan is done, unless the CHCONIDLE bit in ADCn_CTRL is set. Then the output will be on until a new ADC channel is scanned or the IDAC is turned off.

2.5 PRS

The IDAC can be turned on and off by the PRS. An example use case is setting up the timer to generate a PWM signal that turns the IDAC on and off using PRS as seen in the included software example PRS Triggered Charge Injection. To let the PRS take control of the IDAC set OUTENPRS in IDAC_CTRL and select the PRS channel to control it using the PRSSEL bit field.

2.6 Calibration

To get the highest performance out of the IDAC it should be calibrated for offset error. This is where the entire range is offset by a constant amount. The IDAC can be calibrated using the IDAC_CAL_TUNING register. By adjusting the register value the offset error can be calibrated away. In production, the middle step of every range is calibrated at room temperature and the calibration values are stored in the IDAC_CALx registers in the Device Information Page. When the range is set using the emlib IDAC_RangeSet function the calibration value is automatically loaded from flash into the IDAC Calibration register.

2.7 Minimizing Output Transition

The internal output of the IDAC may be at a different voltage than the output pin. By setting the MINOUTTRANS bit in IDAC_CTRL the internal output is lowered to GND or risen to VDD depending on if the IDAC is in current source or sink mode, thereby minimizing unwanted peaks in the current when switching the output on. When the IDAC output is enabled the charge/discharging will stop until OUTEN is cleared. Only the internal net of the IDAC is affected by this setting. If there is external components connected to the IDAC pin, for example a capacitor that needs to be discharged, the pin should be lowered to GND by setting it as a GPIO output for enough time to let the capacitor discharge, then it can be configured back to be used by the IDAC.

3 Software Examples

3.1 IDAC Demo

This demo allows the user to cycle through all the IDACs current range and step selections, sending the current out on IDAC_OUT(PB11). Incrementing the range and step register is done by push button 1 and 2 respectively, while the selection is presented on the kit's LCD along with the current output. The software assumes that the IDAC is correctly calibrated and displayes only the expected current output.

3.2 Calibration Routine

In production the IDAC is calibrated at room temperature. If the temperature where the MCU is used in varies from this, the calibration value might be off. This software example uses the ADC to measure the current sent through an external resistor connected to ADC channel pin, see Figure 3.1 (p. 6). The appropriate resistor value is dependent on the range the user wants to calibrate for since the difference between the lowest and highest current output is three orders of magnitude. The resistor value must be chosen so that the resulting voltage over it can be measured by the ADC with the limiting factor that the maximum current output for that range does not make the voltage over the resistor exceed the supply voltage. See Table 3.1 (p. 5) for suggested resistor values. To get the most accurate results the ADC should be calibrated before running the IDAC calibration routine.

Table 3.1. Resistor Selection

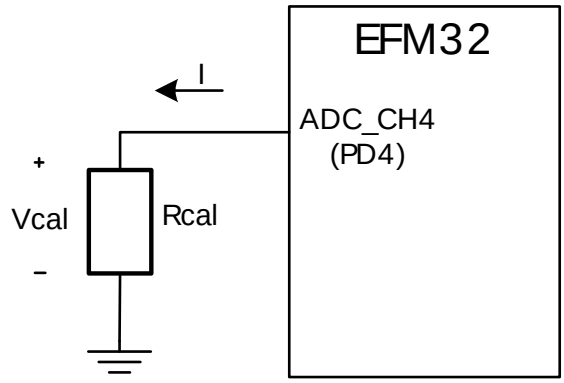
Current Interval	Resistor Value
50 nA - 1.6 µA	500k Ohm
1 µA - 64 µA	18k Ohm

For a given current range and step the value of the resistor and the voltage over it is known. This lets the software, using Ohm's law(Equation 3.1 (p. 5)), find the current going through the resistor, tuning the value in the IDAC Calibration register until the difference between the current and the target value is minimized. The function `calibrateIdac(range, step, resistor, adcCh)` takes the range and step values that is to be calibrated for, the value of the resistor and which ADC channel to use. This lets the user connect different value resistors to different ADC channel pins allowing for calibration of all four current ranges.

Ohm's law

$$I = U_{cal} / R_{cal} \quad (3.1)$$

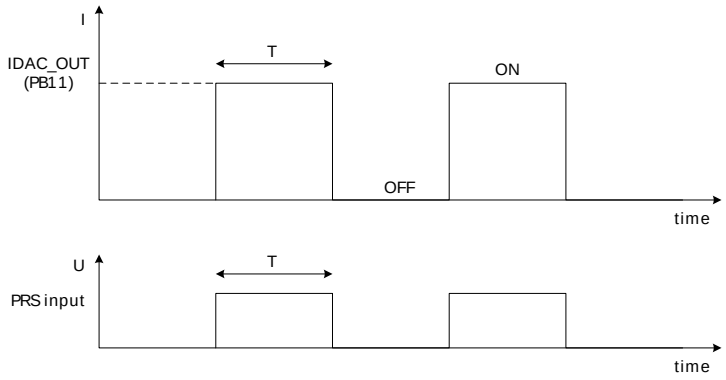
Figure 3.1. Calibration Circuit



3.3 PRS Triggered Charge Injection

This example demonstrates how to use a timer to control the IDAC through the PRS. The IDAC is configured to enable its output based on the signal it receives from PRS channel 0. The timer generates a square waveform signal that is sends to the PRS. See Figure 3.2 (p. 6) .

Figure 3.2. IDAC Charge Injection Example



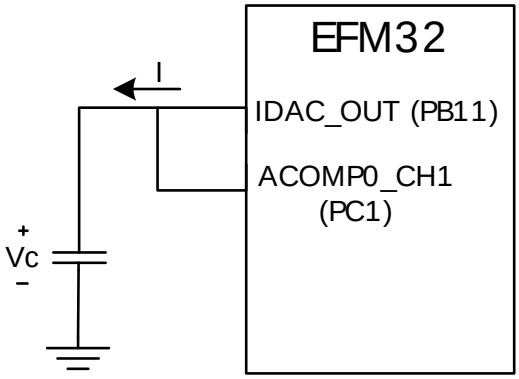
3.4 Capacitance Measurement

This example uses the IDAC, ACMP and TIMER module to measure the capacitance of a capacitor. See Figure 3.1 (p. 6) for the circuit diagram. The IDAC is used to charge the capacitor until the analog comparator detects that the voltage has reached a predetermined value, then the ACMP interrupt stops the timer. Since the current, voltage and time is known the capacitance can be calculated using Equation 3.2 (p. 6). When the IDAC pin is not charging the capacitor , the pin is pulled to ground to discharge the capacitor. This software example works best with capacitors above 1 µF.

Capacitance formula when current is constant

$$C = I\,T / V_C \tag{3.2}$$

Figure 3.3. Capacitance Measuring Circuit



4 Revision History

4.1 Revision 1.01

2014-05-07

Updated example code to CMSIS 3.20.5

Changed source code license to Silicon Labs license

Added project files for Simplicity IDE

Removed makefiles for Sourcery CodeBench Lite

4.2 Revision 1.00

2013-10-08

Initial revision.

A Disclaimer and Trademarks

A.1 Disclaimer

Silicon Laboratories intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Laboratories products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Laboratories reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Laboratories shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products must not be used within any Life Support System without the specific written consent of Silicon Laboratories. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Laboratories products are generally not intended for military applications. Silicon Laboratories products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

A.2 Trademark Information

B Contact Information

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701

Please visit the Silicon Labs Technical Support web page:
http://www.silabs.com/support/pages/contacttechnicalsupport.aspx
and register to submit a technical support request.

Table of Contents

List of Figures

List of Tables

List of Equations

silabs.com


ZERO
ARM Cortex-M0+
TINY
ARM Cortex-M3
GECKO
ARM Cortex-M3
LEOPARD
ARM Cortex-M3
GIANT
ARM Cortex-M3
WONDER
ARM Cortex-M4