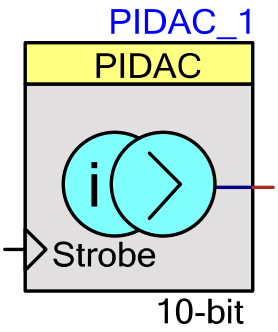
9 - 11 bit Parallel Current DAC (PIDAC)

V1.2

Features



- Two ranges 2047 uA and 255 uA
- Current sink or source selectable
- Configurable for 9,10 or 11 bits
- API compatible with standard IDAC8

General Description

The PIDAC component can be configured as a 9, 10, or 11 bit current DAC (Digital to Analog Converter). This DAC combines two current DACs in parallel in different overlapping ranges, to achieve higher than 8-bits of resolution. The PIDAC is also easily configured as a current sink or source and it's API is compatible to the standard IDAC8.

When to use a PIDAC

Use the PIDAC when more than 8-bits of resolution is needed.

Input/Output Connections

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

output – Analog

The output terminal is the connection to the current source/sink. It may be routed to any analog compatible pin on the device. When the highest current range is selected (2048 uA), the output should only be routed to a specific set of pins that provide a direct low resistive path. The best port to use is P3[0] since it has both a direct low resistive connection and the normal global bus connections.

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 pin will be visible and must be connected to a valid digital

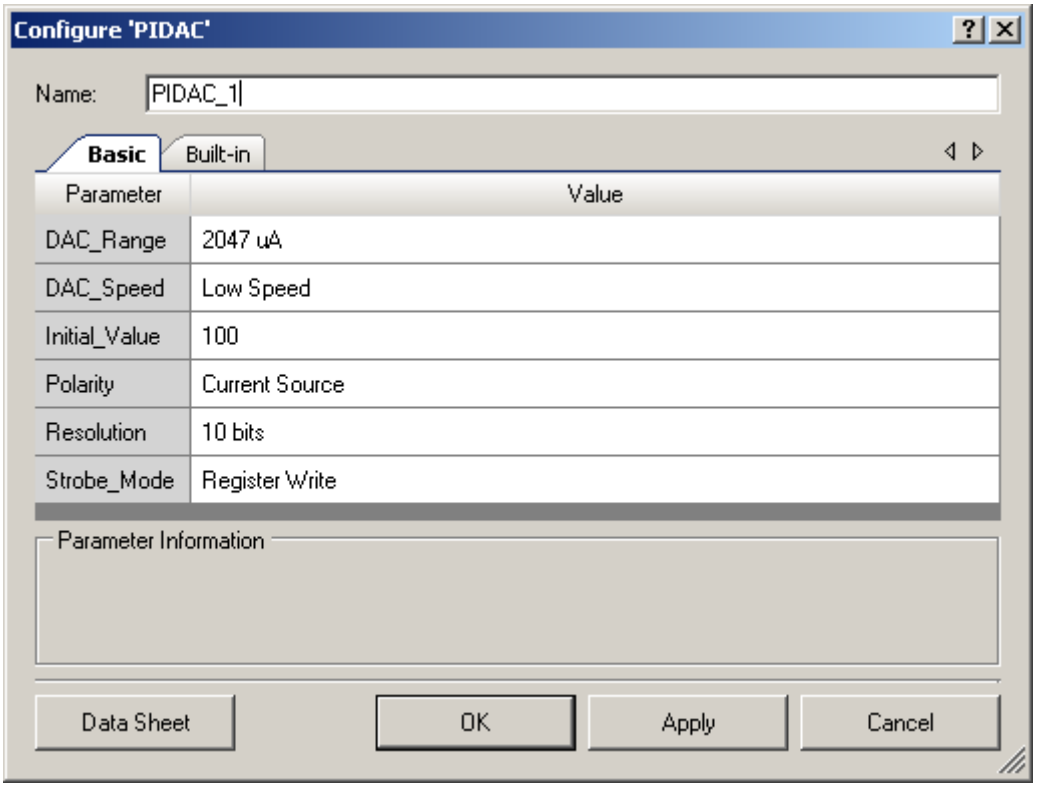
source. In this mode the data is transferred from the PIDAC register to the DAC on the next positive edge of the strobe signal.

If this parameter 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.

Parameters

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

Figure 1 Configure PIDAC Dialog



The PIDAC provides the following parameters.

DAC_Range

This parameter enables the designer to set one of three current ranges as the default value. The range may be changed at anytime during runtime with the SetRange() function. If the highest current range, "0 - 2040uA" is selected, the output should be routed to pin P3[0] for optimal performance.

Range	Lowest Value	Highest Value 9 / 10 / 11 bits	Step Size 9 / 10 / 11 bits
255 uA	0.00 uA	255.5 / 255.75 / 255.875 uA	500 / 250 / 125 nA
2047 uA	0.0 uA	2044 / 2046 / 2047 uA	4 / 2 / 1 uA

DAC_Speed

This parameter provides two settings for the designer, Low Speed and High Speed. In the Low Speed mode, the settling time is slower but it consumes less operating current. In the High Speed mode, the current settle much faster, but at a cost of more operating current.

Initial_Value

This is the initial value the PIDAC 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 $2^{bits} - 1$.

Polarity

The Polarity parameter allows the designer to select whether the PIDAC sinks or sources current to its load. When the "Current Source" option is selected, the output of the DAC will source current to a load that is connected to Vss. In the "Current Sink" mode, it will supply current to a load that is connected to Vdda.

Resolution

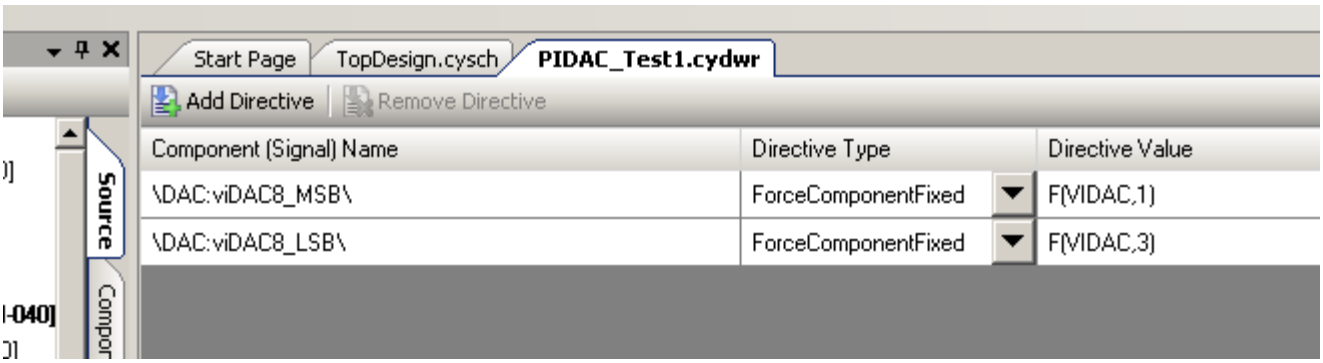
This parameter allows the designer to select between 9, 10, or 11 bits of resolution.

Strobe_Mode

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

Placement

To optimize performance when using the 2047 uA range, it is best to force the placement of the individual IDACs. In the Design Wide Resource window, select the “Directives” tab and enter the following directives. Under the “Component (Signal) Name” column replace the occurrence of “\DAC:viDAC8…” with “\xxx:viDAC8…” where “xxx” is the instance name of the PIDAC component.



Resources

The PIDAC uses two viDAC8 analog blocks.

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 "PIDAC_1" to the first instance of a component in a given design. You can rename it 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 "PIDAC".

Function	Description
void PIDAC_Start(void)	Initialize the PIDAC with default customizer values. Enable and power up the PIDAC.
void PIDAC_Stop(void)	Disables the PIDAC and sets it to the lowest power state.
Void PIDAC_SetSpeed(uint8 speed)	Set DAC speed.
void PIDAC_SetPolarity(uint8 polarity)	Sets the output mode to current sink or source.
void PIDAC_SetRange(uint8 range)	Sets full scale range for PIDAC.
void PIDAC_SetValue(uint16 value)	Sets value between 0 and $2^{bits} - 1$.
void PIDAC_Init(void)	Initializes or restores default PIDAC configuration
void PIDAC_Enable(void)	Enables the PIDAC.
void PIDAC_SaveConfig(void)	Empty function. Provided for future usage.

Function	Description
void PIDAC_RestoreConfig(void)	Empty function. Provided for future usage.
void PIDAC_Sleep(void)	Stops and saves the user configuration.
void PIDAC_WakeUp(void)	Restores and enables the user configuration.

void PIDAC_Start(void)

Description: Initialize the PIDAC with default customizer values. Enable and power up the PIDAC to the given power level.

Parameters: None

Return Value: None

Side Effects: None

void PIDAC_Stop(void)

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

Parameters: None

Return Value: None

Side Effects: None

void PIDAC_SetSpeed(uint8 speed)

Description: Set DAC speed.

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

Speed Setting	Notes
PIDAC_LOWSPEED	Low speed (low power)
PIDAC_HIGHSPEED	High speed (high power)

Return Value: None

Side Effects: None

void PIDAC_SetPolarity(uint8 polarity)

<Description: Sets output polarity to sink or source.

Parameters: (uint8) polarity: Sets current sink or source functionality, see table below.

Polarity Setting	Notes
PIDAC_SOURCE	Set mode as current source.
PIDAC_SINK	Set mode to current sink.

Return Value: None

Side Effects: None

void PIDAC_SetRange(uint8 range)

Description: Sets full scale range for PIDAC.

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

Power Setting	Notes
PIDAC_RANGE_255uA	Set full scale range to 255 uA
PIDAC_RANGE_2mA	Set full scale range to 2047 uA

Return Value: None

Side Effects: None

void PIDAC_SetValue(uint16 value)

Description: Sets value to output on PIDAC. Valid values are between 0 and $2^{bits} - 1$. See table below.

Parameters: (uint16) value: Maximum value is dependent on the resolution parameter

Resolution	Range
9-Bits	0 to 511
10-Bits	0 to 1023
11-Bits	0 to 2047

Return Value: None

Side Effects: None

void PIDAC_Init(void)

Description: Initializes/Restores default PIDAC 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 PIDAC_Enable(void)

Description: Enables PIDAC.

Parameters: None

Return Value: None

Side Effects: None

void PIDAC_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 PIDAC_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 PIDAC_Sleep() may lead to unpredictable behavior.

void PIDAC_SaveConfig(void)

Description: Saves the user configuration.

Parameters: None

Return Value: None

Side Effects: None

void PIDAC_RestoreConfig(void)

Description: Restores the user configuration.

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 PIDAC reference component. This example assumes the component has been placed in a design with the default name "PIDAC_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()
{
    PIDAC_1_Start();              // Enable PIDAC
    PIDAC_1_SetValue(500);        // Set output to 500uA in 11-bit mode
}
```

Functional description and schematic

The PIDAC component consist of two current DACs connected in parallel. The output current is the sum of both DACs. The SetValue API splits the word written to the DAC and writes the most significat bits to the viDAC8_MSB and the remaining least significant bits are written to the viDAC8_LSB DAC. The Strobe input is connected to both DACs so a clock can be used to synchronize the DAC outputs. The Strobe input with a clock and ISR can be used to generate good periodic sampling.

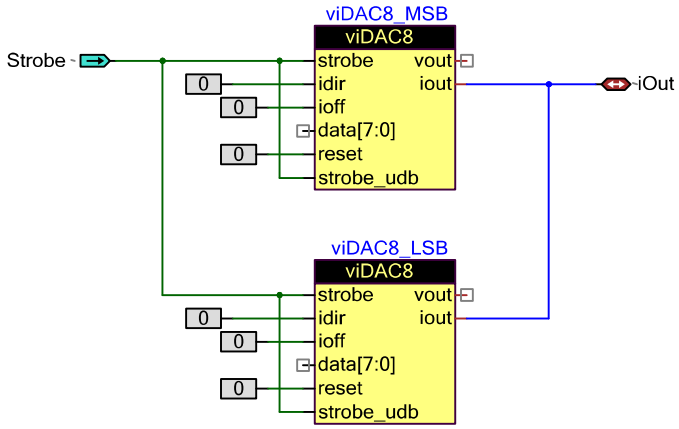


Figure 2 PIDAC schematic

Global Variables

Variable	Description
PIDAC_initVar	Indicates whether the PIDAC has been initialized. The variable is initialized to 0 and set to 1 the first time PIDAC_Start() is called. This allows the component to restart without reinitialization after the first call to the PIDAC_Start() routine. If reinitialization of the component is required, then the PIDAC_Init() function can be called before the PIDAC_Start() or PIDAC_Enable() function.

Registers

The functions provided support most of the common runtime functions that are required for most applications. The register reference below provides a brief description for the advanced user.

Table 1 PIDAC_MSB_CR0

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

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

Table 2 PIDAC_LSB_CR0

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

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

Table 3 PIDAC_MSB_CR1

Bits	7	6	5	4	3	2	1	0
Value	reserved		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 4 PIDAC_LSB_CR1

Bits	7	6	5	4	3	2	1	0
Value	reserved		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 5 PIDAC_MSB_Data

Bits	7	6	5	4	3	2	1	0
Value 10-Bit	Data[8:1]							
Value 10-Bit	Data[9:2]							
Value 11-Bit	Data[10:3]							

- 9-Bit Mode Data [8:1]: DAC data register
- 10-Bit Mode Data [9:2]: DAC data register
- 11-Bit Mode Data [10:3]: DAC data register

Table 6 PIDAC_LSB_Data

Bits	7	6	5	4	3	2	1	0
Value 9-bit	NA					Data[0]	NA	
Value 10-bit	NA					Data[1:0]		NA
Value 11-Bit	NA					Data[2:0]		

- 9-Bit Mode Data [0]: DAC data register
- 10-Bit Mode Data [1:0]: DAC data register
- 11-Bit Mode Data [2:0]: DAC data register

DC and AC Electrical Characteristics

The following values are indicative of expected performance and based on initial characterization data.

DC Characteristics

Parameter	Description	Conditions	Min	Typ	Max	Units
	Output current					
Iout	High	Vdda ≤ 2.7 V, RL 300 Ω	-	-	2.047	mA
	Medium	RL 600 Ω	-	-	255	µA
INL	Integral non linearity	RL 600 Ω, CL=15 pF , 9-bit	-	± 1.5	-	LSB
		RL 600 Ω, CL=15 pF , 10-bit	-	± 3	-	LSB
		RL 600 Ω, CL=15 pF , 11-bit		± 6	-	LSB
DNL	Differential non linearity	RL 600 Ω, CL=15 pF , 9-bit	-	0.25	-	LSB
		RL 600 Ω, CL=15 pF , 10-bit	-	0.5	-	LSB
		RL 600 Ω, CL=15 pF , 11-bit	-	1.1	-	LSB
Ezs	Zero scale error		-	0	±1	LSB
Eg	Gain error	Uncompensated	-	-	2.5	%
		Temperature compensated	-	-	TBD	%
IDAC_ICC	DAC current low speed mode	Code = 0	-	-	200	µA
IDAC_ICC	DAC current high speed mode	Code = 0	-	-	1000	µA

AC Characteristics

Parameter	Description	Conditions	Min	Typ	Max	Units
Fdac	Update rate		-	-	8	Msps
Tsettle	Settling time to 0.5LSB	Full scale transition, 600 Ω load, CL = 15 pF				
	Fast mode	Independent of IDAC range setting (Iout)	-	-	100	ns
	Slow mode	Independent of IDAC range setting (Iout)	-	-	1000	ns

Reference Component Changes

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

Version	Description of Changes
1.0	First release
1.1	Moved component to Concept library tab.
1.2	Updated to support PSoC Creator 2.1 SP1