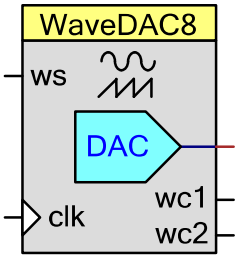
WaveDAC8 (8-Bit Waveform Generator)

1.60

Features

- Supports standard and arbitrary waveform generation
- Output may be voltage or current, sink or source
- Hardware selection between two waveforms
- Waveforms may be up to 4000 points
- Predefined sine, triangle, square, and sawtooth waveforms.
- May changed waveform arrays during runtime



General Description

The WaveDAC8 component provides a simple and fast solution for automatic periodic waveform generation. A high level interface allows the user to select a predefined waveform or a custom arbitrary waveform. Two separate waveforms can be defined then selected with an external pin to create a modulated output. The input clock can also be used to change the sample rate, or modulate the output.

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.

ws – Input

This input selects which waveform will be generated. It can be used to switch quickly between two waveforms to generate an FSK signal.

clk – Input*

The clock input allows the user to use an alternate clock source. When internal clock is slected, this input is not visible.

wc1 – Output

This signal goes high for one bus clock at the end of the waveform 1.

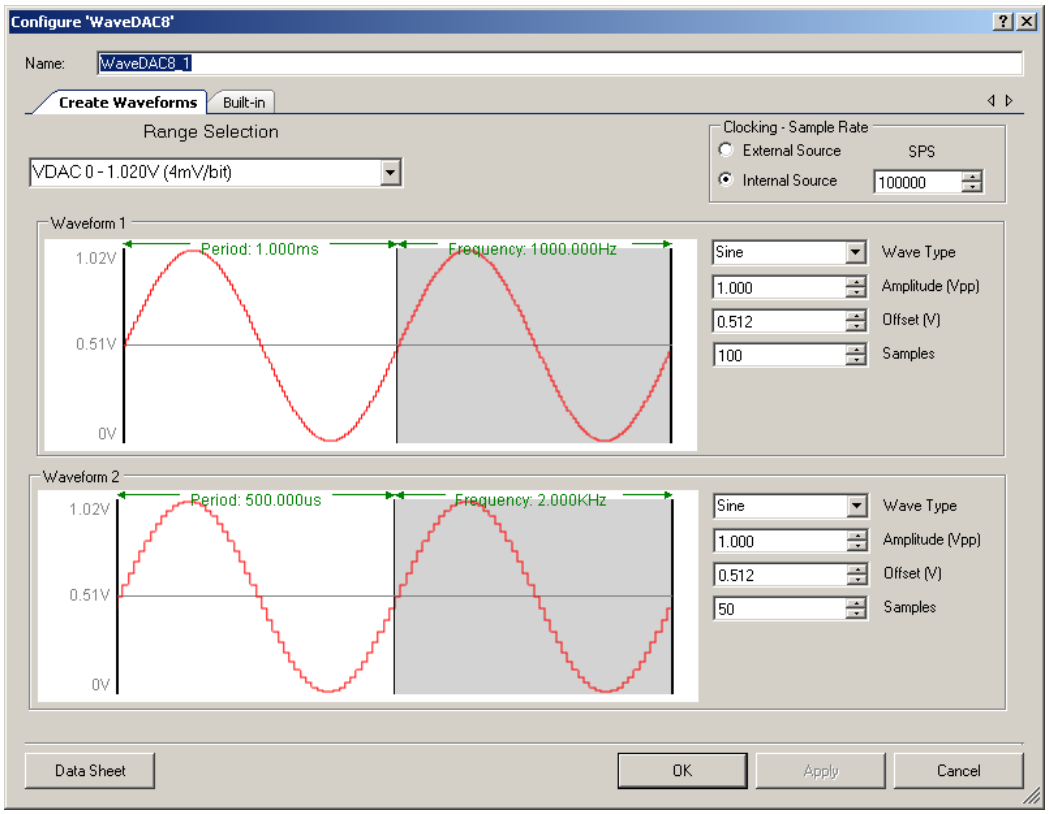
wc2 – Output

This signal goes high for one bus clock at the end of the waveform 2.

Parameters and Setup

Drag a WaveDAC8 component onto your design and double-click it to open the Configure dialog.

Figure 1 Configure WaveDAC8 Dialog



The WaveDAC8 component provides the following parameters.

Range Selection

This parameter selects the output mode and range of the internal DAC.

Range	Mode	Ouput	Step Size
VDAC 0 – 1.020V (4mV/bit)	Voltage	0 to 1.020 Volts	4 mV
VDAC 0 – 4.080 (16mV/bit)	Voltage	0 to 4.080 volts	16 mV
ISink 0 – 2.040 mA	Current Sink	0 to 2.040 mA	8 uA
ISink 0 – 255 uA	Current Sink	0 to 255 uA	1 uA
ISink 0 – 32 uA	Current Sink	0 to 32 mA	0.125 uA
ISource 0 – 2.040 mA	Current Source	0 to 2.040 mA	8 uA
ISource 0 – 255 uA	Current Source	0 to 255 uA	1 uA
ISource 0 – 32 uA	Current Source	0 to 32 mA	0.125 uA

Note: Both waveforms share the range.

SPS (Clocking – Sample Rate)

This parameter selects the sampling frequency rate in Hz. The maximum sample rate is 1 MHz. The frequency of the selected waveform will be the number of samples divided by the sample rate. Both waveforms will share the same sample rate.

$$Waveform\_Period = \frac{Samples}{SPS} \qquad Waveform\_Frequency = \frac{SPS}{Samples}$$

Clock Source (Internal / External)

These radio buttons select whether the clock source is external or external. When an internal clock is selected, the clock pin will not be visible.

Wave Type

This parameter selects one of five waveforms, four are fixed and one allows the user to input his own waveform.

- Sine
- Square
- Triangle
- Sawtooth
- Arbitrary

Resources

The WaveDAC8 component uses either a VDAC8 or IDAC8, two DMA channels, and an optional clock.

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 "WaveDAC8_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 "WaveDAC8".

Function	Description
void WaveDAC8_Start(void)	Starts the DAC and DMA channels
void WaveDAC8_Stop(void)	Disables DAC and DMA channels
void WaveDAC8_Wave1Setup(uint8 * WavePtr, uint16 SampleSize)	Sets the array and size of array used for waveform generation for waveform 1.
void WaveDAC8_Wave2Setup(uint8 * WavePtr, uint16 SampleSize)	Sets the array and size of array used for waveform generation for waveform 2.
void WaveDAC8_SetSpeed(uint8 speed)	Set drive speed of DAC.
void WaveDAC8_Sleep(void)	Stops and saves the user configuration.
void WaveDAC8_Wakeup(void)	Restores and enables the user configuration.
void WaveDAC8_SaveConfig(void)	Empty function. Provided for future usage.
void WaveDAC8_RestoreConfig(void)	Empty function. Provided for future usage.

void WaveDAC8_Start(void)

Description: Performs all of the required initialization for the component and enables power to the block. The first time the routine is executed, the input and feedback resistance values are configured for the operating mode selected in the design. When called to restart the WaveDAC8 following a WaveDAC8_Stop() call, the current component parameter settings are retained.

Parameters: None

Return Value: None

Side Effects: None

void WaveDAC8_Stop(void)

Description: Turn off the WaveDAC8 block.

Parameters: None

Return Value: None

Side Effects: Does not affect WaveDAC8 type or power settings

void WaveDAC8_Wave1Setup(uint8 * WavePtr, uint16 SampleSize)

Description: Select new waveform array for waveform 1 output. The WaveDAC8_Stop function should be called prior to calling this function and WaveDAC8_Start should be called to restart waveform.

Parameters: Uint8 * WavePtr: Pointer to array containing waveform data
Uint16 SampleSize: Size of waveform array pointed to by WavePtr. (Maximum sample size is 4096)

Return Value: None

Side Effects: Does not affect WaveDAC8 type or power settings

void WaveDAC8_Wave2Setup(uint8 * WavePtr, uint16 SampleSize)

Description: Select new waveform array for waveform 2 output. The WaveDAC8_Stop function should be called prior to calling this function and WaveDAC8_Start should be called to restart waveform.

Parameters: Uint8 * WavePtr: Pointer to array containing waveform data
Uint16 SampleSize: Size of waveform array pointed to by WavePtr. (Maximum sample size is 4096)

Return Value: None

Side Effects: Does not affect WaveDAC8 type or power settings

void WaveDAC8_Init(void)

Description: Initializes or restores the component according to the customizer Configure dialog settings. It is not necessary to call WaveDAC8_Init() because the WaveDAC8_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 values according to the customizer Configure dialog.

void WaveDAC8_Enable(void)

Description: Activates the hardware and begins component operation. It is not necessary to call WaveDAC8_Enable() because the WaveDAC8_Start() API calls this function, which is the preferred method to begin component operation.

Parameters: None

Return Value: None

Side Effects: None

void WaveDAC8_SetSpeed(uint8 power)

Description: Sets the drive power to one of four settings;

Parameters: (uint8) power: See the following table for valid power settings.

Power Setting	Notes
WaveDAC8_LOWSPEED	Lowest active power and slowest slew rate.
WaveDAC8_HIGHSPEED	Highest power and fastest slew rate.

Return Value: None

Side Effects: None

void WaveDAC8_Sleep(void)

Description: This is the preferred API to prepare the component for sleep. The WaveDAC8_Sleep() API saves the current component state. Then it calls the WaveDAC8_Stop() function and calls WaveDAC8_SaveConfig() to save the hardware configuration.
Call the WaveDAC8_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 WaveDAC8_Wakeup(void)

Description: This is the preferred API to restore the component to the state when WaveDAC8_Sleep() was called. The WaveDAC8_Wakeup() function calls the WaveDAC8_RestoreConfig() function to restore the configuration. If the component was enabled before the WaveDAC8_Sleep() function was called, the WaveDAC8_Wakeup() function will also re-enable the component.

Parameters: None

Return Value: None

Side Effects: Calling the WaveDAC8_Wakeup() function without first calling the WaveDAC8_Sleep() or WaveDAC8_SaveConfig() function may produce unexpected behavior.

void WaveDAC8_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 WaveDAC8_Sleep() function.

Parameters: None

Return Value: None

Side Effects: None

void WaveDAC8_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 WaveDAC8_Sleep() function.

Parameters: None

Return Value: None

Side Effects: None

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 "WaveDAC8_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>

uint8 otherWave1[8] = {0, 32, 64, 96, 128, 160, 192, 224 };
uint8 otherWave2[4] = {0, 64, 128, 255 };

extern uint8 WaveDAC8_1_wave1[];
extern uint8 WaveDAC8_1_wave2[];

void main()
{
    WaveDAC8_1_Start();
    for(;;)
    {
       CyDelay(2200); // Change to alternate waveforms.
      WaveDAC8_1_Stop();
      WaveDAC8_1_Wave1Setup( otherWave1, 8);
       WaveDAC8_1_Wave2Setup( otherWave2, 4);
      WaveDAC8_1_Start();

      CyDelay(2200);  // Change back to original waveforms.
      WaveDAC8_1_Stop();
      WaveDAC8_1_Wave1Setup(WaveDAC8_1_wave1, WaveDAC8_1_WAVE1_LENGTH);
```

```
        WaveDAC8_1_Wave2Setup(WaveDAC8_1_wave2, WaveDAC8_1_WAVE2_LENGTH);
        WaveDAC8_1_Start();
    }
}
```

DC and AC Electrical Characteristics

DC Characteristics (VDAC Mode)

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	–	V
		4-V scale, V_{DDA} = 5 V	–	4	–	V
	Monotonicity		–	–	Yes	–
V_{OS}	Zero-scale error		–	0	±0.9	LSB
FSGainErr	Full-scale gain error	1-V scale	–	±1.6	±2.5	%
		4-V scale	–	±1.5	±2.5	%
TCGainErr	Temperature coefficient, gain error	1-V scale	–	–	0.02	%FSR/°C
		4-V scale	–	–	0.02	%FSR/°C
I_{DD}	Operating current	Slow mode	–	–	100	µA
		Fast mode	–	–	500	µA

AC Electrical Characteristics (VDAC Mode)

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
SRP	Slew rate, step 10% to 90%	1-V scale, C_{LOAD} = 15 pF	–	0.3	0.5	µs
		4-V scale, C_{LOAD} = 15 pF	–	0.5	1.3	µs
SRN	Slew rate, step 90% to 10%	1-V scale, C_{LOAD} = 15 pF	–	0.3	0.5	µs
		4-V scale, C_{LOAD} = 15 pF	–	0.3	1.3	µs
Vn4V	Noise density[1]	4-V scale, code 255	–	3	–	µV/rtHz
Vn1V	Noise density**Error! Unknown**	1-V scale, code 255	–	750	–	nV/rtHz

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

	switch argument.					

DC Characteristics (IDAC Mode)

Parameter	Description	Conditions	Min	Typ	Max	Units
	Resolution		–	–	8	bits
I_{OUT}	Output current at code = 255	Range = 2.040 mA, code = 255, $V_{DDA} \geq 2.7$ V, R_{LOAD} = 600 Ω	–	2.040	–	mA
		Range = 2.040 mA, High mode, code = 255, $V_{DDA} \leq 2.7$ V, R_{LOAD} = 300 Ω	–	2.040	–	mA
		Range = 255 µA, code = 255, R_{LOAD} = 600 Ω	–	255	–	µA
		Range = 31.875 µA, code = 255, R_{LOAD} = 600 Ω	–	31.875	–	µA
	Monotonicity		–	–	Yes	
Ezs	Zero scale error		–	0	±1	LSB
Eg	Gain error		–	–	3.5	%
INL	Integral nonlinearity	Sink mode, range = 255 µA, Codes 8 to 255, R_{LOAD} = 2.4 kΩ, Cload = 15 pF	–	±1.2	±1.5	LSB
		Source mode, range = 255 µA, Codes 8 – 255, R_{LOAD} = 2.4 kΩ, C_{LOAD} = 15 pF	–	±0.9	±1	LSB
DNL	Differential nonlinearity	Sink mode, range = 255 µA, R_{LOAD} = 2.4 kΩ, C_{LOAD} = 15 pF	–	±0.3	±0.5	LSB
		Source mode, range = 255 µA, R_{LOAD} = 2.4 kΩ, C_{LOAD} = 15 pF	–	±0.3	±0.5	LSB
Vcompliance	Dropout voltage, source or sink mode	Voltage headroom at max current, R_{LOAD} to V_{DDA} or R_{LOAD} to V_{SSA}, V_{DIFF} from V_{DDA}	1	–	–	V
I_{DD}	Operating current, code = 255	Slow mode, source mode, range = 31.875 µA	–	–	44	µA
		Slow mode, source mode, range = 255 µA,	–	–	33	µA
		Slow mode, source mode, range = 2.04 mA	–	–	33	µA
		Slow mode, sink mode, range = 31.875 µA	–	–	36	µA
		Slow mode, sink mode, range = 255 µA	–	–	33	µA
		Slow mode, sink mode, range = 2.04 mA	–	–	33	µA
		Fast mode, source mode, range = 31.875 µA	–	–	310	µA
		Fast mode, source mode, range = 255 µA	–	–	305	µA
		Fast mode, source mode, range = 2.04 mA	–	–	305	µA

Parameter	Description	Conditions	Min	Typ	Max	Units
		Fast mode, sink mode, range = 31.875 µA	–	–	310	µA
		Fast mode, sink mode, range = 255 µA	–	–	300	µA
		Fast mode, sink mode, range = 2.04 mA	–	–	300	µA

AC Characteristics (IDAC Mode)

Parameter	Description	Conditions	Min	Typ	Max	Units
F_{DAC}	Update rate		–	–	8	Msps
T_{SETTLE}	Settling time to 0.5 LSB	Independent of IDAC range setting (I_{OUT}), full-scale transition, 600-Ω load, C_L = 15 pF, Fast mode	–	–	100	ns
		Independent of IDAC range setting (I_{OUT}), full-scale transition, 600-Ω load, C_L = 15 pF, Slow mode	–	–	1000	Ns

For more detailed specifications of the AC and DC performance of the WaveDAC8, refer to the VDAC8 and IDAC8 data sheets.

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.10		Initial document
1.20	Added the ability to changed waveform arrays during runtime.	Add flexibility.
1.24	Changed component placement	
1.30	Fixed some user interface issues.	
1.50	Replaced viDAC8 primitive with IDAC8 and VDAC8 instance.	
1.60	Added cystate file	For concept type components