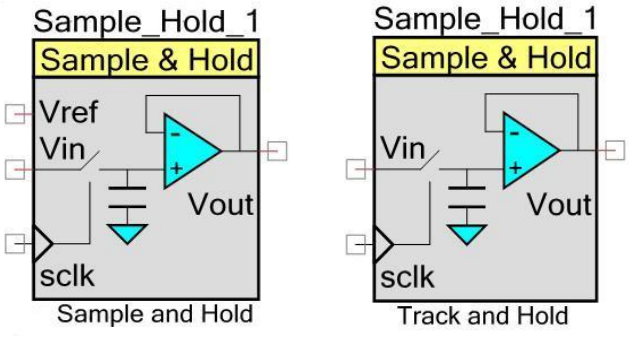
Sample/Track and Hold Component

1.20

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.

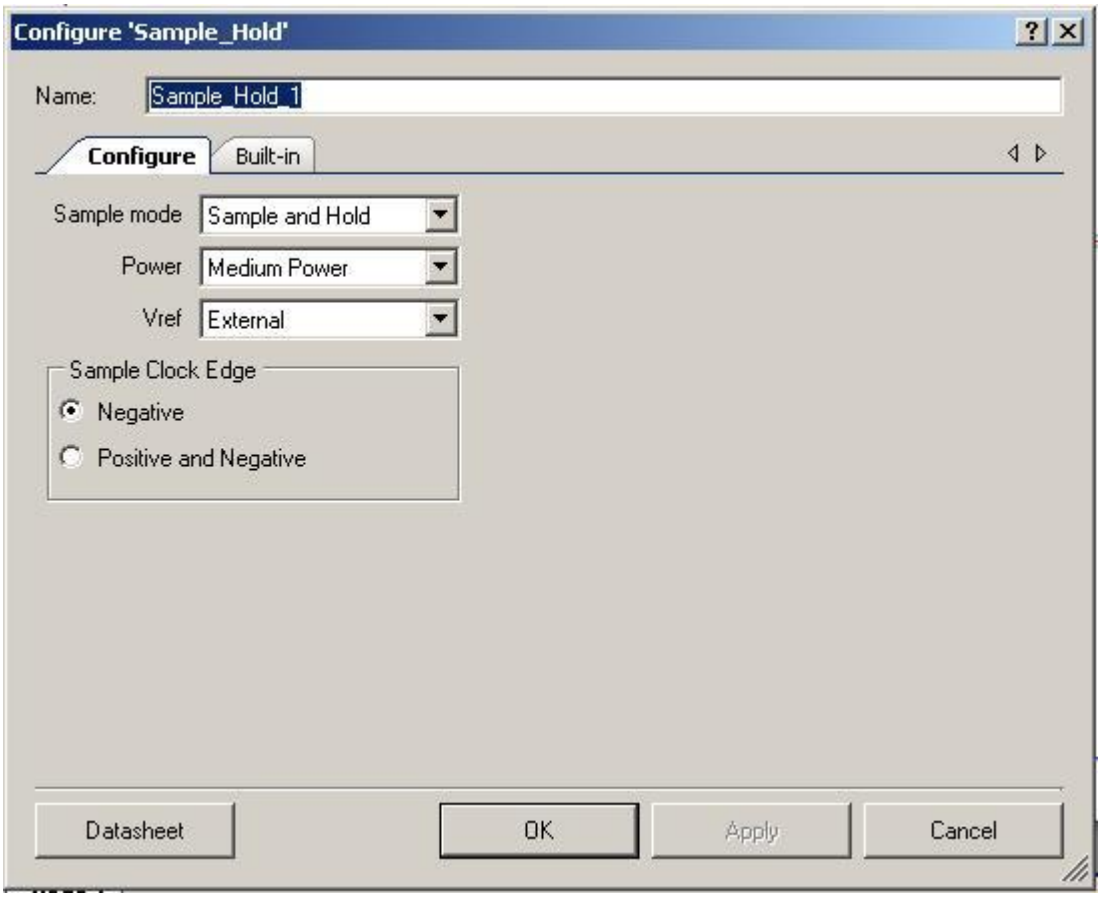
Vref – Input *

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.

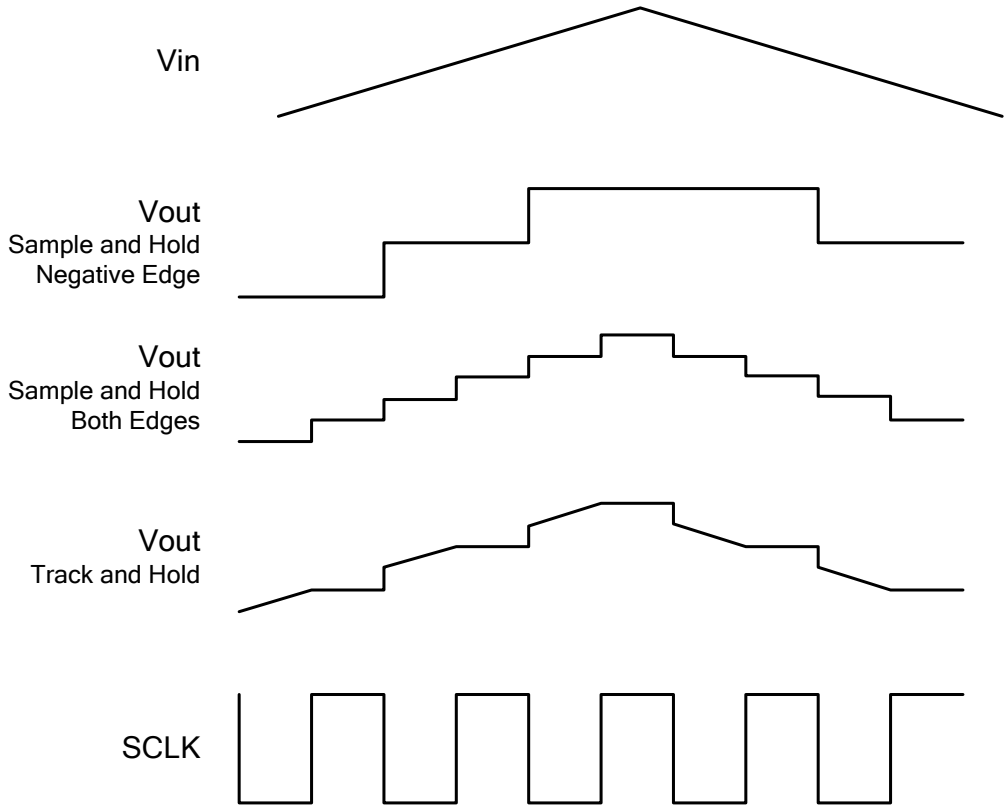
Vref

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.

Function	Description
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 the Sample_Hold 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

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.

User Registers

The functions provided support most of the common run-time functions that are required for most applications. The following register reference provides a brief description for the advanced user.

Table 1. SCx_CR0

Bits	7	6	5	4	3	2	1	0
Value	RSVD		dft		mode			NA

- mode: Configuration mode for SC block.

Table 2. SCx_CR1

Bits	7	6	5	4	3	2	1	0
Value	RSVD		gain	div2	comp		drive	

- gain: Controls the ratio of the feedback cap for S/H Mixer mode and PGA mode.
- div2: When 0, the sample clock only needs to be half the desired sample frequency for the S/H Mixer mode.
- Comp[1:0]: Selects between various compensation capacitor sizes.
- Drive[1:0]: Selects between current settings in output buffer.

Table 3. SCx_CR2

Bits	7	6	5	4	3	2	1	0
Value	gndvref	rval			redc		R20_40b	bias_cntl

- gndvref: Enable direct ground connection to inverting input.
- Rval[2:0]: Feedback resistor control.
- Redc[1:0]: Capacitance adjustment between output and first stage.
- r20_40b: Input impedance 20K or 40K
- bias_cntl: Bias control, normal or ½ (low)

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 5 (GCC)		PSoC 5LP (GCC)	
	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes
Default	168	2	300	12	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

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.20.a	Minor datasheet edit.	
1.20	Added support for PSoC5LP silicon. CYREENTRANT keyword added to all APIs.	
	Updated DC and AC Electrical characteristics.	

Version	Description of Changes	Reason for Changes / Impact
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, 2012-2015. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control, or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.