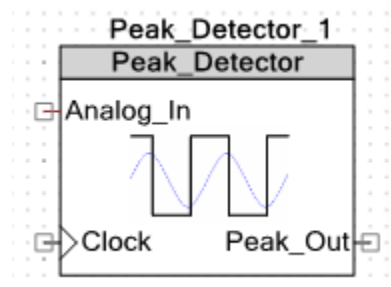


Peak Detector

1.1

General Description

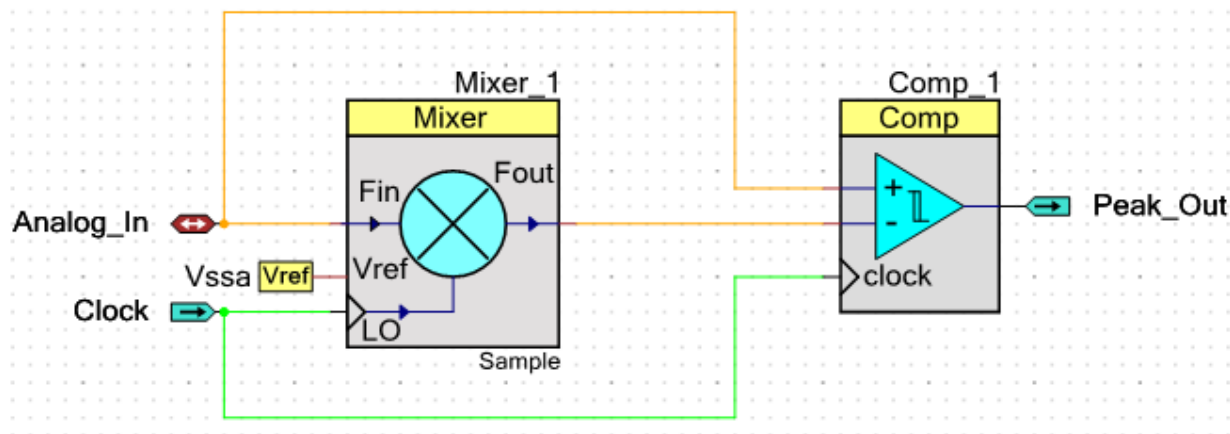
- The peak detector component detects peaks in an input waveform.
- Details on this peak detection method are provided in AN60321.



Quick Start

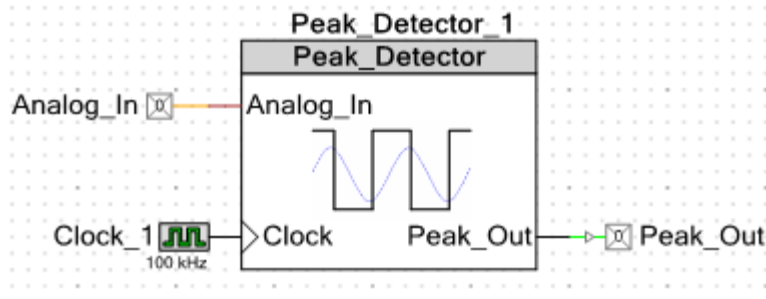
1. To add this component to a project, add a dependency in the project that points to the library project containing the Peak_Detector component. The Peak_Detector component will then be available in the Component Catalog under *Concept -> Peak Detection -> Peak_Detector*.
2. Drag a Peak_Detector component from the Component Catalog onto the design.

Component Schematic



PRELIMINARY

Component Symbol



Input/Output Connections

Analog_In

Analog input signal that will be sampled for peaks.

Clock

Sample clock for the sample and hold component and comparator. The output of the sample and hold is held on the falling edge of the input clock and the comparator will be sampled on the rising edge.

Peak_Out

Digital output signal with the slope and peak information derived from the input waveform. With the Polarity set to 'Non Inverting', the output will be high when the input slope is positive and the output will be low when the input slope is negative.

Parameters and Setup

ClockRange

This is the clock range for the input to the sample & hold component. Set to 'LO Freq less than 100 kHz' if your Clock input is less than 100 kHz. Set to 'LO Freq 100 kHz or greater' if your Clock input is greater than 100 kHz. This parameter is used to determine the appropriate values for the input and feedback resistance of the sample and hold Op-Amp circuit.

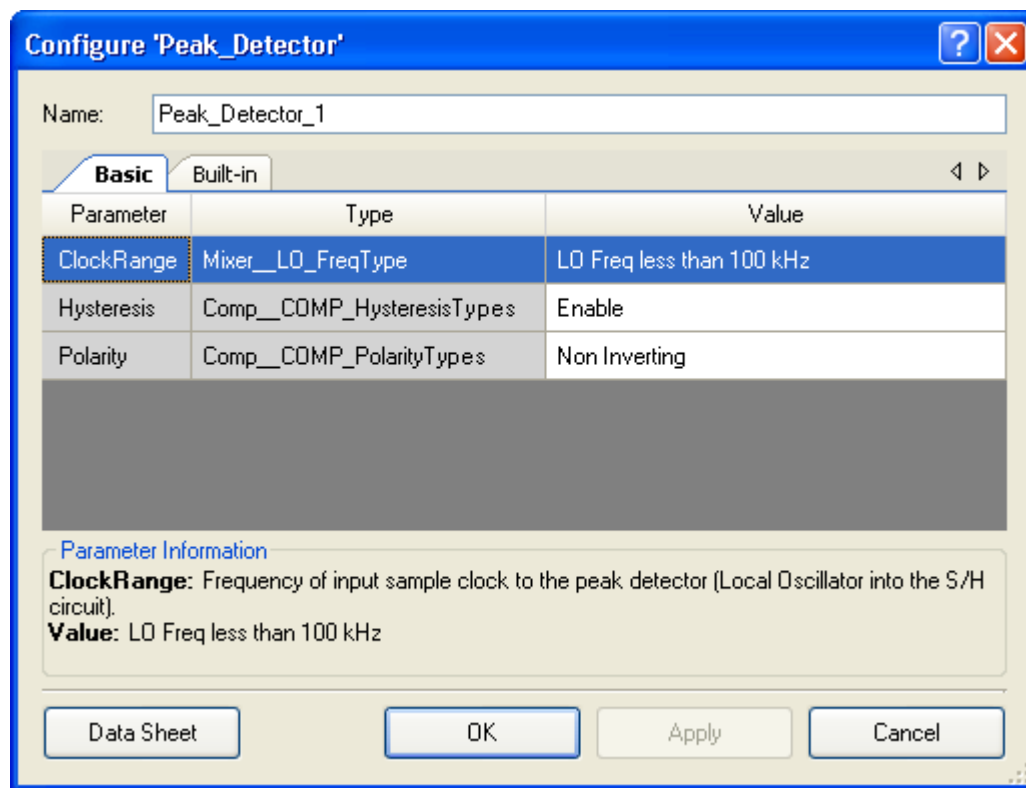
Hysteresis

Enables approximately 10 mV of hysteresis in the comparator. This helps ensure that slowly moving voltages or slightly noisy voltages will not cause the output of the comparator to oscillate when the two inputs are near equal.

Polarity

Sets the polarity of the digital output from the comparator. 'Non Inverting' will output a high signal when the input slope is positive and a low signal when the input slope is negative. Positive peaks

are identified as a falling edge and negative peaks are identified as a rising edge. 'Inverting' will invert this output signal.



Application Programming Interface

Function	Description
<code>void Peak_Detector_Start(void)</code>	Starts the peak detector component.
<code>void Peak_Detector_Stop(void)</code>	Stops the peak detector component.

Note: If additional functionality is needed from either the sample & hold component or the comparator, the original API's for each component are available if they are called directly.

Sample Code

This example will start the Peak Detector component.

```
void main()
{
    Peak_Detector_1_Start();
}
```

Component Changes

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

Version	Description of Changes	Reason for Change
1.0	Initial release of component	
1.1	Internal Mixer component updated from v1.7 to v1.8	Updated to be compliant with PSoC Creator 2.0

© Cypress Semiconductor Corporation, 2011. 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® Creator™, Programmable System-on-Chip™, and PSoC Express™ are trademarks and PSoC® is a registered trademark 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.