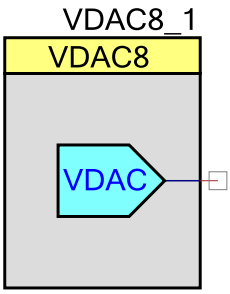
8-Bit Voltage Digital to Analog Converter (VDAC8)

1.50

Features

- Voltage output ranges: 1.020 V and 4.080 V full scale
- Software or clock driven output strobe
- Data source may be CPU, DMA, or UDB



General Description

The VDAC8 component is an 8-bit voltage output Digital to Analog Converter (DAC). The output range may be from 0 to 1.020 Volts (4 mV/bit) or from 0 to 4.08 Volts (16 mV/bit). The VDAC8 may be controlled by hardware, software, or a combination of both hardware and software.

Input/Output Connections

This section describes the various input and output connections for the VDAC8. An asterisk (*) in the list of I/Os indicates that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.

Vout – Analog

The Vout terminal is the connection to the DAC's voltage output. It may be routed to any analog compatible pin on the PSoC.

data[7:0] – Input *

This 8-bit wide data signal connects the VDAC8 directly to the DAC Bus. The DAC Bus may be driven by UDB-based components or control registers, or it may be routed directly from GPIO pins. This input is enabled by setting the **Data_Source** parameter to "DAC Bus". If the "CPU or DMA" option is selected instead, the bus connection will disappear from the component symbol.

Use the data[7:0] input when hardware is capable of setting the proper value without CPU intervention. When using this option, the strobe option should be set as "External" as well.

For many applications this input is not required, but instead the CPU or DMA will write a value directly to the data register. In firmware, use the SetRange() function or directly write a value to the VDAC8 data register.

strobe – Input *

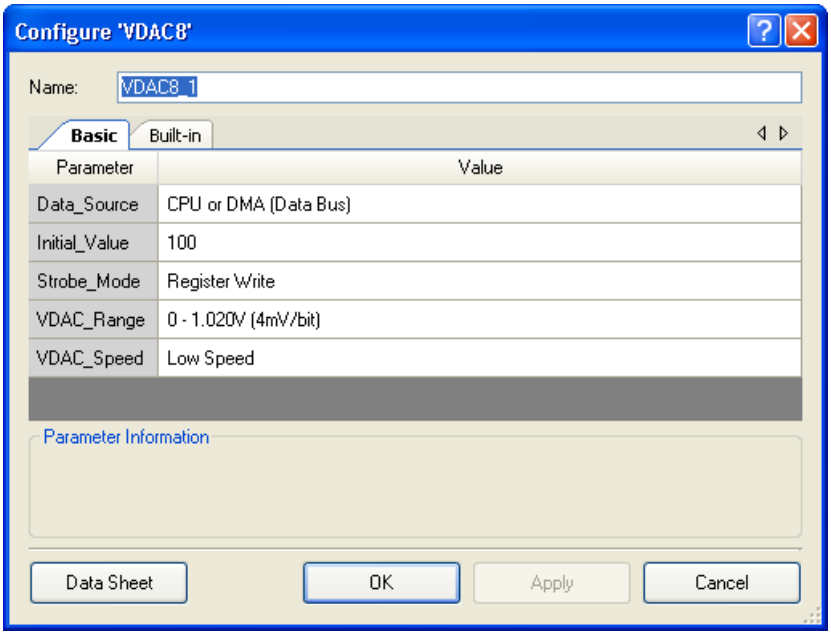
The strobe input is an optional signal input and is selected with the **Strobe_Mode** parameter.

- If **Strobe_Mode** is set to "External," the strobe pin will be visible and must be connected to a valid digital source. In this mode the data is transferred from the VDAC8 register to the DAC on the next positive edge of the strobe signal.
- If **Strobe_Mode** is set to "Register Write," the pin will disappear from the symbol and any write to the data registers will be immediately transferred to the DAC.

For audio or periodic sampling applications, the same clock used to clock the data into the DAC could also be used to generate an interrupt. Each rising edge of the clock would transfer data to the DAC and cause an interrupt to get the next value loaded into the DAC register.

Parameters and Setup

Drag a VDAC8 component onto your design and double-click it to open the Configure dialog.



The VDAC8 component provides the following parameters.

Data_Source

This parameter selects the source of the data to be written into the DAC register. If the CPU (firmware) or the DMA will write data to the VDAC8, select "CPU or DMA." If data is written directly from the UDBs or a UDB-based component, select "DAC Bus."

When DAC Bus is selected, the input is indicated on the VDAC symbol. There is only one DAC Bus, so multiple VDACs cannot have independent hardware (UDB) data sources.

Initial_Value

This is the initial value the VDAC8 will present after the Start() command is executed. The SetValue() function or a direct write to the DAC register will override the default value at anytime. Legal values are between 0 and 255, inclusive.

Strobe_Mode

This parameter selects whether the data is immediately written to the DAC as soon as the data is written into the VDAC8 data register. This mode is selected when the "Register Write" option is selected. When the "External" option is selected, a clock or signal from the UDBs controls when the data is written from the DAC register to the actual DAC.

VDAC_Range

This parameter allows you to set one of two voltage ranges as the default value. The range may be changed at any time during runtime with the SetRange() function.

Range	Lowest Value	Highest Value	Step Size
Range_1_Volt	0.0 mV	1.020 V	4 mV
Range_4_Volt	0.0 mV	4.080 V	16 mV

Output equations:

- 1 Volt range – Vout = (value/256) * 1.024 Volts
- 4 Volt range – Vout = (value/256) * 4.096 Volts

Note The term "value" is a number between 0 and 255.

VDAC_Speed

This parameter provides two settings: "Low Speed" and "High Speed." In "Low Speed" mode, the settling time is slower but consumes less operating current. In "High Speed" mode, the voltage settles much faster, but at a cost of more operating current.

Resources

The VDAC8 component uses one viDAC8 analog block.

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name "VDAC8_1" to the first instance of a component in a given design. You can rename the instance to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance name used in the following table is "VDAC8".

Function	Description
void VDAC8_Init(void)	Initializes or Restores default VDAC8 configuration
void VDAC8_Enable(void)	Enable the VDAC8.
void VDAC8_Start(void)	Initialize the VDAC8 with default customizer values.
void VDAC8_Stop(void)	Disables the VDAC8 and sets it to the lowest power state.
void VDAC8_SetSpeed(uint8 speed)	Set DAC speed.
void VDAC8_SetValue(uint8 value)	Sets value between 0 and 255 with the given range.
void VDAC8_SetRange(uint8 value)	Sets range to 1 or 4 volts.
void VDAC8_Sleep(void)	Stops and saves the user configuration.
void VDAC8_WakeUp(void)	Restores and enables the user configuration.
void VDAC8_SaveConfig(void)	Empty function. Provided for future usage.
void VDAC8_RestoreConfig(void)	Empty function. Provided for future usage.

Global Variables

Variable	Description
VDAC8_initVar	Indicates whether the VDAC8 has been initialized. The variable is initialized to 0 and set to 1 the first time IDAC8_Start() is called. This allows the component to restart without reinitialization after the first call to the IDAC8_Start() routine.

	If reinitialization of the component is required, then the IDAC8_Init() function can be called before the IDAC8_Start() or IDAC8_Enable() function.

void VDAC8_Init(void)

Description: Initializes/restores default VDAC8 configuration.

Parameters: None

Return Value: None

Side Effects: All registers will be set to their initial values. This will re-initialize the component. Calling the Init() function requires a call to SetValue() if you intend to set a new value other than what is currently in the register.

void VDAC8_Enable(void)

Description: Enables the VDAC8.

Parameters: None

Return Value: None

Side Effects: None

void VDAC8_Start(void)

Description: Initialize the VDAC8 with default customizer values. Enable and power up the VDAC8 to the given power level. A power level of 0 is the same as executing the stop function.

Parameters: None

Return Value: None

Side Effects: None

void VDAC8_Stop(void)

Description: Powers down VDAC8 to lowest power state and disables output.

Parameters: None

Return Value: None

Side Effects: None

void VDAC8_SetSpeed(uint8 speed)

Description: Set DAC speed.

Parameters: (uint8) speed: Sets DAC speed, see table below for valid parameters.

Option	Description
VDAC8_LOWSPEED	Low speed (low power)
VDAC8_HIGHSPEED	High speed (high power)

Return Value: None

Side Effects: None

void VDAC8_SetRange(uint8 range)

Description: Sets range to 1 or 4 volts.

Parameters: (uint8) range: Sets full scale range for VDAC8. See table below for ranges.

Option	Description
VDAC8_RANGE_1V	Set full scale range of 1.020 V
VDAC8_RANGE_4V	Set full scale range of 4.080 V

Return Value: None

Side Effects: None

void VDAC8_SetValue(uint8 value)

Description: Sets value to output on VDAC8. Valid values are between 0 and 255.

Parameters: (uint8) value: Value between 0 and 255. A value of 0 is the lowest (zero) and a value of 255 is the full scale value. The full scale value is dependent on the range which is selectable with the SetRange API.

Return Value: None

Side Effects: On PSoC 3 ES2 and PSoC 5 ES1 silicon, the SetValue() function should be called after enabling power to the VDAC.

void VDAC8_Sleep(void)

Description: Stops the operation. Saves the user configuration and the component enable state. Should be called just prior to entering sleep.

Parameters: None

Return Value: None

Side Effects: None

void VDAC8_Wakeup(void)

Description: Restores the configuration registers and component enable state. Should be called just after awaking from sleep.

Parameters: None

Return Value: None

Side Effects: Calling this function without previously calling VDAC8_Sleep() may lead to unpredictable behavior.

void VDAC8_SaveConfig(void)

Description: Saves the user configuration.

Parameters: None

Return Value: None

Side Effects: Empty function. Implemented for future usage. No effect on component by calling this function.

void VDAC8_RestoreConfig(void)

Description: Restores the user configuration.

Parameters: None

Return Value: None

Side Effects: Empty function. Implemented for future usage. No effect on component by calling this function.

DMA Wizard

VDAC8 components do not require implementation of a DMA Request signal. The typical usage is signal generation and the data rate to VDAC8 components should be controlled externally. The DMA Wizard can be used to configure DMA operation as follows:

Name of DMA source / destination in DMA Wizard	Direction	DMA Req Signal	DMA Req Type	Description
VDAC8_Data_PTR	Destination	N/A	N/A	Stores the DAC value between 0 to 255

Sample Firmware Source Code

The following is a C language example demonstrating the basic functionality of the VDAC8 component. This example assumes the component has been placed in a design with the default name "VDAC8_1."

Note If you rename your component you must also edit the example code as appropriate to match the component name you specify.

```
#include <device.h>

void main()
{
    VDAC8_1_Start();                        // Enable VDAC8
    VDAC8_1_SetRange(VDAC8_1_RANGE_1V);     // Set full scale range to 1.024V
    VDAC8_1_SetValue(100);                  // Set value to 400 mV
}
```

Functional Description

When used as a VDAC8, the viDAC8 analog block is configured as voltage DAC and can be used as voltage source.

When used as a VDAC, the output is an 8-bit digital-to-analog conversion voltage to support applications where reference voltages are needed. Here the reference source is a voltage reference from the Analog reference block called VREF(DAC). The DAC can be configured to work in voltage mode by setting the DACx_CR0 [4] register. In this mode, there are two output ranges selected by the DACx_CR0[3:2] register:

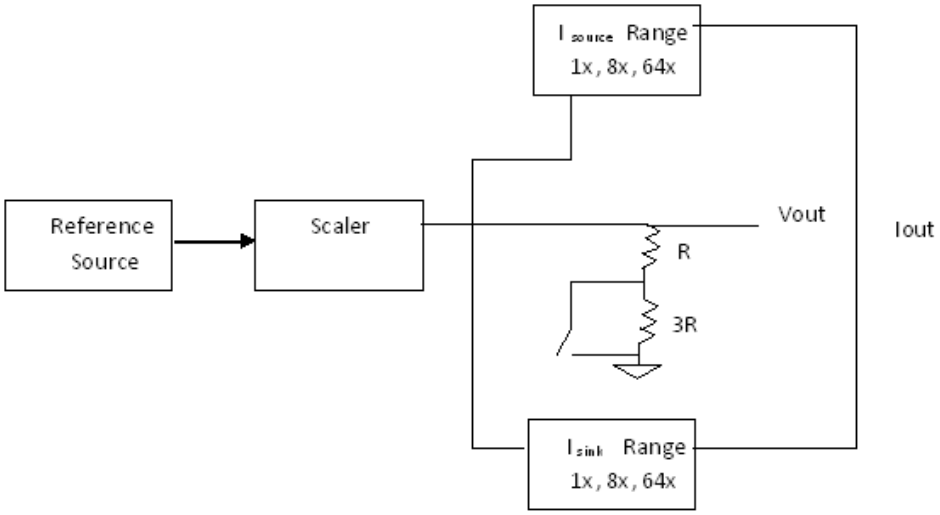
- 0 V to 1.024 V
- 0 V to 4.096 V

Both output ranges have 255 equal steps. The VDAC is implemented by driving the output of the current DAC through resistors and obtaining a voltage output. Because no buffer is used, any DC current drawn from the DAC affects the output level. Therefore, in this mode any load connected to the output should be capacitive.

The VDAC is capable of converting up to 1 Msps. In addition, the DAC is slower in 4 V mode than 1 V mode, because the resistive load to Vssa is 4 times larger. In 4 V mode, the VDAC is capable of converting up to 250 ksps.

Block Diagram and Configuration

The following shows the block diagram for the VDAC8 component.



Registers

The functions provided support most of the common runtime functions required for most applications. The following register references provide brief descriptions for the advanced user. The VDAC8_Data register may be used to write data directly to the DAC without using the API. This may be useful for either the CPU or DMA.

Table 1 VDAC8_CR0

Bits	7	6	5	4	3	2	1	0
Value	RSVD			mode	Range[1:0]		hs	RSVD

- mode: Sets DAC to either voltage or current mode.
- range[1:0]: DAC range settings.
- hs: Use to set data speed.

Table 2 VDAC8_CR1

Bits	7	6	5	4	3	2	1	0
Value	RSVD		mx_data	reset_udb_en	mx_idir	idirbit	Mx_ioff	ioffbit

- mx_data: Select data source.

- reset_udb_en: DAC reset enable.
- mx_idir: Mux selection for DAC current direction control.
- idirbit: Register source for DAC current direction.
- mx_off: Mux selection for DAC current off control.
- ioffbit: Register source for DAC current off

Table 3 VDAC8_DATA

Bits	7	6	5	4	3	2	1	0
Value	Data[7:0]							

- Data[7:0]: DAC data register.

DC and AC Electrical Characteristics

The following values are based on characterization data. Specifications are valid for -40°C ≤ T_A ≤ 85°C and T_J ≤ 100°C except where noted. Unless otherwise specified in the tables below, all Typical values are for T_A = 25°C, Vdda = 5.0V, output referenced to analog ground (Vssa), fast mode.

Note More specifications at other voltages and graphs may be added after characterization.

5.0 V/3.3 V DC Electrical Characteristics

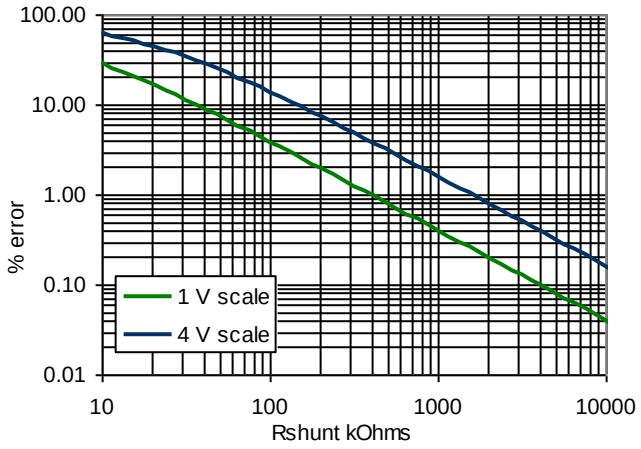
Parameter	Description	Conditions	Min	Typ	Max	Units
Resolution				8		bits
INL1	Integral non-linearity 1 V scale		na	tbc	tbc	LSB
INL4	Integral non-linearity 4 V scale		na	tbc	tbc	LSB
DNL1	Differential non-linearity 1 V scale		na	tbc	tbc	LSB
DNL4	Differential non-linearity 4 V scale		na	tbc	tbc	LSB
Rout	Output resistance					
	High	Vout = 4 V		16		K Ohms

Parameter	Description	Conditions	Min	Typ	Max	Units
	Low	Vout = 1 V		4		K Ohms
Vout	Output Voltage range					
	High	Code = 255, Vdda ≥ 5 V		4		V
	Low	Code = 255		1		V
Monotonicity			na	tbc	YES	
Vos	Zero-scale error		na	tbc	tbc	mV
TCVos	Temp. coeff. input offset voltage, absolute value		na	tbc	tbc	uV/°C
FSGainErr1	Full scale gain error, 1 Volt Range		na	tbc	tbc	%
FSGainErr4	Full scale gain error 4 Volt Range		na	tbc	tbc	%
TCGain1	Temp coeff gain, 1 V Range		na	tbc	tbc	%FSR/degC
TCGain4	Temp coeff gain, 4 V Range		na	tbc	tbc	%FSR/degC
IddSlow	Operating current	Slow mode, Typical at mid scale	na	tbc	na	uA
	Operating current	Slow mode, Max at full scale	na	na	tbc	uA
IddFast	Operating current	Fast mode, Typical at mid scale	na	tbc	na	uA
	Operating current	Fast mode, Max at full scale	na	na	tbc	uA
PSRR_DC	Power supply rejection ratio, DC	Measured as shift in voltage at mid-scale	tbc	tbc		dB

Figures

INL vs DAC Code, 1.0V Range X axis code, bits Y axis INL counts	INL vs DAC Code, 4.0V Range X axis code, bits Y axis INL counts
DNL vs DAC Code, 1.0V Range X axis code, bits Y axis DNL counts	DNL vs DAC Code, 4.0V Range X axis code, bits Y axis DNL counts
INL vs Temperature, 1.0V Range X axis code, bits Y axis INL counts	INL vs Temperature, 4.0V Range X axis code, bits Y axis INL counts
DNL vs Temperature, 1.0V Range X axis code, bits Y axisDNL counts	DNL vs Temperature, 4.0V Range X axis code, bits Y axis DNL counts
Full-Scale Error vs Temp, 1.0V Range X axis temp -40 to 85 C Y axis max |%| deviation from nominal 1 Slow Mode 2 Fast Mode	Full-Scale Error vs Temp, 4.0V Range X axis temp -40 to 85 C Y axis max |%| deviation from nominal 1 Slow Mode 2 Fast Mode
Operating current vs code, 1.0 V Range X axis Code 0 to 255 Y axis Op current uA 1 Slow Mode 2 Fast Mode	Operating current vs code, 4.0 V Range X axis Code 0 to 255 Y axis Op current uA 1 Slow Mode 2 Fast Mode
Operating current vs temp, 1.0 V Range Slow mode, Code = 255 X axis Temp, -40 to +85C Y axis op current uA 1 Typ at 2.7V 2 Max at 2.7V 3 Typ at 5.5V 4 Max at 5.5V	Operating current vs temp, 4.0 V Range Slow mode, Code = 255 X axis Temp, -40 to +85C Y axis op current uA 1 Typ at 5.5V 2 Max at 5.5V
Operating current vs temp, 1.0 V Range Fast mode, Code = 255 X axis Temp, -40 to +85C Y axis op current uA 1 Typ at 2.7V 2 Max at 2.7V 3 Typ at 5.5V 4 Max at 5.5V	Operating current vs temp, 4.0 V Range Fast mode, Code = 255 X axis Temp, -40 to +85C Y axis op current uA 1 Typ at 5.5V 2 Max at 5.5V

Scale Error vs Shunt Load Resistance



INL vs Vdd, 4.0 V Range, Fast mode
X axis code, bits, 0xc0 to 0xff
Y axis Volts

1 Vdd = 4.5
2 Vdd = 5.0
3 Vdd = 5.5V

5.0 V/3.3 V AC Electrical Characteristics

Parameter	Description	Conditions	Min	Typ	Max	Units
Tsettle1P	Settling time to 0.1%, 1 V scale	Step 25% to 75%, Cload=15 pF	na	tbc	tbc	usec
Tsettle1N	Settling time to 0.1%, 1 V scale	Step 75% to 25%, Cload=15 pF	na	tbc	tbc	usec
Tsettle4P	Settling time to 0.1%, 4 V scale	Step 25% to 75%, Cload=15 pF	na	tbc	tbc	usec
Tsettle4N	Settling time to 0.1%, 4 V scale	Step 75% tp 25%, Cload=15 pF	na	tbc	tbc	usec
SR1P	Slew rate, positive 10% to 90%	1 V Range, Cload=15 pF	na	tbc	tbc	V/usec
SR1N	Slew rate, negative 10% to 90%	1 V Range, Cload=15 pF	na	tbc	tbc	V/usec
SR4P	Slew rate, positive 10% to 90	4 V Range, Cload=15 pF	na	tbc	tbc	V/usec
SR4N	Slew rate, negative 10% to 90%	4 V Range, Cload=15 pF	na	tbc	tbc	V/usec

Parameter	Description	Conditions	Min	Typ	Max	Units
VglP4	Glitch voltage peak amplitude, positive	4 V Range, Transition 0x7f to 0x80	na	tbc	tbc	mV
VglN4	Glitch voltage peak amplitude, negative	4 V Range, Transition 0x7f to 0x80	na	tbc	tbc	mV
Vn1	Voltage noise, 1 V Range	f=10 kHz, Fast mode, output at 1/2 scale, Cload = 15 pF	na	tbc	na	nV/rtHz
Vn4	Voltage noise, 4V Range	f=10 kHz, Fast mode, output at 1/2 scale, Cload=15 pF	na	tbc	na	nV/rtHz

Figures

Full scale step response, Slow mode, 1.0V X axis Time, usec Y axis Volts 1 0x00 to 0xff 2 0xff to 0x00	10-90% Response Time, Slow mode, 4.0V X axis Vdda Time usec Y axis Volts 1 0x00 to 0xff 2 0xff to 0x00
Full scale step response, Fast mode, 1.0V X axis Time, usec Y axis Volts 1 0x00 to 0xff 2 0xff to 0x00	10-90% Response Time, Fast mode, 4.0V X axis Time, usec Y axis Volts 1 0x00 to 0xff 2 0xff to 0x00
Digital to Analog Glitch, Slow mode, 1.0V X axis Time usec Y axis mV 1 0x7f to 0x80 2 0x80 to 0x7f	Digital to Analog Glitch, Slow mode, 4.0V X axis Time usec Y axis mV 1 0x7f to 0x80 2 0x80 to 0x7f
Digital to Analog Glitch, Fast mode, 1.0V X axis Time usec Y axis mV 1 0x7f to 0x80 2 0x80 to 0x7f	Digital to Analog Glitch, Fast mode, 4.0V X axis Time usec Y axis mV 1 0x7f to 0x80 2 0x80 to 0x7f
Voltage noise, Fast mode, 1.0V Xaxis freq kHz .01 to 1000 kHz Yaxis voltage noise nV/rtHz	Voltage noise, Fast mode, 4.0V Range Xaxis freq kHz .01 to 1000 kHz Yaxis voltage noise nV/rtHz

	PSRR vs freq, Vdda = 5.0V, 4 V scale X axis freq 100 Hz to 1.0 MHz Y axis dB

Terminology

Integral Nonlinearity (INL)

INL, integral nonlinearity, is a measure of the maximum deviation, in LSBs, from a best fit straight line over the operating range of the DAC.

Differential Nonlinearity (DNL)

DNL, differential nonlinearity, is the difference between the measured change and the ideal 1 LSB change between any two adjacent codes. This VDAC is guaranteed monotonic by design. The output is "thermometer-encoded," each successive step is made by turning on a separate output source which is summed with previously enabled output sources.

Montonicity

A DAC is defined as monotonic if the output increases or stays the same wih eaching increasing digital code input value. The VDAC8 component is monotonic over the full operating range of voltage and temperature.

Zero-scale Error

Zero-scale error is the difference between the measured value at code 0x00 and the value of the best-fit straight line at code 0x00.

Full Scale Gain Error

Full scale gain error is the measure of the difference between the measured value and the nominal value at maximum code. The maximum value is either 1.020 V or 4.080 V at code = 255 (0x00).

Full Scale Gain Temperature Coefficient (TC)

Full scale gain temperature coefficient is a measure of the change in full scale value (maximum code oxff) with change in temperature. Gain changes at lower values are proportional to code value.

Power Supply Rejection Ratio (PSRR)

Power supply rejection ratio measures the isolation of the VDAC's output from the power supply.

Settling Time

Settling time is the amount of time required for the output to settle to a specific level for a specific digital input change.

Slew Rate

The slew rate is the maximum rate of change of the output of the VDAC. Slew rate is measured from 10% to 90% of full scale value

Glitch Amplitude

Glitch ampitude is the peak amplitude of the pulse injected into the output when the input code changes a single count at mid-scale (0x7f to 0x80). The pulse is in excess of the difference between the static values before and after data change.

Voltage Noise

Voltage noise is the sum of the noise of the VDAC's output resistance and the current output noise times the output resistance of the VDAC. This noise will vary as a function of code value.

Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes	Reason for Changes / Impact
1.50.a	Minor datasheet edits.	
1.50.a	Minor datasheet edits.	
1.50	Added Sleep/Wakeup and Init/Enable APIs.	To support low power modes, as well as to provide common interfaces to separate control of initialization and enabling of most components.
	Added DMA capabilities file to the component.	This file allows the VDAC8 to be supported by the DMA Wizard tool in PSoC Creator.

© Cypress Semiconductor Corporation, 2009-2016. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage (“Unintended Uses”). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

PRELIMINARY