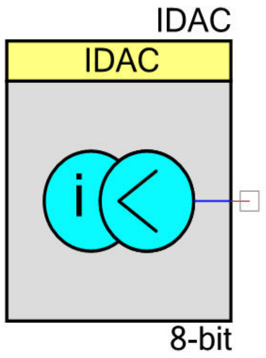
PSoC 4 Current Digital to Analog Converter (IDAC)
1.0

Features



- 7 or 8-bit resolution
- 7-bit range: 0 to 152.4 or 304.8 μA
- 8-bit range: 0 to 306 or 612 μA
- Current sink or source selectable

General Description

The IDAC component gives you a programmable current with a resolution of either 7 or 8 bits. The 8-bit ranges are approximately 612 and 306 μA and the 7-bit ranges are approximately 304.8 and 152.4 μA.

When to Use IDAC

- Resistance measurements
- Current sink or source
- Capacitance measurements other than CapSense
- Sensor current
- Temperature measurement (diode sensor)

Input/Output Connections

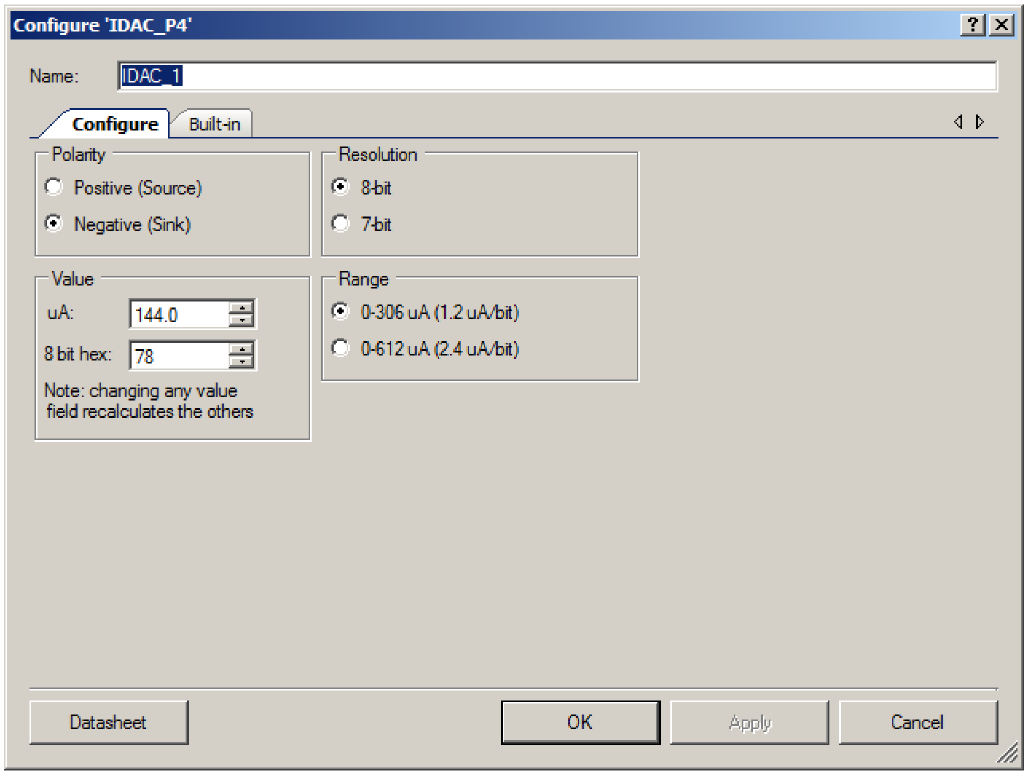
This section describes the various IDAC input and output connections.

Iout – Analog

The connection to the DAC's current source/sink.

Component Parameters

Drag the IDAC onto your design desktop and double-click it to open the Configure dialog box.



Name: IDAC_1

Polarity
- Positive (Source)
- [✓ Negative (Sink)]

Resolution
- [✓ 8-bit]
- 7-bit

Value
uA: 144.0
8 bit hex: 78
Note: changing any value field recalculates the others

Range
- [✓ 0-306 uA (1.2 uA/bit)]
- 0-612 uA (2.4 uA/bit)

Polarity

Mode of operation. Negative/Sink (default) or Positive/Source.

Resolution

Resolution of the IDAC. 8-bit (default) or 7-bit.

Range

IDAC dynamic range:

- 8-bit resolution - 306 μA (default) or 612 μA.
- 7-bit resolution 152.4 μA or 304.8 μA

Value

IDAC hexadecimal value (default is 78).

When changing modes the code value is fixed and the current value changes. When you switch from 8-bit to 7-bit and the value exceeds the 7-bit range, the value automatically changes to 7F. When the code value changes the current value updates and vice versa.

Placement

The PSoC 4 IDACs are part of the CapSense CSD hardware block. Two IDACs are available. The 8-bit IDAC is connected to AmuxBusA and the 7-bit IDAC is connected to AmuxBusB.

Resources

Resolution (bits)	Resource Type
	CSD IDAC block
7	1
8	1

Application Programming Interface

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

By default, PSoC Creator assigns the instance name "IDAC_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 table is "IDAC".

Functions

Function	Description
IDAC_Start()	Performs all of the required initialization for the component and enables power to the block.
IDAC_Stop()	Turn off the IDAC block.
IDAC_Init()	Initializes or restores the component according to the customizer Configure dialog settings.
IDAC_Enable()	Activates the hardware and begins component operation.
IDAC_SetValue()	Sets the DAC's output value.

Function	Description
IDAC_Sleep()	This is the preferred API to prepare the component for sleep.
IDAC_Wakeup()	This is the preferred API to restore the component to the state when IDAC_Sleep() was called.
IDAC_SaveConfig()	Saves the configuration of the component.
IDAC_RestoreConfig()	Restores the configuration of the component.

void IDAC_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 component is initialized to the configured settings. When called to restart the IDAC following a IDAC_Stop() call, the current component parameter settings are retained.

Parameters: None

Return Value: None

Side Effects: None

void IDAC_Stop(void)

Description: Turn off the IDAC block.

Parameters: None

Return Value: None

Side Effects: Does not affect the IDAC settings.

void IDAC_Init(void)

Description: Initializes or restores the component according to the customizer Configure dialog settings. It is not necessary to call IDAC_Init() because the IDAC_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 IDAC_Enable(void)

Description:	Activates the hardware and begins component operation. It is not necessary to call IDAC_Enable() because the IDAC_Start() API calls this function, which is the preferred method to begin component operation.
Parameters:	None
Return Value:	None
Side Effects:	None

void IDAC_SetValue(uint32 value)

Description:	Sets the DAC's output value. The least significant 7 or 8 bits are used depending on the resolution of the DAC. This function sets the value from 0 to 0xFF (for 8-bit IDAC) or from 0 to 0x7F (for 7-bit IDAC). The user is responsible for calculation of the correct IDAC value depending on selected resolution and range.
Parameters:	(uint32) value
Return Value:	None

void IDAC_Sleep(void)

Description:	This is the preferred API to prepare the component for sleep. The IDAC_Sleep() API saves the current component state. Then it calls the IDAC_Stop() function and calls IDAC_SaveConfig() to save the hardware configuration. Call the IDAC_Sleep() function before calling the CySysPmDeepSleep() or the CySysPmHibernate() functions.
Parameters:	None
Return Value:	None
Side Effects:	None

void IDAC_Wakeup(void)

Description:	This is the preferred API to restore the component to the state when IDAC_Sleep() was called. The IDAC_Wakeup() function calls the IDAC_RestoreConfig() function to restore the configuration. If the component was enabled before the IDAC_Sleep() function was called, the IDAC_Wakeup() function will also re-enable the component.
Parameters:	None
Return Value:	None
Side Effects:	Calling the IDAC_Wakeup() function without first calling the IDAC_Sleep() or IDAC_SaveConfig() function may produce unexpected behavior.

void IDAC_SaveConfig(void)

Description: This function saves the component configuration and non-retention registers. This function is called by the IDAC_Sleep() function.

Parameters: None

Return Value: None

Side Effects: None

void IDAC_RestoreConfig(void)

Description: This function restores the component configuration and non-retention registers. This function is called by the IDAC_Wakeup() function.

Parameters: None

Return Value: None

Side Effects: None

Global Variables

Function	Description
IDAC_initVar()	Indicates whether or not the IDAC was initialized. The variable is initialized to 0 and set to 1 the first time IDAC_Start() is called. This allows the component to restart without reinitialization after the first call to the IDAC_Start() routine. If reinitialization of the component is required, call IDAC_Init() before calling IDAC_Start(). Alternatively, you can reinitialize the IDAC by calling the IDAC_Init() and IDAC_Enable() functions.

Sample Firmware Source Code

PSoC Creator has 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 diagram. 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.

See the "Find Example Project" topic in the PSoC Creator Help for more information.

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 gives you information on component specific deviations. The project deviations are described in the MISRA Compliance section of the *System Reference Guide* along with information on the MISRA compliance verification environment.

The IDAC component does not have any specific deviations.

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 were taken with the associated compiler configured in release mode with optimization set for size. For a specific design, you can analyze the map file generated by the compiler to determine the memory usage.

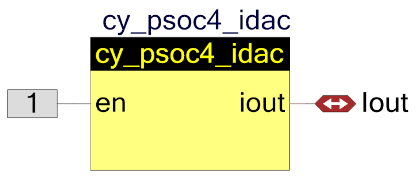
Configuration	Flash Bytes	SRAM Bytes
7 or 8-bit	472	8

Functional Description

Only one instance of each of the 7-bit and 8-bit IDAC components is available in a design. These are shared with the CapSense CSD component. If the CapSense component is present in the design it will use the 8-bit IDAC and depending on the configuration it may also use the 7-bit IDAC.

Block Diagram and Configuration

The component uses the cy_psoc4_idac primitive with hardware enable connected to Logic High. It is configured using the CSD block configuration registers.



Registers

See the chip Technical Reference Manual (TRM) for more information about registers.

DC and AC Electrical Characteristics

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.

AC Specifications

Parameter	Description	Min	Typ	Max	Units	Conditions
IDAC1	DNL for 8-bit resolution	-1	–	1	LSB	
IDAC1	INL for 8-bit resolution	-3	–	3	LSB	
IDAC2	DNL for 7-bit resolution	-1	–	1	LSB	
IDAC2	INL for 7-bit resolution	-3	–	3	LSB	
IDAC1_CRT1	Output current of Idac1 (8-bits) in High range	–	612	–	µA	
IDAC1_CRT2	Output current of Idac1(8-bits) in Low range	–	306	–	µA	
IDAC2_CRT1	Output current of Idac2 (7-bits) in High range	–	305	–	µA	
IDAC2_CRT2	Output current of Idac2 (7-bits) in Low range	–	153	–	µA	
IDAC_SET8	Settling time to 0.5 LSB for 8-bit IDAC	–	–	10	µs	For PSoC 4000 family. Full-scale transition. No external load.
IDAC_SET7	Settling time to 0.5 LSB for 7-bit IDAC	–	–	10	µs	For PSoC4000 family Full-scale transition. No external load.

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.0.c	Edited datasheet.	Updated characterization data.
1.0.b	Edited datasheet.	Updated characterization data.
1.0.a	Edited datasheet.	Updated the current ranges.
1.0	Initial release	