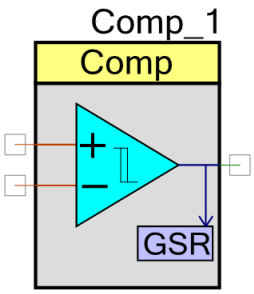
Voltage Comparator (Comp_PDL) 1.0

Features



- Low input offset
- Multiple power/speed levels
- Operates in Deep Sleep mode
- Output can be routed to digital logic blocks or device pins
- Multiple interrupt edge modes

General Description

The Voltage Comparator Component (Comp_PDL) provides a hardware solution to compare two analog input voltages. The output is sampled in software or routed to a digital Component. There are three speed levels to allow optimizing for speed or power consumption. A reference or external voltage can be connected to either input terminals.

The input offset is designed to be less than 1 mV over temperature and voltage. The Component supports a 10 mV input hysteresis that can be enabled.

The comparator can operate in Deep Sleep mode and its interrupt can wake up the device from Deep Sleep mode.

When to Use The Comparator

The comparator gives you a fast comparison between two voltages, compared to using an ADC. Although you can use an ADC with software to compare multiple voltage levels, applications requiring a 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.

Quick Start

1. Drag an Comp_PDL Component from the Component Catalog Cypress/Analog/Comparators folder onto your schematic (placed instance takes the name **Comp_1**).
2. Double-click to open the Configure dialog.
3. Set up the desired settings (Power, Interrupt Edge, Enable 10 mV Hysteresis, Enable Deep Sleep Operation). Rename the instance name to **Comp** for readability.
4. Connect the input/output terminals (see Input/Output Connections).
5. Open the Design-Wide Resources Pin Editor and assign the input and output pins for your design.

6. Build the project in order to verify the correctness of your design. This will add the required PDL modules to the Workspace Explorer, and generate configuration data for the **Comp** instance.
7. In the *main.c* file, initialize the peripheral and start the application:

```
(void) Cy_CTB_OpampInit(Comp_CTB_HW, Comp_COMP_NUM, &
  Comp_compConfig);
Cy_CTB_Enable(Comp_CTB_HW);
```

8. Build the project and program the device.

Input/Output Connections

This section describes the various input and output connections for the Comp_PDL Component.

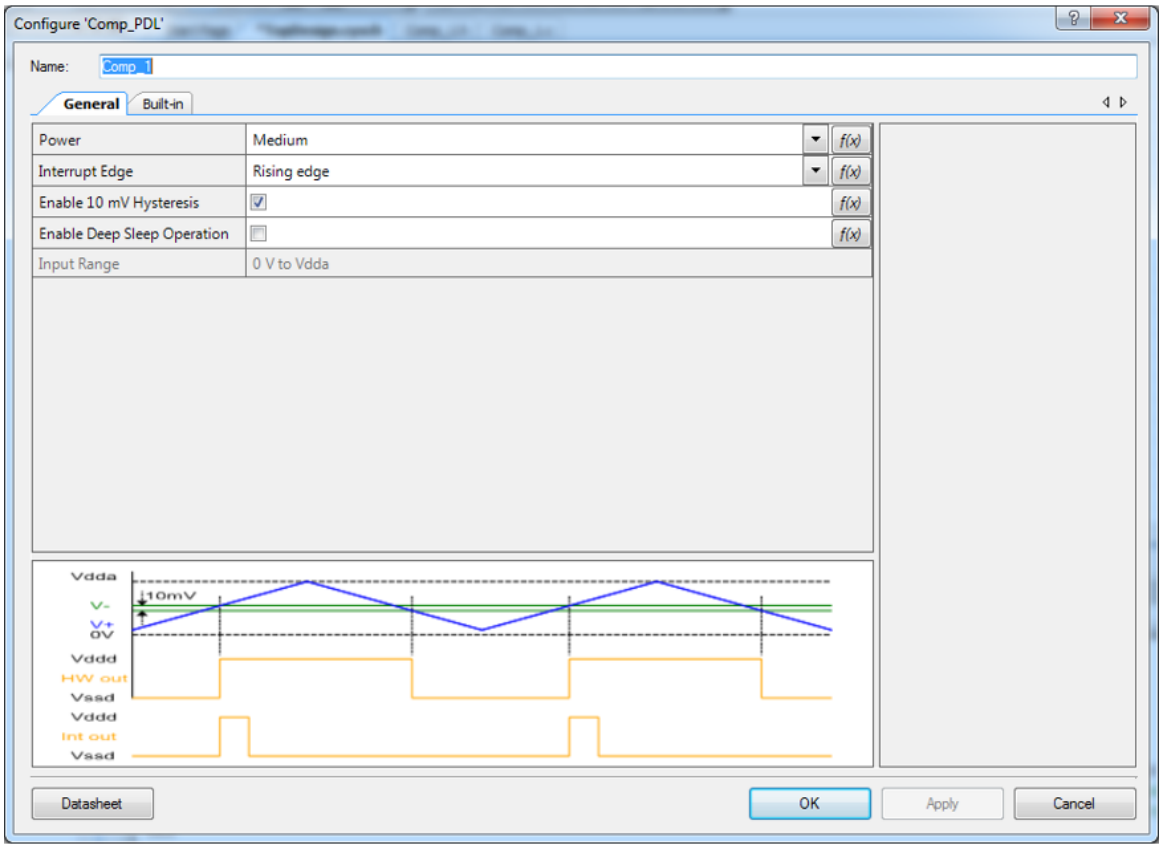
Terminal	I/O Type	Description
Positive Input	Analog Input	This input is usually connected to the voltage that is being compared. This input can be routed from a GPIO or from an internal source.
Negative Input	Analog Input	This input is usually connected to the reference voltage. This input can be routed from a GPIO or from an internal source.
Comparator Out	Digital Output	This is the digital comparison output. This output goes high when the positive input voltage is greater than the negative input voltage. The output can be routed to other Component digital inputs such as interrupts, timers, etc. This digital output is active in Deep Sleep mode. Its value will be latched to the last value in Active mode.

Component Parameters

This section covers the various parameters available from the Component's Configure dialog. Drag a Comp_PDL Component from the Component Catalog onto your schematic, and double-click it to open the dialog.

For any selectable parameter, the option shown here in **bold** is the default.

General Tab



Power

This parameter provides a way to optimize speed versus power consumption.

- Low
- **Medium**
- High

The trade-off is as follows: Fast Speed/High Power, Medium Speed/Medium Power, and Slow Speed/Low Power.

Interrupt Edge

This parameter defines the event that will generate a comparator interrupt.

- Disabled
- **Rising edge**
- Falling edge
- Both edges

See Interrupt Service Routine for more details on handling the comparator interrupt.

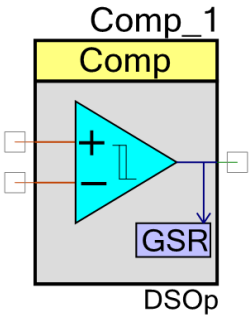
Enable 10 mV Hysteresis

This check box enables approximately 10 mV of hysteresis on the comparator input. This helps to ensure that slowly moving voltages or slightly noisy voltages do not cause the output to oscillate when the two input voltages are nearly equal.

Enable Deep Sleep Operation

This check box enables the comparator hardware to operate while in Deep Sleep mode. It also ensures that only Deep Sleep capable switches are used for analog routing during build time and that the Analog Reference (AREF) block that supplies the reference currents for the opamp is also enabled in Deep Sleep mode.

If this option is enabled, a "DSOp" label will be displayed under the symbol. If two comparators of the same CTB block are used, both must have the same Deep Sleep settings or the PSoC Creator project will not build.



The Cy_CTB_SetDeepSleepMode() function can be used during runtime to disable operation in Deep Sleep mode. However, this function should only be used if this check box is enabled.

Note For correct operation in Deep Sleep mode, Vdda should be at least 2.7 V.

Input Range (display only)

The comparator input range is rail-to-rail if Deep Sleep operation is disabled. If Deep Sleep operation is enabled, the pump is disabled which reduces the input range to 0 V to Vdda - 1.5 V.

- **0 V to Vdda** when Deep Sleep operation disabled
- 0 V to Vdda - 1.5 V when Deep Sleep operation enabled

Application Programming Interface

The Application Programming Interface (API) is provided by the cy_ctb driver module from the PDL. The driver is copied into the "pdl\" directory of the application project after a successful build.

Refer to the PDL documentation for a detailed description of the complete API. To access this document, right-click on the Component symbol on the schematic and choose the **"Open PDL Documentation..."** option in the drop-down menu.

The Component generates the configuration structures and base address described in the Global Variables and Preprocessor Macros sections. Pass the generated data structure and the base address to the associated cy_ctb driver function in the application initialization code to configure the peripheral. Once the peripheral is initialized, the application code can perform run-time changes by referencing the provided base address in the driver API functions.

By default, PSoC Creator assigns the instance name **Comp_1** to the first instance of the Comp_PDL 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 following section is **Comp**.

Global Variables

Global variables used in the Comp_PDL.

Variables

- const cy_stc_ctb_opamp_config_t Comp_compConfig

Variable Documentation

const cy_stc_ctb_opamp_config_t Comp_compConfig

Configuration structure for initializing one comparator using the CTB PDL.

Preprocessor Macros

Preprocessor macros used in the Comp_PDL.

Macros

- #define Comp_CTB_HW
- #define Comp_COMP_NUM

Macro Definition Documentation

#define Comp_CTB_HW

The pointer to the base address of the CTB instance

#define Comp_COMP_NUM

The specific comparator of the CTB instance

Data in Ram

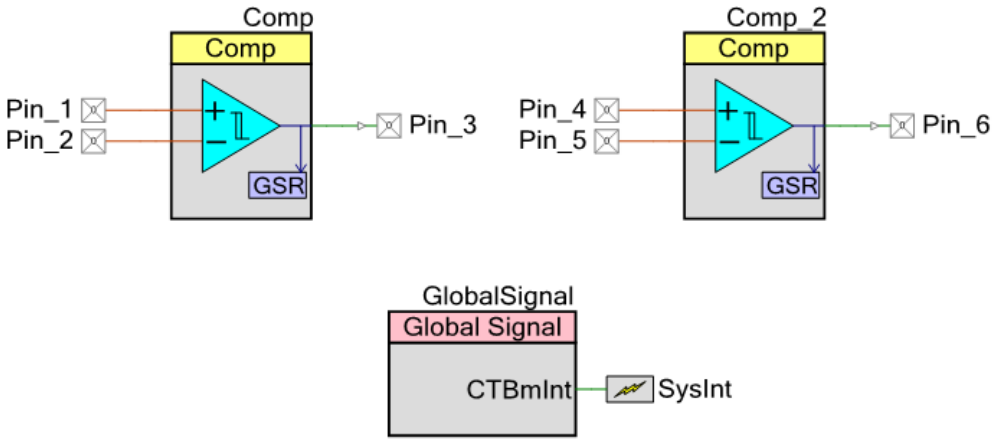
The generated data may be placed in flash memory (const) or RAM. The former is the more memory-efficient choice if you do not wish to modify the configuration data at run-time. Under the Built-In tab of the Configure

dialog, enable or disable the "Config Data in Flash" checkbox to make your selection. The default option is to place the data in flash.

Interrupt Service Routine

There is one global interrupt for all comparators on the device. Therefore to access the comparator interrupt, place the Global Signal Reference (GSR) Component onto the schematic and configure it for the "Combined CTBm interrupt (CTBmInt)" interrupt.

The user routine should check for the interrupt status so that if multiple comparators are available and enabled on the device, the expected interrupt is handled. For the hardware to generate subsequent interrupts, the user routine must clear the interrupt. The ARM CM0+ and CM4 CPUs use bufferable write transfers to the peripherals by default. Because of this, the interrupt register should be read after the write process to ensure the write process is completed. See the ARM website for more information.




```
void Comparator_Interrupt(void)
{
    uint32_t intrStatus;

    intrStatus = Cy_CTB_GetInterruptStatus(Comp_CTB_HW, Comp_COMP_NUM);
    if (0uL != intrStatus)
    {
        /* Clear interrupt followed by dummy read for buffered writes. */
        Cy_CTB_ClearInterrupt(Comp_CTB_HW, Comp_COMP_NUM);

        /* Do something here when Comp interrupt occurs. */
    }

    intrStatus = Cy_CTB_GetInterruptStatus(Comp_2_CTB_HW, Comp_2_COMP_NUM);
    if (0uL != intrStatus)
    {
        /* Clear interrupt followed by dummy read for buffered writes. */
        Cy_CTB_ClearInterrupt(Comp_2_CTB_HW, Comp_2_COMP_NUM);

        /* Do something here when Comp_2 interrupt occurs */
    }
}

int main(void)
```

```
{
    __enable_irq(); /* Enable global interrupts. */

    /* Configure the interrupt and provide the ISR address. */
    (void)Cy_SysInt_Init(&SysInt_cfg, Comparator_Interrupt);

    /* Enable the interrupt. */
    NVIC_EnableIRQ(SysInt_cfg.intrSrc);

    /* Initialize and enable the comparators. */
    (void) Cy_CTB_OpampInit(Comp_CTB_HW, Comp_COMP_NUM, &
      Comp_compConfig);
    Cy_CTB_Enable(Comp_CTB_HW);

    (void) Cy_CTB_OpampInit(Comp_2_CTB_HW, Comp_2_COMP_NUM, &Comp_2_compConfig);
    Cy_CTB_Enable(Comp_2_CTB_HW);

    for(;;)
    {
        /* Put the device into Deep Sleep mode and wait for comparator interrupt to wake it up. */
        Cy_SysPm_DeepSleep(CY_SYSPM_WAIT_FOR_INTERRUPT);
    }
}
```

Interrupt Operation in Deep Sleep Mode

If the Enable Deep Sleep Operation check box is set, the comparator interrupt will wake up the device from Deep Sleep mode. In Deep Sleep mode, the comparator input range is reduced to 0 V to Vdda - 1.5 V therefore a Vdda of at least 2.7 V should be used.

Code Examples and Applications

Code Examples

PSoC Creator provides access to code examples in the Code Example 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 "Code Example" topic in the PSoC Creator Help for more information.

There are also numerous code examples that include schematics and example code available online at the Cypress Code Examples web page.

Application Notes

Cypress provides a number of application notes describing how PSoC can be integrated into your design. You can access the Cypress Application Notes search web page.

Functional Description

The Comp_PDL Component uses the opamp in the CTB configured as a comparator.

The comparator can operate in an ultra low power mode by setting the Opamp Reference Current in the Design-Wide Resources System Editor to 100 nA with a gain bandwidth trade-off. Refer to the device datasheet for the opamp specifications. This selection in the Design-Wide Resources will affect all opamps on the device. In ultra low power mode, the comparators input range is reduced to 0 V to Vdda - 1.5 V.

MISRA Compliance

This section describes the MISRA-C:2004 compliance and deviations for the Component. There are two types of deviations defined:

Deviation Type	Description
Project deviations	Deviations that are applicable for all PSoC Creator Components
Specific deviations	Deviations that are applicable only for this Component

Refer to **PSoC Creator Help** > **Building a PSoC Creator Project** > **Generated Files (PSoC 6)** for information on MISRA compliance and deviations of the files generated by PSoC Creator.

The Comp_PDL Component has no specific deviations.

Resources

The Comp_PDL Component uses a single opamp from the CTB block.

DC and AC Electrical Characteristics

Note Final characterization data for PSoC 6 devices is not available at this time. Once the data is available, the Component datasheet will be updated on the Cypress web site.

Component Changes

This section lists the changes in the Comp_PDL Component from the previous versions.

Version	Description of Changes	Reason for Changes / Impact
1.0.b	Minor datasheet edits	
1.0.a	Production qualified, remove "Prototype" designation	
1.0	Initial version	

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.