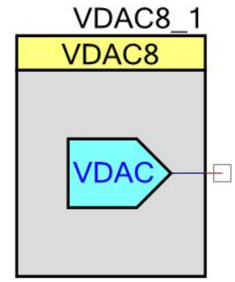
8-Bit Voltage Digital to Analog Converter (VDAC8)

1.90



Features

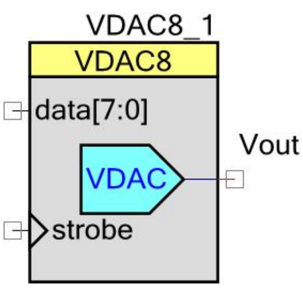
- Voltage output ranges: 1.020-V and 4.080-V full scale
- Software- or clock-driven output strobe
- Data source can be CPU, DMA, or Digital components

General Description

The VDAC8 component is an 8-bit voltage output Digital to Analog Converter (DAC). The output range can be from 0 to 1.020 V (4 mV/bit) or from 0 to 4.08 V (16 mV/bit). The VDAC8 can 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.

Note The VDAC8, when driven to a pin, cannot drive a value that exceeds the VDDIO for that pin. To get the result you want, set the correct VDDIO supply.

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 Digital 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 VDAC8_SetValue() function or directly write a value to the VDAC8 data register.

strobe – Input *

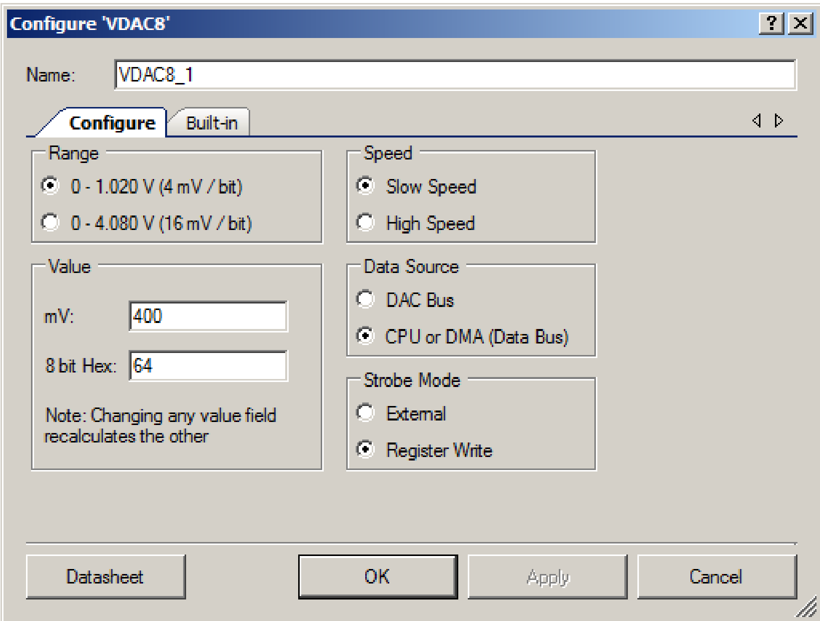
The strobe input is an optional signal input and is selected with the **StrobeMode** parameter.

- If **StrobeMode** is set to **External**, the strobe pin is visible and must be connected to a valid digital source. In this mode, the VDAC8 data register is sampled on the rising edge of the strobe signal, and then transferred to the DAC output on the next falling edge of the strobe signal. The DAC output starts to slew and settle from the falling edge of the strobe signal.
- If **StrobeMode** is set to **Register Write**, the pin disappears from the symbol and any write to the data registers is immediately transferred to the DAC.

For audio or periodic sampling applications, the same clock used to clock the data into the DAC can also be used to generate an interrupt. In this case, 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.

Component Parameters

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



The VDAC8 component provides the following parameters.

Range

This parameter allows to set one of the two voltage ranges as the default value. The range may be changed at any time during runtime with the VDAC8_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-V range: V_{OUT} = (value/256) × 1.024 V
- 4-V range: V_{OUT} = (value/256) × 4.096 V

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

Value

This is the initial value the VDAC8 presents after the VDAC8_Start() command is executed. The VDAC8_SetValue() function or a direct write to the DAC register will override the default value at anytime. Legal values are between 0 and FF, inclusive. The **mV** field represents VDAC8 output voltage in millivolts and the **8 bit Hex** field represents VDAC8 input data value in Hex.

Speed

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

Data Source

This parameter selects the source of the data to be written into the DAC register. Selecting **CPU or DMA (Data Bus) option will select** the CPU (firmware) or the DMA to write data to the VDAC8. Selecting **DAC Bus** option will select data to be written directly from the Digital components or control registers.

When **DAC Bus** is selected, the input is indicated on the VDAC8 symbol. There is only one DAC Bus, so multiple VDAC8s cannot have independent hardware data sources.

When **Data Source** is set as **DAC Bus**, the customizer automatically sets the **Strobe Mode** to **External** and disables the option so that it cannot be changed.

Note In the DAC Bus mode, the output from the DAC is lost during sleep and requires a new value to be strobed from the DAC bus to generate output values again.

Strobe Mode

This parameter selects whether the data is immediately written to the DAC when the data is written into the VDAC8 data register. This mode is enabled when the **Register Write** option is selected. When the **External** option is selected, a clock or signal from the Digital components or control register controls when the data is written from the DAC register to the actual DAC.

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. It can be renamed 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
VDAC8_Start()	Initializes the VDAC8 with default customizer values.
VDAC8_Stop()	Disables the VDAC8 and sets it to the lowest power state.
VDAC8_SetSpeed()	Sets DAC speed.
VDAC8_SetValue()	Sets value between 0 and 255 with the given range.

VDAC8_SetRange()	Sets range to 1 or 4 volts.
VDAC8_Sleep()	Stops and saves the user configuration.
VDAC8_WakeUp()	Restores and enables the user configuration.
VDAC8_Init()	Initializes or restores default VDAC8 configuration
VDAC8_Enable()	Enables the VDAC8.
VDAC8_SaveConfig()	Saves nonretention DAC data register value.
VDAC8_RestoreConfig()	Restores nonretention DAC data register value

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 VDAC8_Start() is called. This allows the component to restart without reinitialization after the first call to the VDAC8_Start() routine. If reinitialization of the component is required, then the VDAC8_Init() function can be called before the VDAC8_Start() or VDAC8_Enable() function.

void VDAC8_Start(void)

Description: This is the preferred method to begin component operation. VDAC8_Start() sets the initVar variable, calls the VDAC8_Init() function, calls the VDAC8_Enable() function, and powers up the VDAC8 to the given power level. A power level of 0 is the same as executing the VDAC_Stop() function.

Parameters: None

Return Value: None

Side Effects: If the initVar variable is already set, this function only calls the VDAC8_Enable() function.

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 the following table 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 the following table for ranges.

Option	Description
VDAC8_RANGE_1V	Sets 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 selected with the VDAC8_SetRange() API.

Return Value: None

Side Effects: On PSoC 3 and PSoC 5LP silicon, the VDAC8_SetValue() function should be called after enabling power to the VDAC.

void VDAC8_Sleep(void)

Description: This is the preferred API to prepare the component for sleep. The VDAC8_Sleep() API saves the current component state. Then it calls the VDAC8_Stop() function and calls VDAC8_SaveConfig() to save the hardware configuration.

Call the VDAC8_Sleep() function before calling the CyPmSleep() or the CyPmHibernate() function. Refer to the PSoC Creator *System Reference Guide* for more information about power management functions.

Parameters: None

Return Value: None

Side Effects: None

void VDAC8_Wakeup(void)

Description: This is the preferred API to restore the component to the state when VDAC8_Sleep() was called. The VDAC8_Wakeup() function calls the VDAC8_RestoreConfig() function to restore the configuration. If the component was enabled before the VDAC8_Sleep() function was called, the VDAC8_Wakeup() function will also re-enable the component.

Parameters: None

Return Value: None

Side Effects: Calling the VDAC8_Wakeup() function without first calling the VDAC8_Sleep() or VDAC8_SaveConfig() function may produce unexpected behavior.

void VDAC8_Init(void)

Description: Initializes or restores the component according to the customizer Configure dialog settings. It is not necessary to call VDAC8_Init() because the VDAC8_Start() API calls this function and is the preferred method to begin component operation.

Parameters: None

Return Value: None

Side Effects: All registers will be set to their initial values. This will reinitialize the component. Calling the VDAC8_Init() function requires a call to VDAC8_SetValue() if you intend to set a new value other than what is currently in the register.

void VDAC8_Enable(void)

Description: Activates the hardware and begins component operation. It is not necessary to call VDAC8_Enable() because the VDAC8_Start() API calls this function, which is the preferred method to begin component operation.

Parameters: None

Return Value: None

Side Effects: None

void VDAC8_SaveConfig(void)

Description: This function saves the component configuration and nonretention registers. This function will also save the current component parameter values, as defined in the Configure dialog or as modified by appropriate APIs. This function is called by the VDAC8_Sleep() function.

Note In the DAC Bus mode, the values are not saved.

Parameters: None

Return Value: None

Side Effects: None

void VDAC8_RestoreConfig(void)

Description: This function restores the component configuration and nonretention registers. This function will also restore the component parameter values to what they were before calling the VDAC8_Sleep() function.

Note In the DAC Bus mode, the values are not restored.

Parameters: None

Return Value: None

Side Effects: Calling this function before calling VDAC_Sleep() may result in unexpected behavior.

DMA Wizard

VDAC8 components do not require implementation of a DMA Request signal. The typical usage is signal generation. The data rate to VDAC8 components should be controlled externally. You can use the DMA Wizard 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 and 255

MISRA Compliance

This section describes the MISRA-C:2004 compliance and deviations for the component. There are two types of deviations defined:

- project deviations – deviations that are applicable for all PSoC Creator components
- specific deviations – deviations that are applicable only for this component

This section provides information on component-specific deviations. Project deviations are described in the MISRA Compliance section of the *System Reference Guide* along with information on the MISRA compliance verification environment.

The VDAC8 component does not have any specific deviations.

Sample Firmware Source Code

PSoC Creator provides many example projects that include schematics and example code in the Find Example Project dialog. For component-specific examples, open the dialog from the Component Catalog or an instance of the component in a schematic. For general examples, open the dialog from the Start Page or **File** menu. As needed, use the **Filter Options** in the dialog to narrow the list of projects available to select.

Refer to the “Find Example Project” topic in the PSoC Creator Help for more information.

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 VDAC8, the output is an 8-bit digital-to-analog conversion voltage to support applications that need reference voltages. In this case, 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

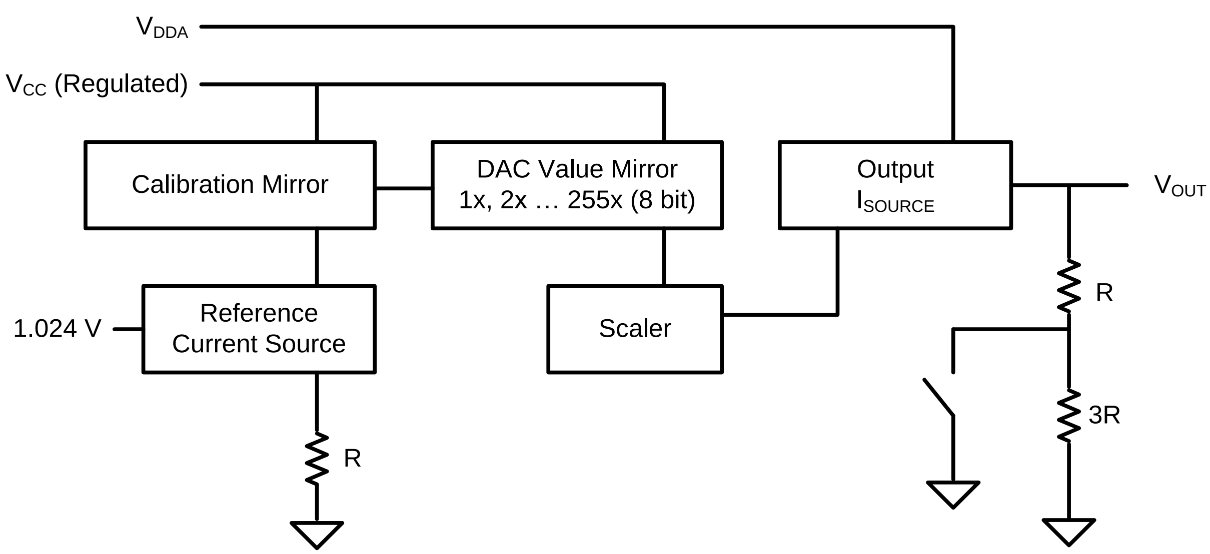
Both output ranges have 255 equal steps. The VDAC8 is implemented by driving the output of the current DAC through resistors and obtaining a voltage output. Because no buffer is used, any DC current drawn from the DAC affects the output level. Therefore, in this mode any load connected to the output should be capacitive.

The VDAC8 can convert up to 1 Msps. Also, the DAC is slower in 4-V mode than 1-V mode, because the resistive load to V_{SSA} is four times larger. In 4-V mode, the VDAC8 can convert up to 250 ksps.

Block Diagram and Configuration

Figure 1 shows the block diagram for the VDAC8 component.

Figure 1. VDAC8 Block Diagram



Registers

The functions provided with the component support most of the common runtime functions that most applications require. 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 an API. This can be useful for either the CPU or DMA.

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: Sets data speed

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: Selects 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

VDAC8_DATA

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

- Data[7:0]: DAC data register

Resources

The VDAC8 component uses one viDAC8 analog block.

API Memory Usage

The component memory usage varies significantly, depending on the compiler, device, number of APIs used and component configuration. The following table provides the memory usage for all APIs available in the given component configuration.

The measurements have been done with the associated compiler configured in Release mode with optimization set for Size. For a specific design the map file generated by the compiler can be analyzed to determine the memory usage.

Configuration	PSoC 3 (Keil_PK51)		PSoC 5LP (GCC)	
	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes
Default	237	3	348	5

DC and AC Electrical Characteristics for PSoC 3

Specifications are valid for –40 °C ≤ T_A ≤ 85 °C and T_J ≤ 100 °C except where noted. Specifications are valid for 1.71 V to 5.5 V, except where noted. Typical values are for T_A = 25 °C.

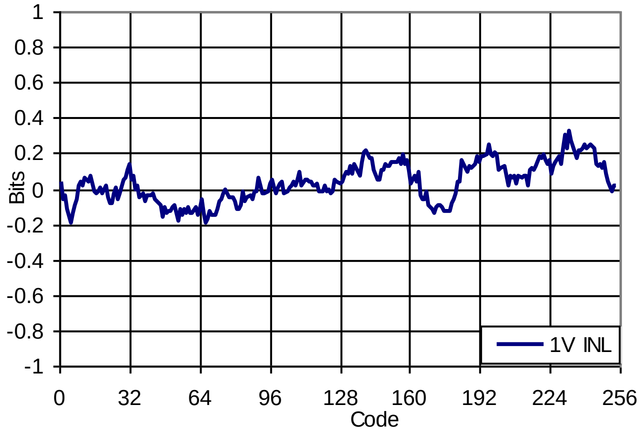
DC Characteristics

Parameter	Description	Conditions	Min	Typ	Max	Units
	Resolution		–	8	–	bits
INL1	Integral nonlinearity	1-V scale	–	±2.1	±2.5	LSB
DNL1	Differential nonlinearity	1-V scale	–	±0.3	±1	LSB
R_{OUT}	Output resistance	1-V scale	–	4	–	kΩ
		4-V scale	–	16	–	kΩ
V_{OUT}	Output voltage range, code = 255	1-V scale	–	1.02	–	V
		4-V scale, V_{DDA} = 5 V	–	4.08[1]	–	V
	Monotonicity		–	–	Yes	–
V_{OS}	Zero-scale error		–	0	±0.9	LSB
Eg	Gain error	1-V scale	–	±1.6	±2.5	%
		4-V scale	–	±1.5	±2.5	%
TC_Eg	Temperature coefficient, gain error	1-V scale	–	–	0.03	%FSR/°C
		4-V scale	–	–	0.03	%FSR/°C
I_{DD}	Operating current	Slow mode	–	–	100	µA
		Fast mode	–	–	500	µA

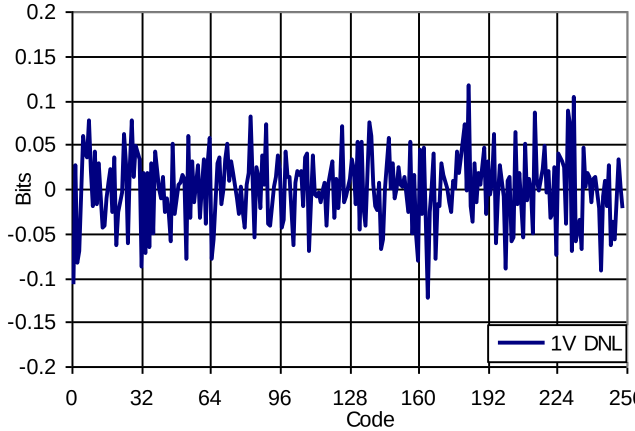
[1] For V_{DDA} voltage below 5 V, the output only complies to specifications for output voltages below (V_{DDA} – 1 V).

Figures

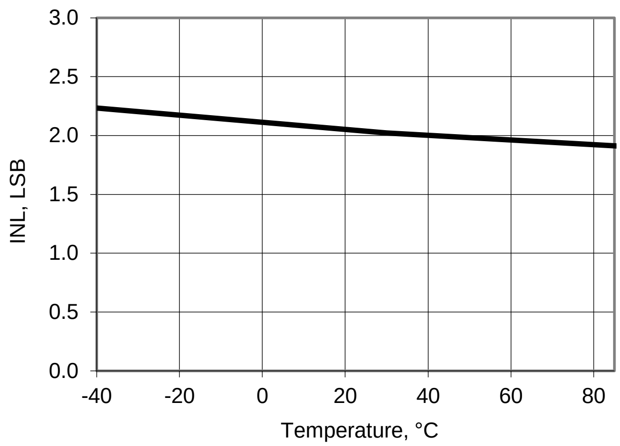
INL versus Input Code, 1.0-V Range



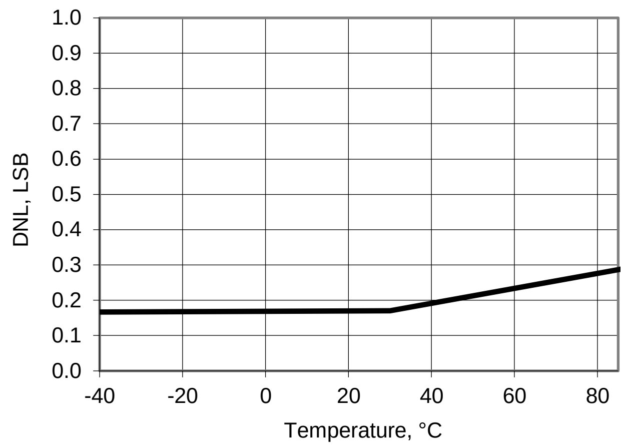
Typical DNL versus Input Code, 1.0-V Range



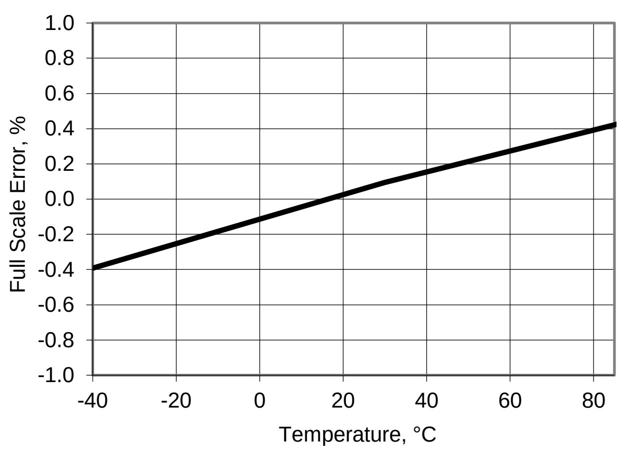
INL versus Temperature, 1-V Mode



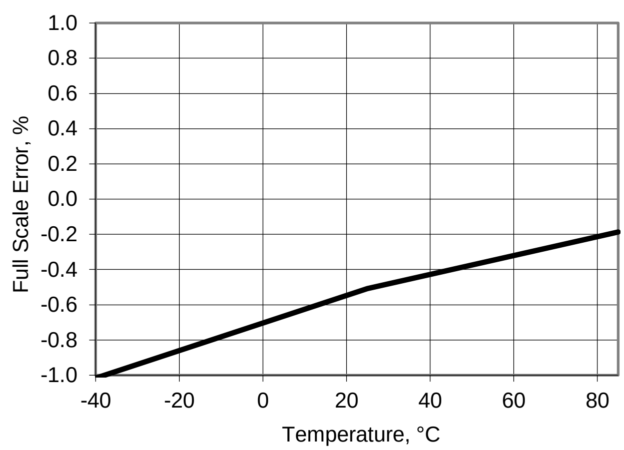
DNL versus Temperature, 1-V Mode



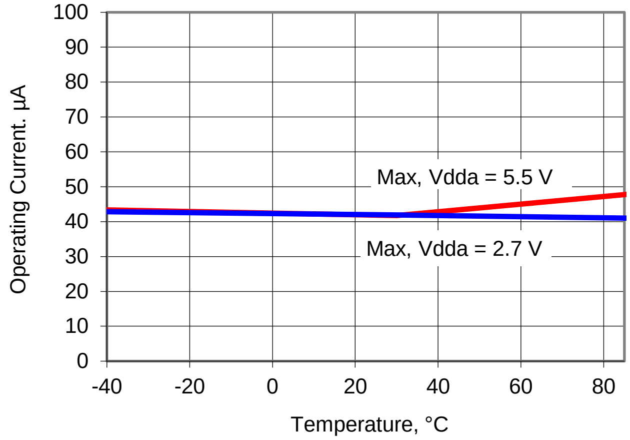
Full Scale Error versus Temperature, 1-V Mode



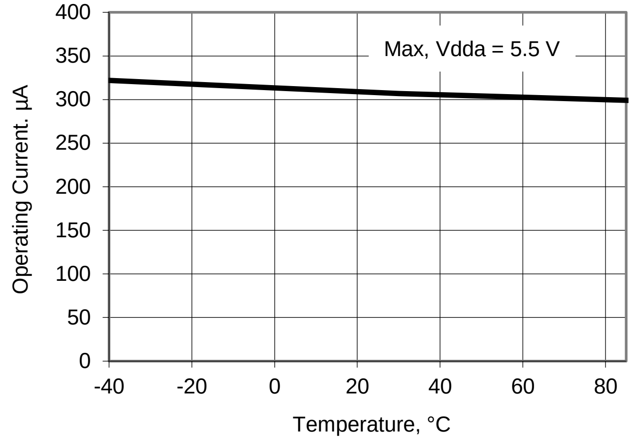
Full Scale Error versus Temperature, 4-V Mode



Operating Current versus Temperature, 1-V Mode, Slow Mode



Operating Current versus Temperature, 1-V Mode, Fast Mode

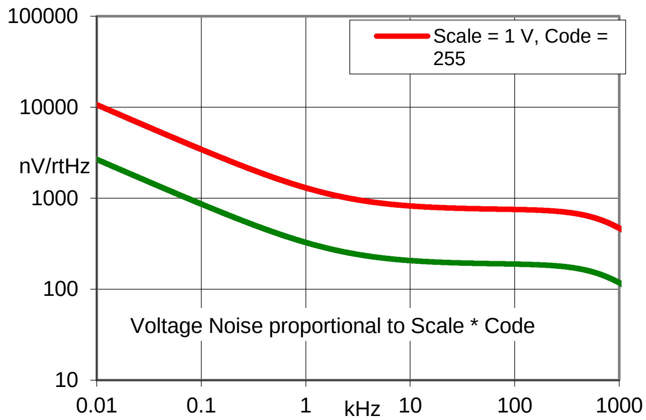


AC Characteristics

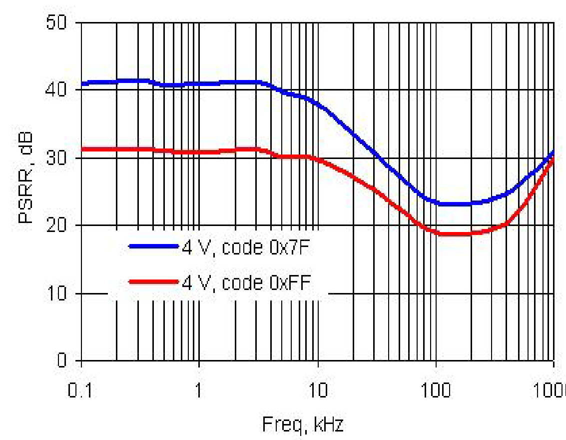
Parameter	Description	Conditions	Min	Typ	Max	Units
F_{DAC}	Update rate	1-V scale	–	–	1000	ksps
		4-V scale	–	–	250	ksps
TsettleP	Settling time to 0.1%, step 25% to 75%	1-V scale, C_{LOAD} = 15 pF	–	0.45	1	µs
		4-V scale, C_{LOAD} = 15 pF	–	0.8	3.2	µs
TsettleN	Settling time to 0.1%, step 75% to 25%	1-V scale, C_{LOAD} = 15 pF	–	0.45	1	µs
		4-V scale, C_{LOAD} = 15 pF	–	0.7	3	µs
Vn1V	Voltage noise [2]	Range = 1 V, fast mode, Vdda = 5 V, 10 kHz	–	750	–	nV/sqrtHz

Figures

Noise Levels, Scale 1 V

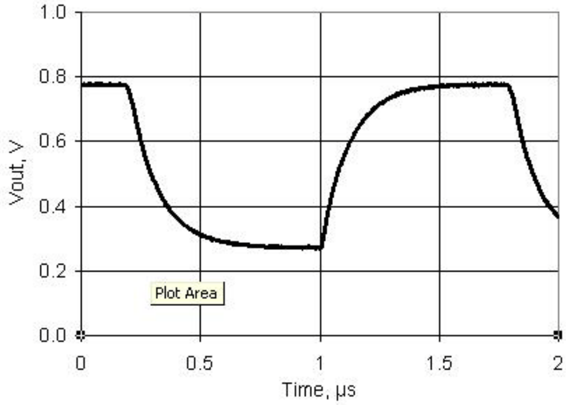


PSRR vs Frequency

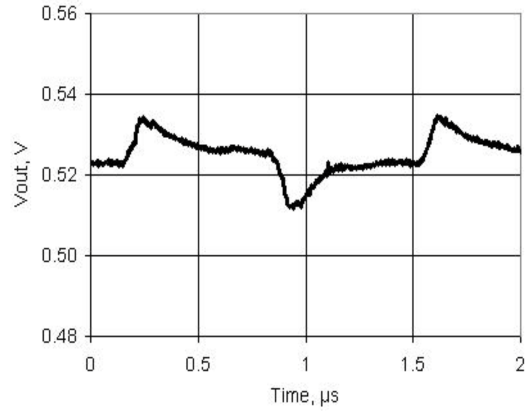


[2] Output noise is directly proportional to code value.

Step Response, Codes 0x40 - 0xC0, 1 V Mode, Fast Mode, Vdda = 5 V



Glitch Response, Codes 0x7F - 0x80, 1V Mode, Fast Mode, Vdda = 5 V



DC and AC Electrical Characteristics for PSoC 5LP

Specifications are valid for –40° C ≤ T_A ≤ 85° C and T_J ≤ 100° C, except where noted. Specifications are valid for 2.7 V to 5.5 V, except where noted. Typical values are for T_A = 25° C.

DC Characteristics

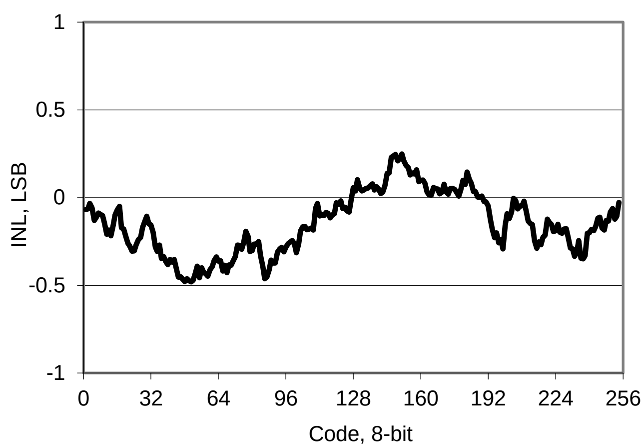
Parameter	Description	Conditions	Min	Typ	Max	Units
	Resolution		–	8	–	bits
INL1	Integral nonlinearity	1-V scale	–	±2.1	±2.5	LSB
DNL1	Differential nonlinearity	1-V scale	–	±0.3	±1	LSB
R_{OUT}	Output resistance	1-V scale	–	4	–	kΩ
		4-V scale	–	16	–	kΩ
V_{OUT}	Output voltage range, code = 255	1-V scale	–	1.02	–	V
		4-V scale, V_{DDA} = 5 V	–	4.08[3]	–	V
	Monotonicity		–	–	Yes	–
V_{OS}	Zero-scale error		–	0	±0.9	LSB
Eg	Gain error	1-V scale	–	–	±5	%
		4-V scale	–	–	±5	%
TC_Eg	Temperature coefficient, gain error	1-V scale	–	–	0.03	%FSR/°C
		4-V scale	–	–	0.03	%FSR/°C

[3] For V_{DDA} voltage below 5 V, the output only complies to specifications for output voltages below (V_{DDA} – 1 V).

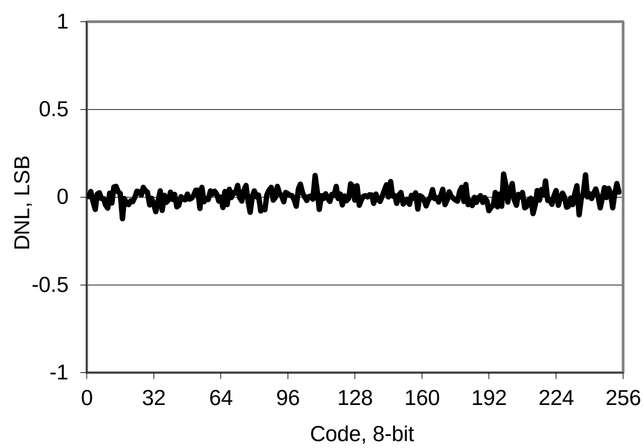
Parameter	Description	Conditions	Min	Typ	Max	Units
I_{DD}	Operating current	4-V Slow mode	–	–	100	µA
		4-V Fast mode	–	–	500	µA
		1-V Slow mode			300	µA
		1-V Fast mode			600	µA

Figures

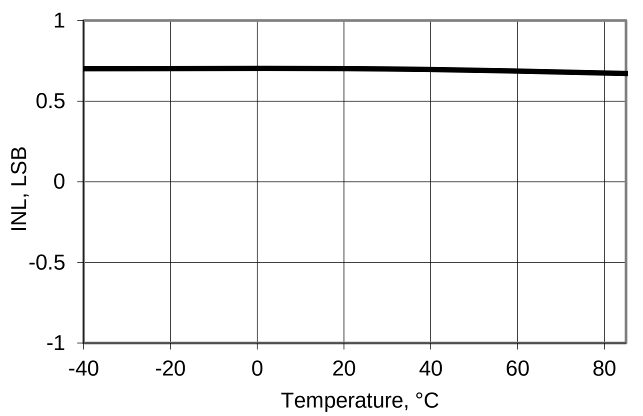
INL versus Input Code, 1-V Mode



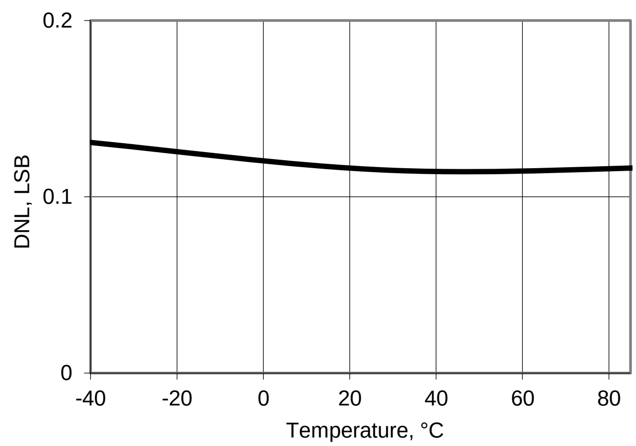
DNL versus Input Code, 1-V Mode



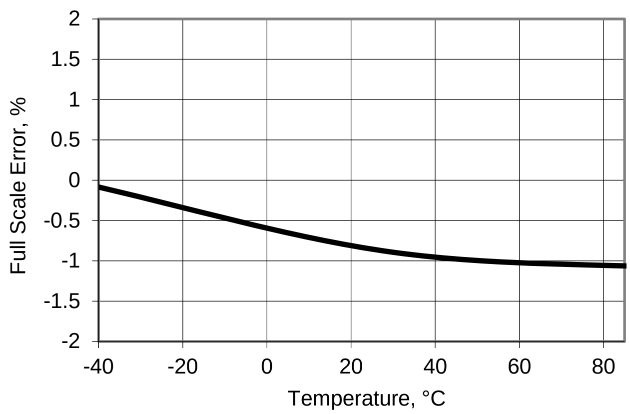
INL versus Temperature, 1-V Mode



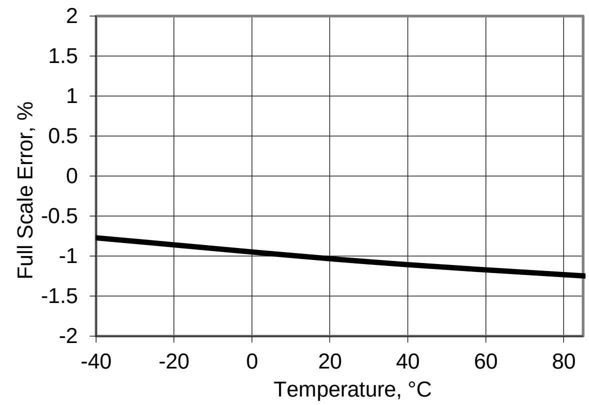
DNL versus Temperature, 1-V Mode



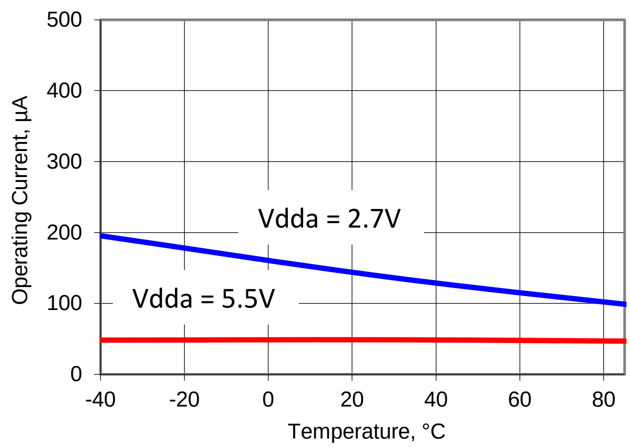
Full Scale Error versus Temperature, 1-V Mode



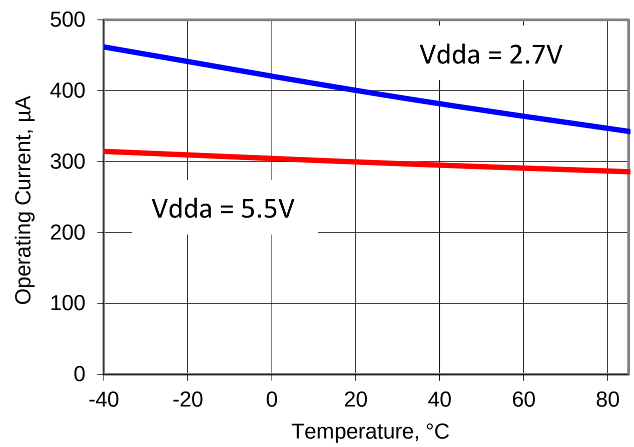
Full Scale Error versus Temperature, 4-V Mode



Operating Current versus Temperature, 1-V Mode, Slow Mode



Operating Current versus Temperature, 1-V Mode, Fast Mode

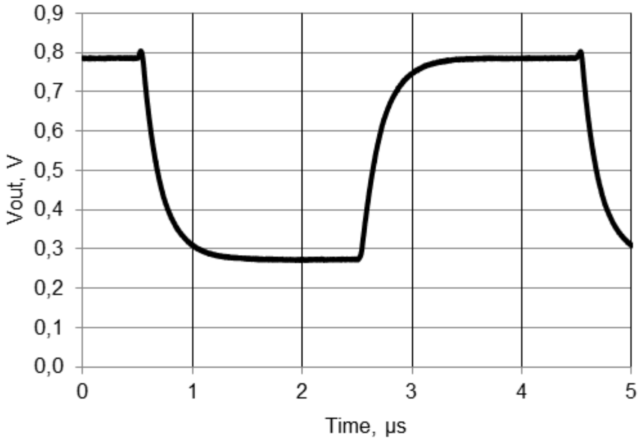


AC Characteristics

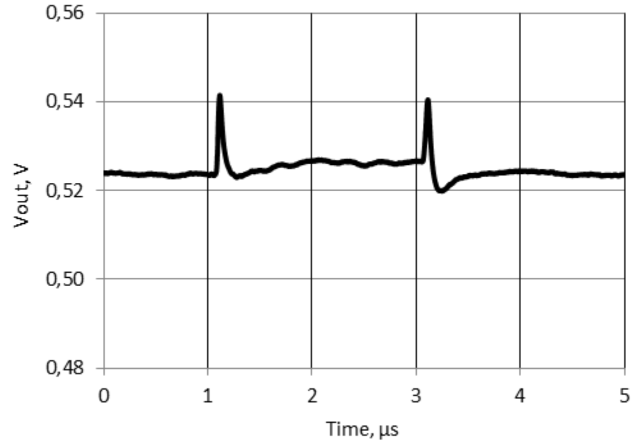
Parameter	Description	Conditions	Min	Typ	Max	Units
F_{DAC}	Update rate	1-V scale	–	–	1000	ksps
		4-V scale	–	–	250	ksps
TsettleP	Settling time to 0.1%, step 25% to 75%	1-V scale, C_{LOAD} = 15 pF	–	0.45	1	µs
		4-V scale, C_{LOAD} = 15 pF	–	0.8	4	µs
TsettleN	Settling time to 0.1%, step 75% to 25%	1-V scale, C_{LOAD} = 15 pF	–	0.45	1	µs
		4-V scale, C_{LOAD} = 15 pF	–	0.7	4	µs
	Voltage Noise	Range = 1 V, fast mode, V_{DDA} = 5 V, 10 kHz	–	750	–	nV/sqrtHz

Figures

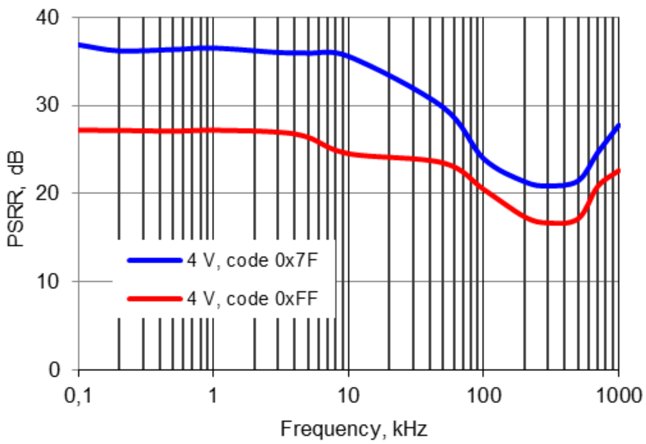
Step Response, Codes 0x40 to 0xC0, 1-V Mode, Fast Mode, VDDA = 5 V



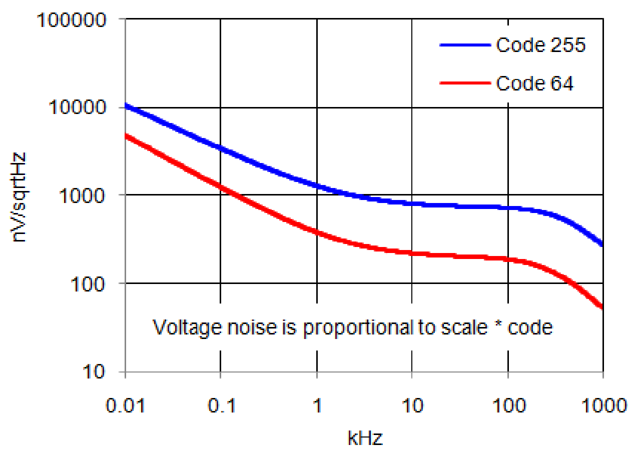
Glitch Response, Codes 0x7F to 0x80, 1-V Mode, Fast Mode, VDDA = 5 V



PSRR versus Frequency



Voltage Noise, 1 V Mode, Fast Mode, VDDA = 5 V



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 VDAC8 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 with each 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 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 the change in full-scale value (maximum code 0xFF) 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 VDAC8'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 VDAC8. Slew rate is measured from 10 percent to 90 percent of full-scale value

Glitch Amplitude

Glitch amplitude 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 greater than the difference between the static values before and after data change.

Voltage Noise

Voltage noise is the sum of the noise of the VDAC8's output resistance and the current output noise times the output resistance of the VDAC8. This noise varies 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
1.90.d	Minor datasheet edits.	
1.90.c	Minor datasheet edits.	
1.90.b	Updated the Strobe Input behavior description.	Make the datasheet more clear.
1.90.a	Changed the layout on the Configure dialog.	Optimized the layout to remove the scroll bars.
1.90	Added MISRA Compliance section.	The component does not have any specific deviations.
1.80	Added PSoC 5LP support	
	CYREENTRANT keyword added to all APIs.	Not all APIs are truly reentrant. Comments in the component API source files indicate which functions are candidates. This change is required to eliminate compiler warnings for functions that are not reentrant used in a safe way: protected from concurrent calls by flags or Critical Sections.
	Updated DC and AC Electrical characteristics, Resource and API memory usage section.	
1.70.a	Added DC and AC Electrical characteristics data for PSoC 5	
	Minor datasheet edits and updates	
1.70	VDAC8_Stop() API modified for PSoC 5	Change required to prevent the component from impacting unrelated analog signals when stopped, when using PSoC 5.
	Updated VDAC customizer.	▪ To make VDAC layout same as IDAC layout. ▪ To force the Strobe mode to External when Data Source is selected as DAC Bus.
1.60	Added a GUI Configuration Editor	Previous configuration window did not provide enough information for ease of use.
	Added characterization data to datasheet	
	Minor datasheet edits and updates	
1.50	Added Sleep/Wakeup and Init/Enable APIs.	To support low-power modes, and 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.