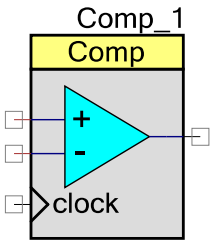
Comparator (Comp)
1.60

Features



- Low input offset
- User controlled offset calibration
- Multiple speed modes
- Low power mode
- Output routable to digital logic blocks or pins
- Selectable output polarity
- Configurable operation mode during Sleep and Hibernate

General Description

The Comparator (Comp) component provides a hardware solution to compare two analog input voltages. The output can be sampled in software or digitally routed to another component. Three speed levels are provided to enable you to optimize for speed or power consumption. A reference or external voltage may be connected to either input.

You can also invert the output of the comparator using the Polarity parameter.

When to use a Comparator

The Comparator can provide a fast comparison between two voltages as compared to using an ADC. Although an ADC can be used with software to compare multiple voltages levels, applications requiring fast response or little software intervention are good candidates for this comparator. Some example applications include CapSense®, power supplies, or simple translation from an analog level to a digital signal.

A common configuration is to create an adjustable comparator by connecting a voltage DAC to the negative input terminal.

Input/Output Connections

This section describes the input and output connections for the Comp. 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.

Positive Input – Analog

This input is usually connected to the voltage that is being compared. This input can be routed to GPIOs and internal signals through analog globals, and to a selection of references.

Negative Input – Analog

This input is usually connected to the reference voltage. This input can be routed to GPIOs and internal signals through the analog globals and to a selection of references.

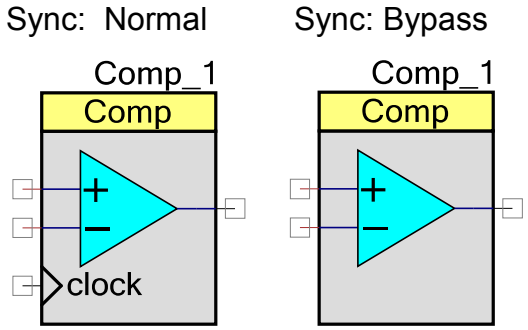
Comparator Out – Digital Output

The output of the comparison. For the non-inverting configuration, this output goes high when the positive input voltage is greater than the negative input voltage. If the polarity is set to inverting, the output will go high when the negative input voltage is greater than the positive input voltage. The output can be routed to the digital interconnect and interrupt structures.

clock – Digital Input *

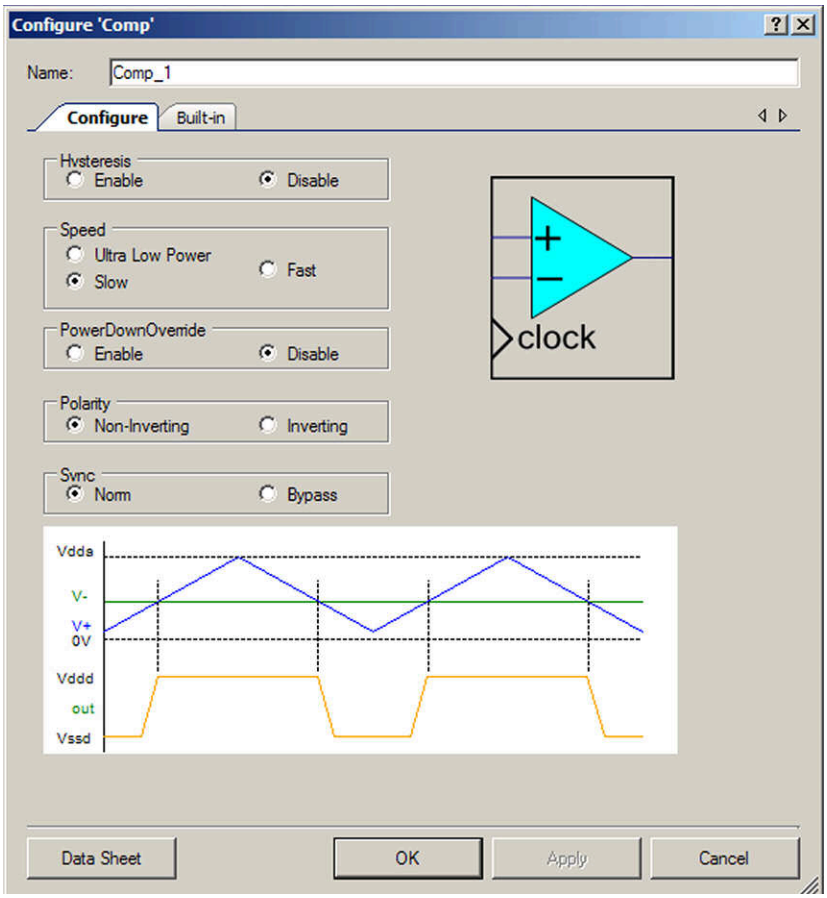
The clock input will synchronize the comparator output to the rising edge of the clock when the Sync parameter is set to "Normal." This forces the comparator output to be sampled on the rising edge of the clock.

When the Sync parameter is set to "Bypass" the output is not synchronized and the clock input terminal no longer displayed on the component symbol.



Parameters and Setup

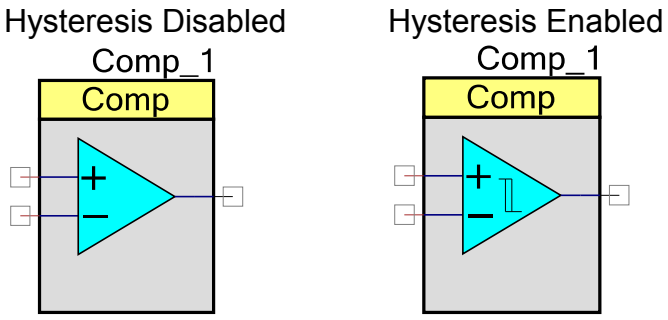
Drag a Comparator onto your design and double-click it to open the Configure dialog.



The Comparator provides the following parameters.

Hysteresis

This parameter enables allows you to add approximately 10 mV of hysteresis to the comparator. This will help to ensure that slowly moving voltages or slightly noisy voltages will not cause the output of the comparator to oscillate when the two input voltages are near equal.



Speed

This parameter provides a way for the user to optimize speed verses power consumption.

Speed Options	Description
Ultra Low Power	Use this setting for very low power applications.
Slow (default)	Use this setting for signals requiring response times slower than 80ns
Fast	Use this setting for signals requiring response times faster than 80ns

Power Down Override

Enabling the power down override parameter causes the comparator to stay active during Sleep and Hibernate modes.

Polarity

This parameter allows you to invert the output of the comparator. This is useful for peripherals that require an inverted signal from the comparator. The sampled signal state returned by the software API is not affected by this parameter.

Polarity Options	Description
Inverting	Output goes high when positive input is less than the negative input
Non Inverting (default)	Output goes high when positive input is greater than negative input

Sync

This parameter selects between synchronizing the output with a clock and connecting directly to the comparator output. When Normal is selected, the output will change on the rising edge of the clock input.

Sync Options	Description
Normal (default)	Sync the comparator output with the clock input.
Bypass	Connect the analog comparator directly to the output signal.

Placement

There are no placement specific options.

Resources

Analog Block	Digital Blocks					API Memory (Bytes)		Pins (per External I/O)
	Datapaths	Macro cells	Status Registers	Control Registers	Counter7	Flash	RAM	
Comparator fixed HW *	N/A	N/A	N/A	N/A	N/A	597	2	3

*The Comparator component uses one analog comparator block.

Application Programming Interface

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

By default, PSoC Creator assigns the instance name "Comp_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 "Comp".

Function	Description
void Comp_Start(void)	Initializes the Comparator with default customizer values.
void Comp_Stop(void)	Turn off Comparator
void Comp_SetSpeed(uint8 speed)	Set speed of comparator.
uint8 Comp_ZeroCal(void)	Zero the input offset of comparator.
uint8 Comp_GetCompare(void)	Returns compare result.
void Comp_LoadTrim(uint8 trimVal)	Write a value to the comparator trim register
void Comp_Sleep(void)	Stops the comparator operation and saves the user configuration.
void Comp_Wakeup(void)	Restores and enables the user configuration.
void Comp_SaveConfig(void)	Empty function. Provided for future usage.
void Comp_RestoreConfig(void)	Empty function. Provided for future usage.
void Comp_PwrDwnOverrideEnable(void)	Enables comparator operation in sleep mode
void Comp_PwrDwnOverrideDisable(void)	Disables comparator operation in sleep mode

void Comp_Init(void)	Initialize or Restore default Comparator configuration.
void Comp_Enable(void)	Enable the Comparator.

Global Variables

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

void Comp_Start(void)

Description: This is the preferred method to begin component operation. Comp_Start() sets the initVar variable, calls the Comp_Init() function, and then calls the Comp_Enable() function.

Parameters: None

Return Value: None

Side Effects: If the initVar variable is already set, this function only calls the Comp_Enable() function.

void Comp_Stop(void)

Description: Disable and power down the comparator.

Note This API is not recommended for use on PSoC 3 ES2 and PSoC 5 ES1 silicon. These devices have a defect that causes connections to several analog resources to be unreliable when not powered. The unreliability manifests itself in silent failures (e.g. unpredictably bad results from analog components) when the component utilizing that resource is stopped. It is recommended that this component always be powered up by calling the Comp_Start() routine. Do not call the Comp_Stop() function.

Parameters: None

Return Value: None

Side Effects: None

void Comp_SetSpeed(uint8 speed)

Description: This function selects one of three speed modes for the comparator. The comparator power consumption increases for the faster speed modes.

Parameters: (uint8) speed: Speed parameter, see table below for valid settings.

Speed Options	Description
Comp_LOWPOWER	Use this setting for very low power applications.
Comp_SLOWSPEED	Use this setting for signals requiring response times slower than 80ns
Comp_HIGHSPEED	Use this setting for signals requiring response times faster than 80ns

Return Value: None

Side Effects: None

uint8 Comp_ZeroCal(void)

Description: Performs custom calibration of the input offset to minimize error for a specific set of conditions: comparator reference voltage, supply voltage and operating temperature.
A reference voltage in the range at which the comparator will be used must be applied to the negative input of the comparator while the offset calibration is performed. The comparator component must be configured for Fast or Slow operation when calibration is performed. The calibration process will not work correctly if the comparator is configured in Low Power mode.

Parameters: None

Return Value: (uint8) the value from the comparator trim register after the offset calibration is complete.

This value has the same format as the input parameter for the Comp_LoadTrim() API routine. Refer to the *PSoC3, PSoC5 Technical Reference Manual* for a description of the comparator trim register.

Side Effects: During the calibration procedure the comparator output may behave erratically.
During the calibration procedure the analog routing switches for the comparator positive input will be reconfigured. This reconfiguration may affect the analog signal routing for other components that are connected to the comparator positive input.
When calibration is complete all routing and comparator configuration registers will be restored to the state they were in before calibration occurred.

uint8 Comp_GetCompare(void)

Description: This function returns a non-zero value when the voltage connected to the positive input is greater than the negative input voltage. This value is not affected by the Polarity parameter. This value always reflects a non-inverted state.

Parameters: None

Return Value: (uint8) comparator output state – none zero value when the positive input voltage is greater than the negative input voltage, otherwise the return value is zero.

Side Effects: None

void Comp_LoadTrim(uint8 trimVal)

Description: This function writes a value into the comparator trim register.

Parameters: (uint8) trimVal: Value to be stored in the comparator trim register.
This value has the same format as the parameter returned by the Comp_ZeroCal() API routine. Refer to the *PSoC3, PSoC5 Technical Reference Manual* for a description of the comparator trim register.

Return Value: None

Side Effects: None

void Comp_SaveConfig(void)

Description: This function saves the component configuration. This will save non-retention 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 Comp_Sleep() function.

Parameters: None

Return Value: None

Side Effects: Empty function. Implemented for future usage. No effect by calling this function.

void Comp_RestoreConfig(void)

Description: This function restores the component configuration. This will restore non-retention registers. This function will also restore the component parameter values to what they were prior to calling the Comp_Sleep() function.

Parameters: None

Return Value: None

Side Effects: Empty function. Implemented for future usage. No effect by calling this function.

void Comp_Sleep(void)

Description: This is the preferred routine to prepare the component for sleep. The Comp_Sleep() routine saves the current component state. Then it calls the Comp_Stop() function and calls Comp_SaveConfig() to save the hardware configuration.

Call the Comp_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 Comp_Wakeup(void)

Description: This is the preferred routine to restore the component to the state when Comp_Sleep() was called. The Comp_Wakeup() function calls the Comp_RestoreConfig() function to restore the configuration. If the component was enabled before the Comp_Sleep() function was called, the Comp_Wakeup() function will also re-enable the component.

Parameters: None

Return Value: None

Side Effects: Calling the Comp_Wakeup() function without first calling the Comp_Sleep() or Comp_SaveConfig() function may produce unexpected behavior.

void Comp_PwrDwnOverrideEnable(void)

Description: This is the power down override feature. This function allows the component to stay active during sleep mode.

Parameters: None

Return Value: None

Side Effects: None

void Comp_PwrDwnOverrideDisable(void)

Description: This is the power down override feature. This function allows the comparator to stay inactive during sleep mode.

Parameters: None

Return Value: None

Side Effects: None

void Comp_Init(void)

Description: Initializes or restores the component according to the customizer Configure dialog settings. It is not necessary to call Comp_Init() because the Comp_Start() routine 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 Comp_Enable(void)

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

Parameters: None

Return Value: None

Side Effects: None

Sample Firmware Source Code

PSoC Creator provides numerous 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.

Functional Description

The Comparator is functionally a high-gain high-bandwidth differential amplifier (an opamp with the compensation removed). The comparator is trimmed at the factory to achieve low input offset voltage. It can be trimmed at run-time in the customer's code to achieve improved input offset voltage precision at a specific point. Hysteresis is enabled by adding offsetting currents to the input stage. The nominal hysteresis is 10 mV (33 mV maximum), which is enough to be significantly larger than the sum of any input self noise of the comparator and internal routing interference.

Input offset voltage is normally specified as the absolute value of the difference between the two inputs when the output of the Comparator switches state.

DC and AC Electrical Characteristics

The Comp will operate at all valid supply voltages.

Comparator DC Specifications

Parameter	Description	Conditions	Min	Typ	Max	Units
V_{IOFF}	Input offset voltage in fast mode	Factory trim	–	2.0	10	mV
	Input offset voltage in slow mode	Factory trim	–	2.0	10	mV
V_{IOFF}	Input offset voltage in fast mode[1]	Custom trim	–	1.0	4.0	mV
	Input offset voltage in slow mode[1]	Custom trim	–	1.0	4.0	mV
V_{IOFF}	Input offset voltage in ultra low-power mode		–	12	–	mV
V_{HYST}	Hysteresis	Hysteresis enable mode	–	10	33	mV
V_{ICM}	Input common mode voltage	High current / fast mode	0	–	V_{DDA} – 0.1	V
		Low current / slow mode	0	–	V_{DDA}	V
		Ultra low power mode	0	–	V_{DDA} – 0.9	V
CMRR	Common mode rejection ratio		30	60	–	dB
I_{CMP}	High current mode/fast mode[2]		–	250	400	µA
	Low current mode/slow mode[2]		–	40	100	µA
	Ultra low-power mode[2]		–	6.0	–	µA

[1] The recommended procedure for using a custom trim value for the on-chip comparators can be found in the TRM.

[2] Based on device characterization (Not production tested).

Comparator AC Specifications

Parameter	Description	Conditions	Min	Typ	Max	Units
T_{resp}	Response time, high current mode[1]	50 mV overdrive, measured pin-to-pin	–	80	110	ns
	Response time, low current mode[1]	50 mV overdrive, measured pin-to-pin	–	155	200	ns
	Response time, ultra low-power mode[1]	50 mV overdrive, measured pin-to-pin	–	55	–	µs

[1] Based on device characterization (Not production tested).

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.60	Updated configuration window with an accurate waveform including hysteresis.	Previous configuration window did not provide enough information for ease of use.
	Corrected Hysteresis enable bit setting implementation	The meaning of the enable hysteresis bit was flipped. This has been corrected to correctly enable hysteresis on all versions of silicon
	Added characterization data to datasheet.	
	Minor datasheet edits and updates.	
1.50	Added Sleep/Wakeup and Init/Enable APIs.	To support low power modes, as well as to provide common interfaces to separate control of initialization and enabling of most components.
	Updated Configure dialog with customized interface.	The updated Configure dialog makes it easier to use. There is also a preview of how the component will change based on various selections.
	Added Power Down Override parameter to the Configure dialog.	To allow configuration of Comparator to operate during sleep and hibernate modes.
	Added _PwrDwnOverrideEnable / _PwrDwnOverrideDisable APIs.	To allow the component to stay active / inactive during sleep mode.

© Cypress Semiconductor Corporation, 2009-2010. 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® and CapSense® are registered trademarks, 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.