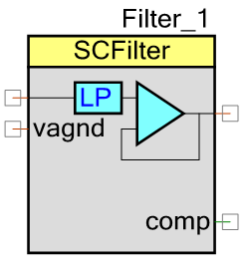
**Please note that Cypress is an Infineon Technologies Company.**
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of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

# Switched-Cap Filter
1.10

## Features




- 4 selectable filter types
  - Low pass
  - High pass
  - Band pass
  - Band stop
- Continuous-time output
- Automatic sample rate for best performance or set by user
- Filter response graphs in frequency and time domains
- Optional input to change the polarity of filter output
- Comparator output as zero-crossing detector

## General Description

The Switched-Cap Filter Component is a 2nd-order inverting filter using the Universal Analog Block (UAB). The filter input is sampled at a specified sample rate resulting in a "stepped" output waveform. User-defined parameters such as the corner frequency, center frequency, and bandwidth are realized as a set of capacitor values. A high bandwidth operational amplifier from the Continuous Time Block (CTB) is used to buffer the filter output.

### When to Use the Component

The filter processes input signals to deliver an output with lower noise, rejecting interference outside of the desired band. This is commonly done for low frequency or DC signals, or band-limited signals for data modulation (e.g., Frequency-shift keying (FSK)). It may be used as a notch filter to eliminate specific interfering frequencies.

**Note** If you need to connect a filtered signal to the Scan_ADC Component, use the filter function within the Scan_ADC Component to maintain proper clock synchronization.

## Quick Start

1. Drag an Switched-Cap Filter Component from the Component Catalog Cypress/Analog folder onto your schematic (placed instance takes the name **Filter_1**).
2. Double-click to open the Configure dialog.
3. Set up the desired settings (Sample rate, Filter type, Gain, etc.). Rename the instance name to **Filter** 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 adds the required modules to the Workspace Explorer, and generates configuration data for the **Filter** instance.
7. In the *main.c* file, initialize the peripheral and start the application:

```
/* Before calling Filter_Start(), the VDDA supply, any voltages used for AGND,
 * and the Pol/mod signal (if used) must be enabled and stable. */

/* If the PVref component is used for AGND, remember to start it. */

/* Filter_Start() initializes and enables the Component.
 * The filter offset will be trimmed when Filter_Start() is called the first time. */
Filter_Start();
```

8. Build the project and program the device.

# Input/Output Connections

This section describes the various input and output connections for the Switched-Cap Filter Component. An asterisk (∗) after the terminal name indicates that the terminal may not be shown on the Component symbol for the conditions listed in the description of that I/O terminal.

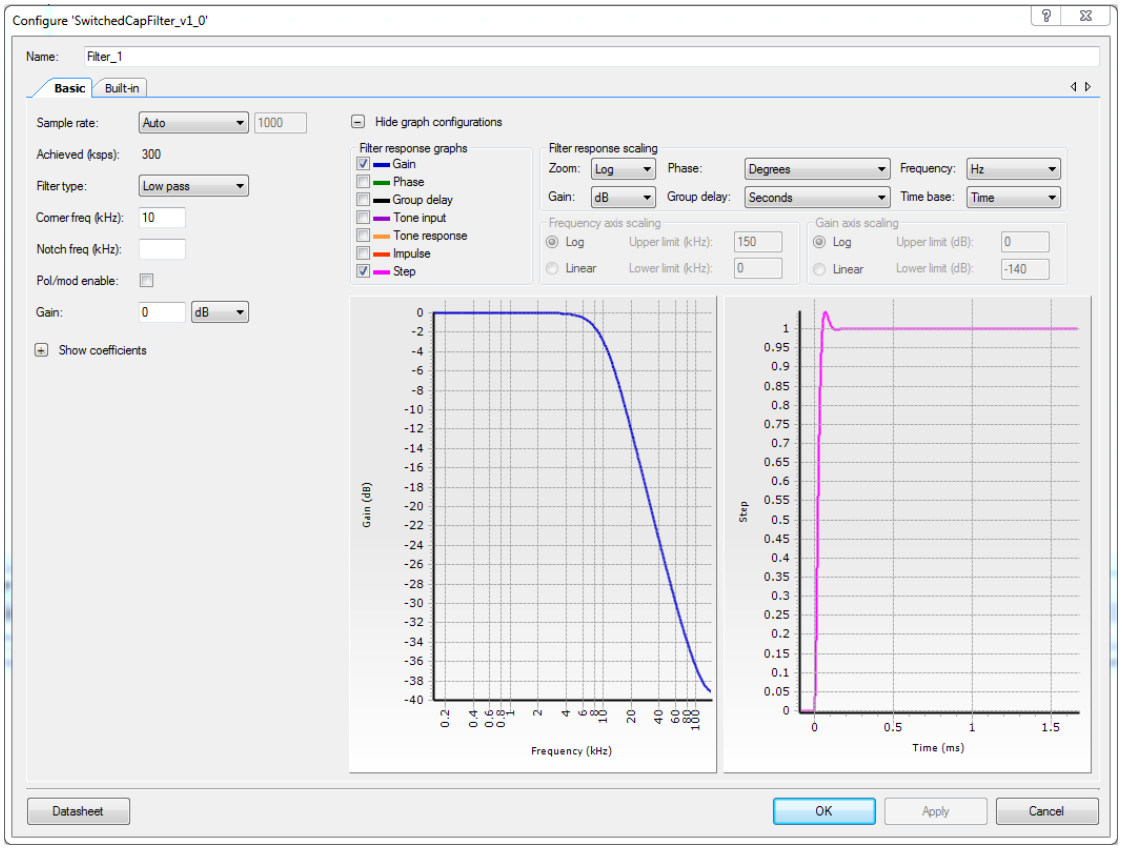
| Terminal | I/O Type | Description |
|---|---|---|
| in | Analog Input | The filter input analog signal. |
| vagnd | Analog Input | The input capacitor is charged between the reference voltage and an internally-generated voltage called Agnd. This is typically set to be equal to half the analog supply voltage, though sometimes a more stable voltage derived from the system bandgap reference voltage is chosen. This connection is typically provided from a PVref Component. |
| pol_mod∗ | Digital Input | This terminal appears when the Pol/mod enable checkbox is checked.<br>A high on this terminal gives a non-inverting filter and a low gives an inverting filter. When the Pol/mod enable checkbox is not checked, the filter is inverting and the pol_mod terminal is not shown.<br>A clock signal (from a PWM, not a clock component) on this pin gives a multiplying mixer. |

| Terminal | I/O Type | Description |
|---|---|---|
| clock∗ | Digital Input | This terminal is for a user defined clock input when the Sample rate selection is set to External. The clock frequency should be between 80 kHz and 8 MHz. The filter sample rate is one-quarter of the clock frequency. |
| out | Analog Output | This is a buffered signal of the filter output. The UAB output is connected to a buffer from a CTB. This buffer is included in the component. |
| comp | Digital Output | This is a comparator output signal that is high when the filter output is above vagnd. The Interrupt Service Routine section shows how to handle the comparator interrupt. |

# Component Parameters

This section covers the various parameters available from the Component's Configure dialog. Drag a Switched-Cap Filter 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. An asterisk (∗) near the parameter name indicates that the parameter will appear only under certain conditions.




The Basic tab contains two groups of parameters:

- Filter Parameters (on the left) determine the filter response

- Display Parameters (on the right) determine which filter response graphs are displayed and how they are scaled.

# Filter Parameters

## Sample rate

Configures the sample rate of the filter.

- **Auto**

  The component auto-selects a sample rate with a target OSR of 30 using an internal clock. A filter's OSR is calculated as:

$$OSR = \frac{Sample\ Rate}{F_{Center/Corner}}$$

- User entry (ksps)

  When selected, enter a desired sample rate in ksps between 20 kHz and 2 MHz. The Component selects a clock divider of an internal clock to achieve a sample rate closest to the entered value.

- External

  When selected, a clock terminal appears on the Component symbol. Provide a clock input and set the frequency. The filter sample rate is one-quarter of the external clock frequency.

  The use case for this parameter is a sweeping filter where the corner or center frequency is adjustable by changing only the clock.

The clock source, either internal or external, must be an integer divider of the high frequency clock (HFCLK) and the divider value must be at least 2. The clock frequency must be between 80 kHz and 8 MHz.

## Achieved (ksps) - display only

This text box displays the achieved sample rate.

## Filter type

Filter type selection. The type is displayed in the schematic symbol in an abbreviated form.

- **Low pass**
- High pass
- Band pass
- Band stop

When Low pass or High pass are selected, enter the corner and notch frequencies. When Band pass or Band stop are selected, enter the center frequency and bandwidth.

## Corner freq (kHz) (Low pass and High pass)

The corner frequency is the -3 dB point.

## Center freq (khz) (Band pass and Band stop)

The center frequency is the arithmetic mean of the upper and lower -3 dB points.

### Notch freq (kHz) (Low pass and High pass)

When this entry field is empty, the notch is not implemented.

### Bandwidth (kHz) (Band pass and Band stop)

For a Band pass, the bandwidth is the difference between the upper and lower -3 dB point.

For a Band stop, the upper and lower points are defined as:

$$F_{Lower} = \frac{\sqrt{Bandwidth^2 + 4F_{center}{}^2} - Bandwidth}{2}$$

$$F_{Upper} = F_{Lower} + Bandwidth$$

### Pol/mod enable

When checked, a terminal appears on the schematic symbol for polarity control. A low on the terminal gives an inverting filter and a high gives a non-inverting filter. When unchecked, the filter is inverting.

This checkbox is only available for the Low pass and Band pass filter types. Also when checked, notch operation is disabled.

### Gain

The default gain value is 0 dB. The linear gain should be between 0.1 to 10. The dB gain should be between -20 dB to 20 dB.

If the achieved gain is not within 5 % of the user-specified value, an info message is displayed indicating so.

### Gain unit

The gain unit selection, either dB or Linear, will update the user-defined gain value.

- **dB**
- Linear

### Show/Hide coefficients

This button will toggle a display text box that contains the final filter coefficients based on the following transfer function:

$$H(z) = \frac{Num_0 + Num_1 z^{-1} + Num_2 z^{-2}}{1 + Den_1 z^{-1} + Den_2 z^{-2}}$$

The coefficients are displayed in the order of Num0, Num1, Num2, Den1, and Den2.

## Display Parameters

Display parameters only affect the way the filter response is presented in the Configure dialog. They have no effect on the code generation or filter settings.

### Hide/Show graph configurations

This button toggles the display of the parameter selections. When hidden, the graphs automatically resize to fill up the space to make them easier to read.

When many subplots are shown in the graph area, the plot axes may not be numbered as expected because of limitations in the automatic plot routines.

The plots are divided into two subplot areas, for frequency and time parameters. Right-click on either subplot to copy the image to the clipboard.

### Filter response graphs

- **Gain**

  Displays the amplitude of the overall filter response over frequency.

- Phase

  Displays the phase shift of the overall filter response over frequency.

- Group delay

  Displays the group delay of the overall filter response over frequency.

- Tone input

  Displays a sine wave signal at the center or cutoff frequency to be used as the input to the filter.

- Tone response

  Displays the filter's response to the Tone input sine wave.

- Impulse

  Displays the filter's response to a positive-going single-sample impulse.

- **Step**

  Displays the filter's response to a postive going unit step function.

### Filter response scaling: Zoom

- **Log**

  Provides a view of the filter's response with a logarithmic frequency scale from DC to the Nyquist frequency, Fs/2.

- Linear

  Provides a view of the filter's response with a linear frequency scale from DC to the Nyquist frequency, Fs/2.

- Custom

  Enables custom control of the minimum and maximum limits of gain and frequency. The frequency and gain axes can be set in both linear and logarithmic modes.

  When selected, the Frequency axis scaling and Gain axis scaling fields are enabled.

### Filter response scaling: Gain

Display the amplitude of the Gain response graph in a dB or linear scale.

- **dB**
- Linear

### Filter response scaling: Phase

Display the phase of the Phase response graph in degrees or radians.

- **Degrees**
- Radians

### Filter response scaling: Group delay

Display the delay of the Group delay response graph in microseconds or as a number of samples.

- **Seconds**
- Sample counts

### Filter response scaling: Frequency

Display the frequency axis of all frequency response graphs in kHz or as a fraction of the sample rate, Fs.

- **Hz**
- Fraction of Fs

### Filter response scaling: Time base

Display the time axis of all time response graphs in milliseconds or in sample counts.

- **Time**
- Samples

### Frequency axis scaling: Log/Linear selection

This option is valid only when Custom zoom is selected.

Sets the Frequency response x-axis to log or linear scale.

- **Log**
- Linear

### Frequency axis scaling: Upper limit

This option is valid only when Custom zoom is selected.

Sets the upper limit for the x-axis of the frequency response graph. The maximum upper limit should be less than or equal to half of the sample rate.

Default is **150 kHz**.

### Frequency axis scaling: Lower limit

This option is valid only when Custom zoom is selected.

Sets the lower limit for the x-axis of the frequency response graph. The lower limit should be less than the upper limit value and should not be less than zero.

Default is **0 kHz**.

### Gain axis scaling: Log/Linear selection

This option is valid only when Custom zoom is selected.

Sets the Gain response x-axis to log or linear scale.

- **Log**
- Linear

### Gain axis scaling: Upper limit

This option is valid only when Custom zoom is selected.

Sets the upper limit for gain response in either dB or linear scale.

Default is **0 dB**.

### Gain axis scaling: Lower limit

This option is valid only when Custom zoom is selected.

Sets the lower limit for gain response in either dB or linear scale. The lower limit should be less than the upper limit value.

Default is **-140 dB**.

# Application Programming Interface

By default, PSoC Creator assigns the instance name **Filter_1** to the first instance of the Switched-Cap Filter 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 **Filter**.

- General APIs

  *General APIs are used for run-time configuration of the component during active power mode.*
- Power Management APIs

  *Power management APIs perform the necessary configurations to the components to prepare it for entering low power modes.*

## General APIs

General APIs are used for run-time configuration of the component during active power mode.

These include starting the, stopping, reading from registers, and writing to registers.

### Functions

- void Filter_Start (void)
- void Filter_Stop (void)
- void Filter_ClearInterrupt (void)
- uint32 Filter_GetInterruptStatus (void)
- uint32 Filter_GetCompStatus (void)
- void Filter_Init (void)
- void Filter_Enable (void)
- void Filter_TrimFilterVos (void)

### Function Documentation

*void Filter_Start ( void )*

Start the Component. Must be called before using the Component.

Invokes Filter_Init() (the first time it is called only) and Filter_Enable(). After this function call, the Component is enabled and operating. This is the preferred method to begin Component operation.

**Note** This function also calls Filter_TrimFilterVos() the first time it is called to reduce the filter offset. This requires that the Agnd be applied to the component before calling this function the first time. If the Pol/Mod bit is used, apply the desired the Pol/Mod level as well before calling Filter_Start().

*void Filter_Stop ( void )*

Stop the Component.

Power down the UAB and output buffer blocks.

*void Filter_ClearInterrupt ( void )*

Clear the comparator interrupt so that subsequent interrupts can be triggered and handled.

### *uint32 Filter_GetInterruptStatus ( void )*

Return the comparator interrupt status.

**Returns**

- 0: Comparator did not trigger
- 1: Comparator interrupt triggered

### *uint32 Filter_GetCompStatus ( void )*

Return the comparator output status.

**Returns**

- 0: Comparator output is low
- 1: Comparator output is high

### *void Filter_Init ( void )*

Initialize the Component according to parameters defined in the customizer.

It is not necessary to call Filter_Init() because the Filter_Start() API calls this function and is the preferred method to begin the Component operation.

### *void Filter_Enable ( void )*

Enable the Component.

It is not necessary to call Filter_Enable() because the Filter_Start() API calls this function and is the preferred method to begin the Component operation.

### *void Filter_TrimFilterVos ( void )*

Run an algorithm to reduce voltage offset using the UAB's Agnd buffer and Opamp trim.

During trimming, the filter inputs are disconnected from the UAB block, the non-inverting input of both UAB opamps are connected to Agnd, and the output buffer is configured as a comparator.

Trimming is done by comparing the filter output with Agnd. The algorithm steps through the Opamp trim codes first and then the Agnd trim codes to find where the filter output crosses Agnd. For each trim code, a blocking delay is used to allow the filter output to settle before reading the comparator status. This delay is equivalent to $9\tau$, where tau is the filter time constant:

$$\tau = \frac{1}{2 \times \pi \times F}$$

For Band pass and Band stop, F is the bandwidth. For Low pass and High pass, F is the corner frequency. The algorithm can check up to 40 different trim codes, but it will typically complete in much less time.

Once trimming is complete, the UAB and output buffer are restored to their original configuration.

**Note** Agnd must be applied to the component before calling this function. If the Pol/Mod bit is used, set the Pol/Mod level to the desired value before calling this function.

## Power Management APIs

Power management APIs perform the necessary configurations to the components to prepare it for entering low power modes.

These APIs must be used if the intent is to put the chip to sleep, then to continue the component operation when it comes back to active power mode.

### Functions

- void Filter_Sleep (void)
- void Filter_Wakeup (void)

### Function Documentation

*void Filter_Sleep ( void )*

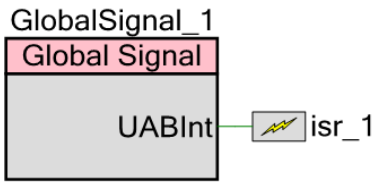
Disable block operation and saves its configuration. Should be called just prior to entering sleep.

*void Filter_Wakeup ( void )*

Enable block operation and restores its configuration. Should be called just after awaking from sleep.

## Interrupt Service Routine

To access the UAB interrupt, place the Global Signal Reference Component onto the schematic and configure it for the "Combined UAB interrupt (UABInt)" interrupt.




As with any interrupt, initialize and enable the interrupt and assign the user routine to be called.

```
    CyGlobalIntEnable; /* Enable global interrupts. */

    /* Enable interrupt and assign the user routine to be called. */
    isr_1_StartEx(Filter_Interrupt);
```

The user routine should check for the interrupt status in the case multiple UABs are available on the device. For the hardware to generate subsequent interrupts, the user routine must clear the interrupt.

```
void Filter_Interrupt(void)
{
    uint32 compStatus;

    if (1UL == Filter_GetInterruptStatus())
    {
        /* Do something here when an interrupt occurs such as reading the comparator output status. */
```

```
        compStatus = Filter_GetCompStatus();
        /* Clear the interrupt so subsequent interrupts can be serviced properly. */
        Filter_ClearInterrupt();
    }
}
```

## Code Examples and Applications

### Code Examples

PSoC Creator provides access to code examples in the Find 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 "Find 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.

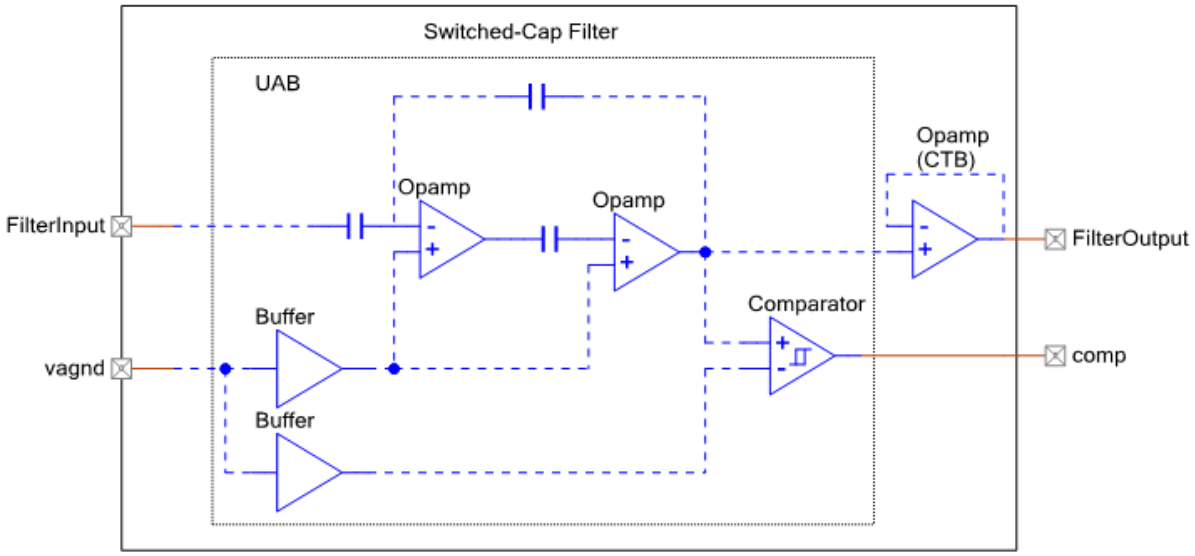
### Application Notes

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 Switched-Cap Filter Component implements two types of biquad filters, the High-Q and Low-Q Gregorian and Temes (G-T) Biquad. The G-T biquad is generally regarded as having low sensitivity and small non-ideal effects in their filter realizations.

As shown in the following block diagram, the filter is implemented using the opamps, buffers, and various switched capacitor arrays in the UAB. A comparator in the UAB is used to compare the filter output with the analog ground input to provide a zero-crossing detector.



# 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 the *System Reference Guide* for information on MISRA compliance and deviations of the files generated by PSoC Creator.

The Switched-Cap Filter Component has no specific deviations.

# Resources

The Switched-Cap Filter Component uses the following device resources:

| Resource Type | Used |
|---|---|
| UAB | 1 |
| Opamp | 1 |

# DC and AC Electrical Characteristics (Preliminary)

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

# Component Errata

This section lists known problems with the Component.

| Cypress ID | Component Version | Problem | Workaround |
|---|---|---|---|
| 254437 | All | For parts marked ES (Engineering Sample), when SYSCLK is divided from HFCLK, the component will not function as expected. | Use a SYSCLK divider value of 1 in the Configure System Clocks -> High Frequency Clocks tab. |

# Component Changes

This section lists the changes in the Switched-Cap Filter Component from the previous versions.

| Version | Description of Changes | Reason for Changes / Impact |
|---|---|---|
| 1.10 | Clock all statically open and closed switches in the UAB. | Static switches in the UAB introduce a tone at Fs/4. By clocking them, SNR performance is improved. |

| Version | Description of Changes | Reason for Changes / Impact |
|---|---|---|
| 1.0 | Initial version | |

© Cypress Semiconductor Corporation, 2018. 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.