Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

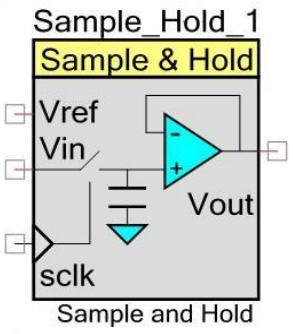
Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

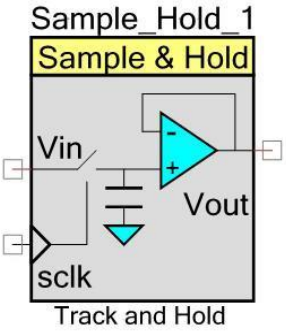
Sample/Track and Hold Component

1.40

Features

- Two operating modes: Sample and Hold, Track and Hold
- Four power mode settings





General Description

The Sample/Track and Hold component provides a way to sample a continuously varying analog signal and to hold or freeze its value for a finite period of time. It supports both Track and Hold and Sample and Hold functions, which can be selected in the customizer.

Input/Output Connections

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

Vin – Analog

The Vin terminal is the connection to the Sample/Track and Hold component's input. Connect any analog signals to be sampled or tracked to this input.

Vout – Analog

The Vout terminal is the connection to the Sample/Track and Hold's output. This signal can be routed to any pin or analog input; for instance, a comparator or ADC.

sclk – Input *

The sclk input defines the sample clock input to the Sample/Track and Hold component.

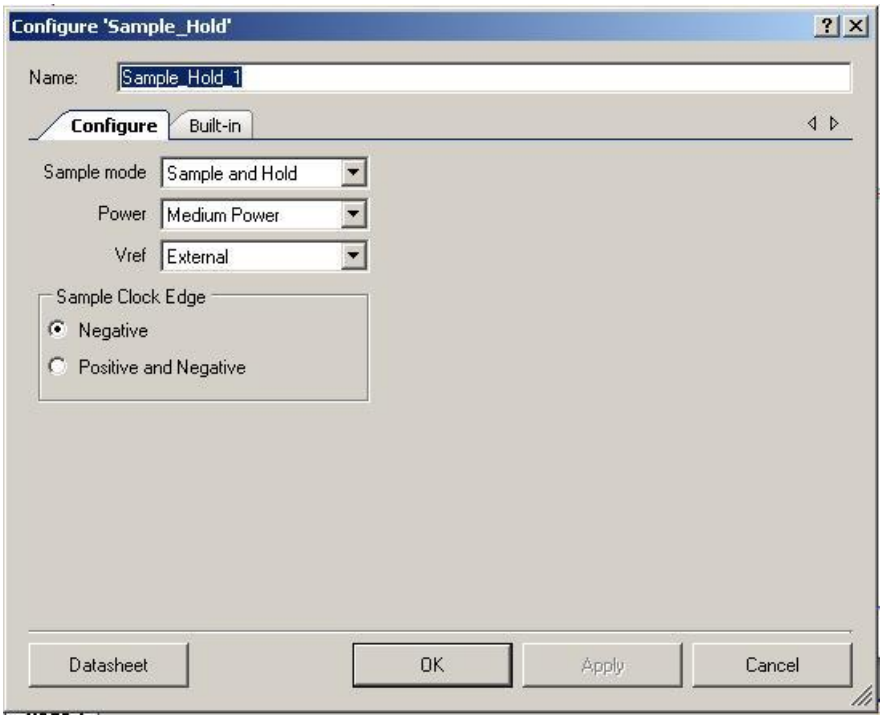
Vref – Input *

The Vref input is an optional input and is selected with the **Sample mode** parameter.

- If **Sample mode** is **Sample and Hold** and **Vref** is **External** then this pin is visible and is connected to a valid Vref source.
- If **Sample mode** is **Track and Hold** this pin disappears from the symbol.

Parameters and Setup

Drag a Sample/Track and Hold onto your design and double-click it to open the **Configure** dialog.



The Sample/Track and Hold component provides the following parameters.

Sample mode

The **Sample and Hold** option samples the signal on the falling edge of the clock, or optionally on both the falling and rising edge of the clock.

The **Track and Hold** mode samples the signal on the falling edge of the sample clock, but tracks the input signal while the sample clock remains low.

Power

This parameter sets the initial drive power of the Sample/Track and Hold component. The power determines the speed with which the Sample/Track and Hold reacts to changes in the input signal. There are four power settings available: **Minimum Power**, **Low Power**, **Medium Power** (default), and **High Power**. A **Minimum Power** setting results in the slowest response time and a **High Power** setting results in the fastest response time.

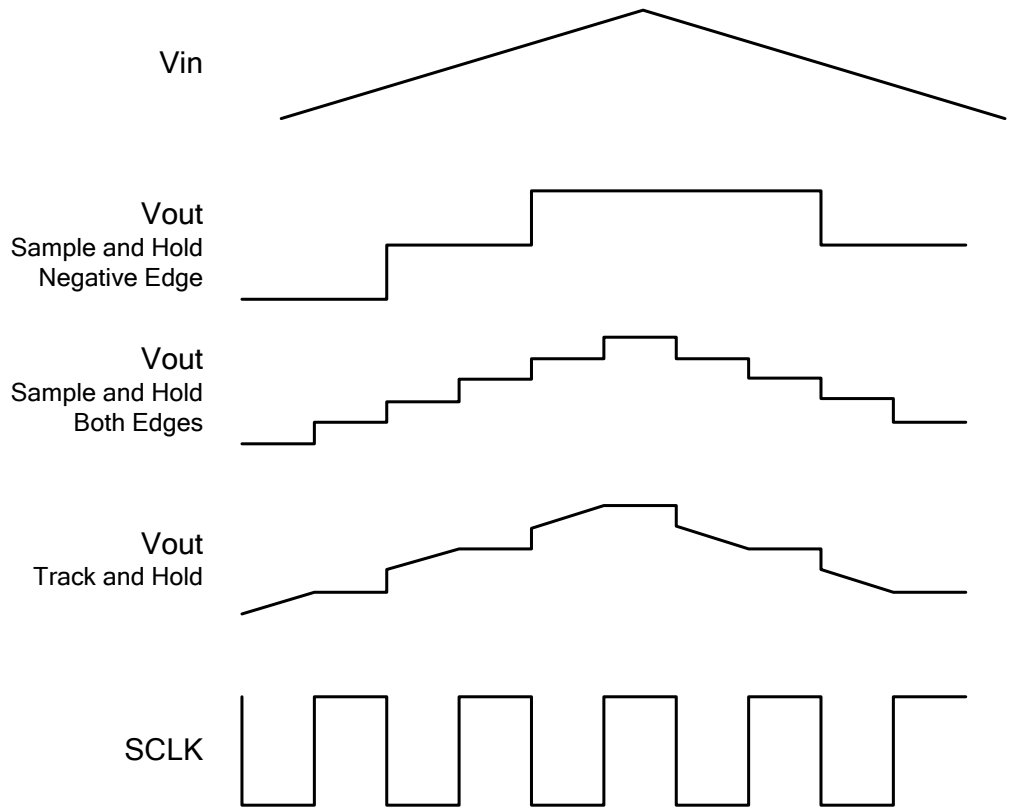
Vref

Vref mode is used to select the reference voltage as **Internal** or **External**. If **Vref** is **External**, an external reference voltage is applied to the Sample/Track and Hold component. If the Vref mode is set as **Internal**, the component takes the reference voltage from the internal source Vss, which is the ground signal internal to the component providing the amplifier reference.

Sample Clock Edge

This parameter provides the clock edge settings for the designer. This parameter is valid only in Sample and Hold mode. There are two types of edge settings: **Negative** and **Positive and Negative**.

Figure 1. Sample/Track and Hold Waveforms



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 “Sample_Hold_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 “Sample_Hold.”

Function	Description
Sample_Hold_Start()	Configures and enables power of Sample/Track and Hold.
Sample_Hold_Stop()	Turns off the Sample/Track and Hold block.
Sample_Hold_SetPower()	Sets the drive power of Sample/Track and Hold.
Sample_Hold_Sleep()	Puts the Sample/Track and Hold into sleep mode.
Sample_Hold_Wakeup()	Wakes up Sample/Track and Hold.
Sample_Hold_Init()	Initializes the Sample/Track and Hold component.
Sample_Hold_Enable()	Activates the hardware and begins component operation.
Sample_Hold_SaveConfig()	Empty function. Provided for future use.
Sample_Hold_RestoreConfig()	Empty function. Provided for future use.

void Sample_Hold_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 sample mode, clock edge, and power are set to their default values. When called to restart following a Sample_Hold_Stop() call, the current component parameter settings are retained.

Parameters: None

Return Value: None

Side Effects: None

void Sample_Hold_Stop(void)

Description: Turns off the Sample/Track and Hold block.

Parameters: None

Return Value: None

Side Effects: Does not affect Sample and Hold modes or power settings.

void Sample_Hold_SetPower(uint8 power)

Description: Sets the drive power to one of four settings; minimum, low, medium, or high.

Parameters: uint8 range: Sets full scale range for Sample_Hold. See the following table for ranges.

Power Setting	Notes
Sample_Hold_MINPOWER	Lowest active power and slowest reaction time
Sample_Hold_LOWPOWER	Low power and speed
Sample_Hold_MEDPOWER	Medium power and speed
Sample_Hold_HIGHPOWER	Highest active power and fastest reaction time

Return Value: None

Side Effects: None

void Sample_Hold_Sleep(void)

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

Call the Sample_Hold_Sleep() function before calling the CyPmSleep() or the CyPmHibernate() function.

Parameters: None

Return Value: None

Side Effects: None

void Sample_Hold_Wakeup(void)

Description: This is the preferred API to restore the component to the state when Sample_Hold_Sleep() was called. The Sample_Hold_Wakeup() function calls the Sample_Hold_RestoreConfig() function to restore the configuration. If the component was enabled before the Sample_Hold_Sleep() function was called, the Sample_Hold_Wakeup() function also re-enables the component.

Parameters: None

Return Value: None

Side Effects: Calling the Sample_Hold_Wakeup() function without first calling the Sample_Hold_Sleep() or Sample_Hold_SaveConfig() function may produce unexpected behavior.

void Sample_Hold_Init(void)

Description: Initializes or restores the component according to the customizer Configure dialog settings. It is not necessary to call Sample_Hold_Init() because the Sample_Hold_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 Sample_Hold_Enable(void)

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

Parameters: None

Return Value: None

Side Effects: None

void Sample_Hold_SaveConfig(void)

Description: Empty function. Provided for future use.

Parameters: None

Return Value: None

Side Effects: None

void Sample_Hold_RestoreConfig(void)

Description: Empty function. Provided for future use.

Parameters: None

Return Value: None

Side Effects: None

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 Sample/Track and Hold 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.

Resources

The Sample/Track and Hold component uses one SC/CT block per instance.

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	168	2	244	5

DC and AC Electrical Characteristics

Specifications are valid for –40 °C ≤ TA ≤ 85 °C and TJ ≤ 100 °C, except where noted.
Specifications are valid for 1.71 V to 5.5 V, except where noted.

Parameter	Conditions and Notes	Min	Typical	Max	Units
Input offset voltage		–	–	–	mV
Quiescent current		–	0.9	2	mA
sclk, sample clock frequency	Sample and Hold mode Track and Hold mode	–	–	4	MHz
Vin, input signal frequency	Sample and Hold mode Track and Hold mode	–	–	14	MHz
SR					
Slew rate		–	–	3	V/µs
Acquisition time	A 5.5-V step to 1%	–	–	1	µs
Droop Rate		–	–	–	mV/ms
Feedthrough Attenuation Ratio	Input 310 kHz, 4V p-p	–	30	–	dB

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.40.d	Minor datasheet edit.	1.40.c
1.40.c	Minor datasheet edit.	
1.40.b	Updated datasheet.	Added "Feedthrough Attenuation Ratio" parameter to the Characteristics section.
1.40.a	Updated datasheet.	Removed references to obsolete PSoC 5 device.
1.40	Added variable Vdda support.	
	Added MISRA Compliance section.	The component does not have any specific deviations.
1.30	For low voltage VDDA operation uses a boost clock shared by all the SC/CT based components.	Reduces the number of analog clocks required in the system for boost clocks. With this change a single boost clock is shared instead of using a separate clock for each SC/CT based component
1.20	Added support for PSoC5LP silicon. CYREENTRANT keyword added to all APIs.	

Version	Description of Changes	Reason for Changes / Impact
	Updated DC and AC Electrical characteristics.	
1.10	Datasheet changes: ▪ Added Sample/Track and Hold waveforms above the Resources section ▪ Added component symbol for track and hold mode ▪ Minor edits and updates	
	Solved internal Vref issue. Updated the C code changes in the Sample_Hold_Init() API.	Internal Vref issue existed in both sample and track and hold modes.

© Cypress Semiconductor Corporation, 2012-2017. 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.