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.

THIS DOCUMENT IS OBSOLETE

Spec No: 001-97601

Spec Title: CE97601 - IMPROVING THE ACCURACY OF THE PSOC(R) 4 INTERNAL MAIN OSCILLATOR

Replaced by: None

CE97601 - Improving the Accuracy of the PSoC® 4 Internal Main Oscillator

Objective

This example demonstrates how to trim the PSoC 4 internal main oscillator (IMO) to improve its accuracy.

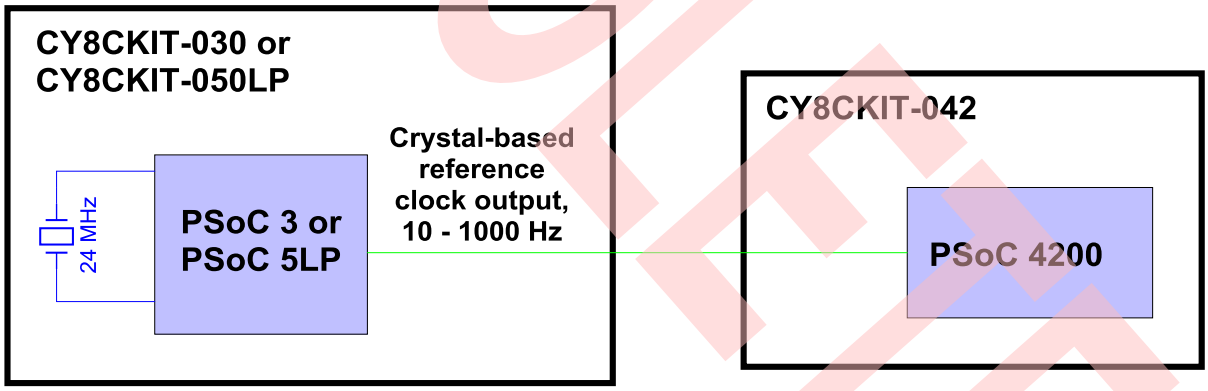
Overview

The PSoC 4 IMO accuracy specification in the device datasheets is ±2%. This may be unacceptable, for example in communication channels such as UART. This code example demonstrates adjusting the frequency of the IMO to make it more accurate. A PSoC 4 trim register is adjusted to change the IMO frequency; see the PSoC 4 Registers Technical Reference Manual (TRM) for details.

The IMO is the highest-accuracy clock source in PSoC 4. Therefore, to trim the IMO, a higher accuracy *external* clock source is required. In this example, a reference clock is provided from a source external to the PSoC 4, as Figure 1 shows.

A separate PSoC Creator project provides a clock based on the PSoC 3 or PSoC 5LP external crystal oscillator (MHzECO). A crystal typically has an accuracy on the order of 100 parts per million (ppm), which is much better than that of the PSoC 4 IMO.

Figure 1. Using an External Crystal-Based Reference Clock to Trim the PSoC 4 IMO



Note This code example is designed for the PSoC 4200 device and associated CY8CKIT-042 kit. It is easily portable to other PSoC 4 devices and kits. See Porting to Other PSoC 4 Devices and Kits for details.

Requirements

Tool: PSoC Creator 3.2.

Programming Language: C - GCC 4.8.4 or MDK for the PSoC 4 trim project
DP8051 Keil 9.51 for the PSoC 3 reference clock project
GCC 4.8.4 or MDK for the PSoC 5LP reference clock project

Associated Parts: All PSoC 4200 parts. See Porting to Other PSoC 4 Devices and Kits.

Related Hardware: Target PSoC 4 kit: CY8CKIT-042
Crystal-based reference clock kits: CY8CKIT-030, CY8CKIT-050

OBSOLETE

Design

The PSoC 4 IMO trim design uses only a Timer Component (TCPWM-based), as Figure 2 shows. The Timer's clock input can be connected to any high-frequency clock that is derived from the IMO. The other Timer inputs – reload and capture – are connected to the reference clock.

Figure 2. PSoC 4 IMO Trim Design Schematic

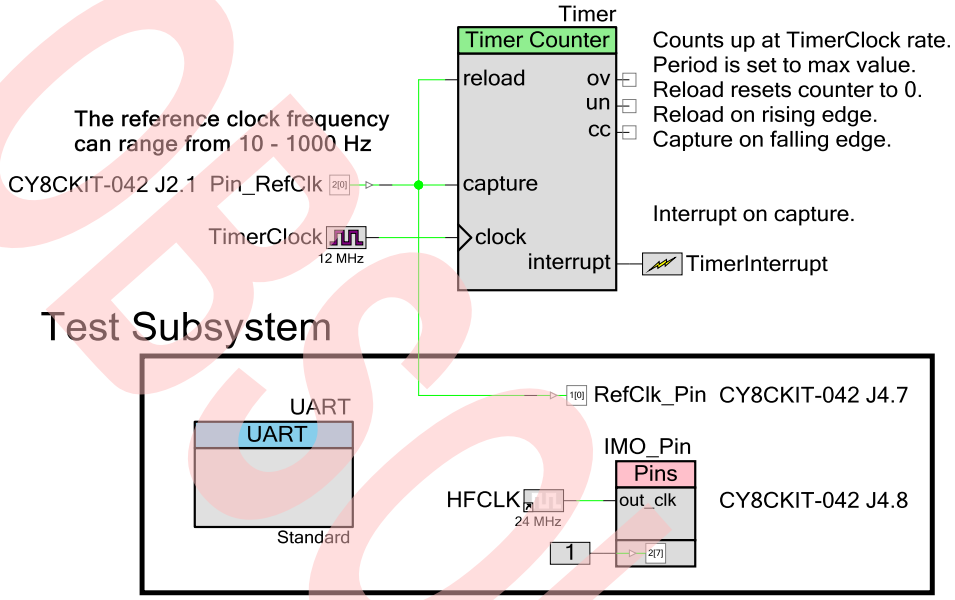
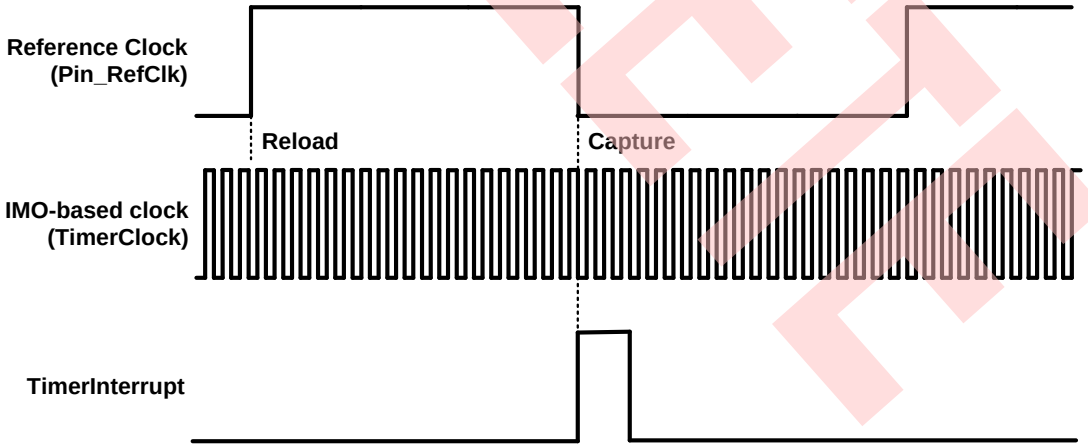


Figure 3 shows the related timing diagram. Note that counting takes place only during half of the reference clock.

Figure 3. Timing Diagram Template



On reload, the counter in the Timer Component is initialized with 0. The counter counts up (see Figure 5). On capture, the counter current value is copied to a capture register.

On a timer capture event, an interrupt is generated. Firmware reads the captured value and compares it to an expected target value. The code then changes the IMO trim control register, `CYREG_CLK_IMO_TRIM1`, accordingly. The firmware also detects when no more trimming is needed, and ends the trim operation.

Test pins and a UART are included to facilitate performance monitoring, as Figure 2 shows. Note that the IMO output cannot be monitored directly. Instead, HFCLK, which is derived from the IMO, is monitored on IMO_Pin.

OBSOLETE

Reference Clock

A second project is included to generate a reference clock to test the trim project (see Figure 1). The reference clock frequency can be adjusted; this simulates the target IMO clock drifting from the ideal value.

The CY8CKIT-030 and CY8CKIT-050 kits include a 24-MHz crystal, which is used as the basis for the accurate reference clock. The crystal is connected to PSoC 3 or PSoC 5LP pins P15[1:0], which are dedicated to the MHzECO block. For more information, see a device datasheet.

The reference clock is generated using a PWM Component, as Figure 4 shows. The PWM period, or reference clock frequency, is chosen such that the target timer value is half of a 16-bit timer max value, or 65,536 / 2 = 32,768. That is, if the target IMO frequency is at precisely the desired value, the timer capture value (Figure 2) is 32,768.

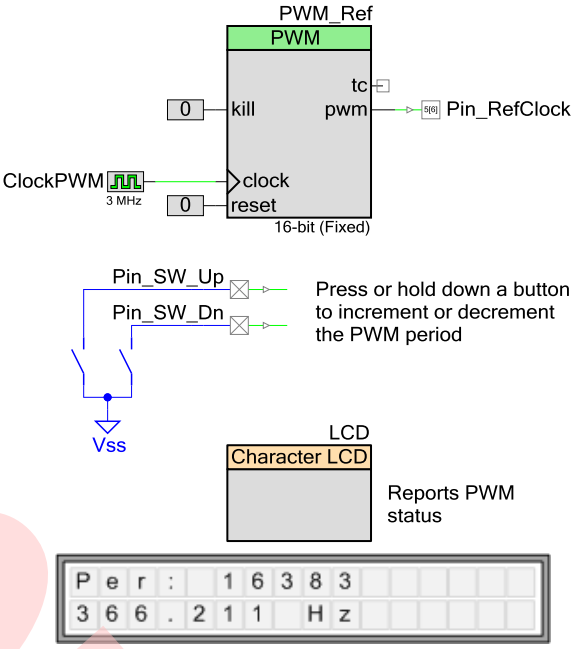
The kit LCD displays the PWM period and reference clock frequency. You can use kit buttons to adjust the reference clock frequency. The PWM duty cycle is always 50%.

Macros in both projects' firmware make it easy to adapt the design to different values of:

- ClockPWM in the reference clock design (Figure 4)
- TimerClock in the target design (Figure 2)

See Software Setup below.

Figure 4. Design for Reference Clock



Design Considerations

- The trim design can be adapted to work in any system that provides an accurate clock that can be connected to a spare PSoC 4 pin. The reference clock can be connected to any PSoC 4 pin – the PSoC 4 dedicated external clock pin is not necessarily needed and is not used in this design.
- The firmware in the trim design can be adapted to be encapsulated in a function, to be called as needed by the larger system.
- The trim design includes a Tx-only UART (see Figure 2) which can be used to monitor trimming progress. The UART clock is derived from the IMO. Note that if the IMO is trimmed too far from the ideal value, UART data may become corrupted due to bit rate mismatch.
- It is possible to trim the IMO such that the device operates above its maximum rated frequency. This should be avoided.

Hardware Setup

- Connect the two kits to a PC, using the USB cables.
- Make a wire connection between the reference clock kit's output pin and the target kit's input pin (see Figure 1). Make sure that the voltage levels – i.e., 5 V and 3.3 V – are the same in both kits.
- You can use the UART in the target design (see Figure 2) through the kit's USB connection. To set up the kit to do this, see the kit's User Guide.

Software Setup

- Change the values of the macros FTGTTMRCLK, FPWMCLK, FTMRCLK, and FREF, in both designs, as needed. The comments in the source code describe these macros and their usage.

PRELIMINARY

Components

Table 1 lists the PSoC Creator Components used in this example, as well as the hardware resources used by each.

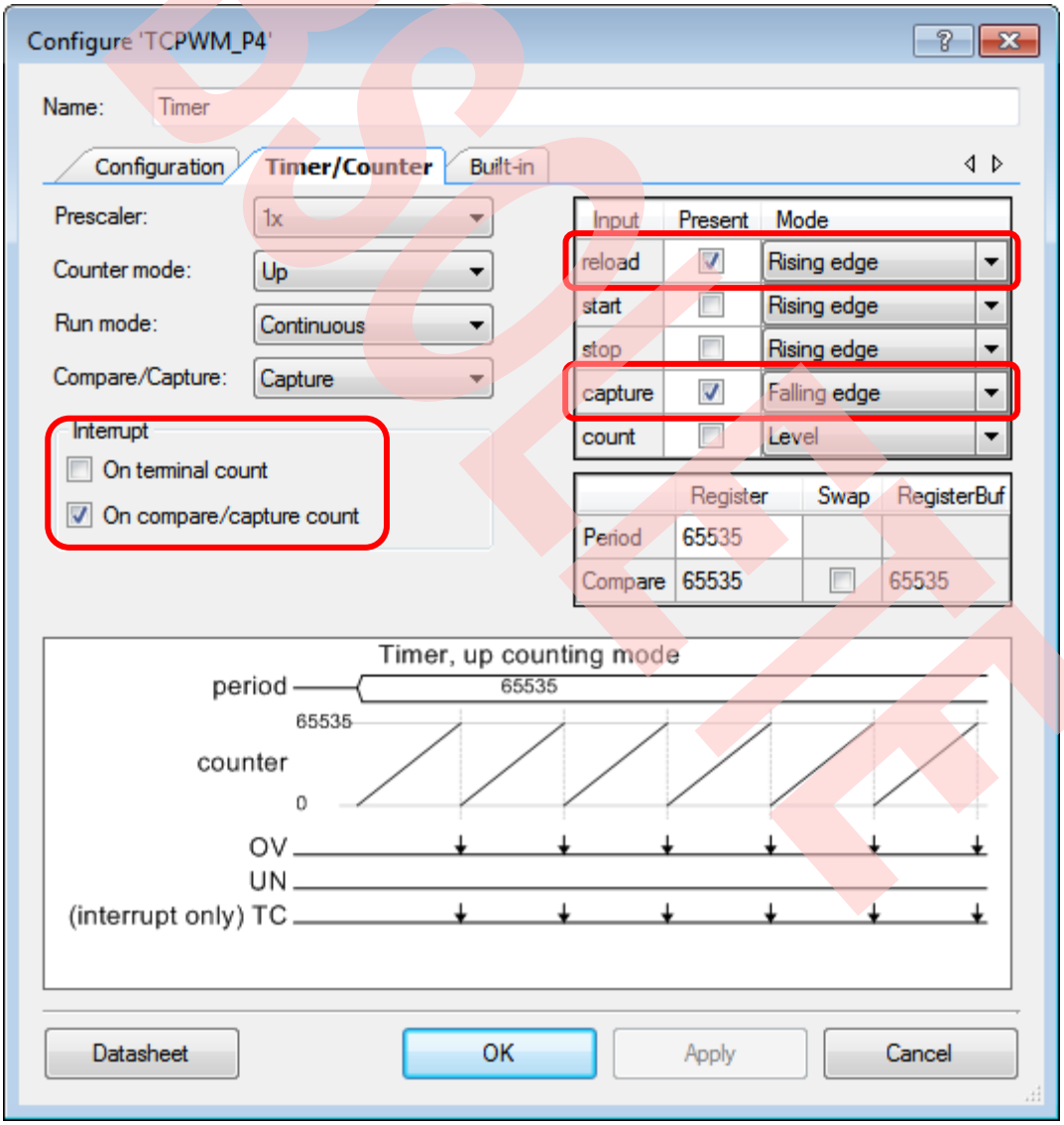
Table 1. List of PSoC Creator Components

Component	Hardware Resources
Timer	1 TCPWM
TimerClock	1 clock divider
Pin_RefClk	1 pin

Parameter Settings

The TimerClock frequency is set to the value shown in Figure 2. Figure 5 shows the changed settings for the Timer Component.

Figure 5. Trim Timer Configuration



OBSOLETE

Operation

1. Set up the system as described in Hardware Setup.
2. Build the reference clock project, and program it into a CY8CKIT-030 or CY8CKIT-050 kit.
   a. Using an oscilloscope or frequency counter, confirm that the output clock frequency is the expected value, e.g., 183.105 Hz.
   b. Confirm that the kit LCD displays the correct PWM period value, and twice the output frequency. For example:
      ```
      Per.: 16383
      366.211 Hz
      ```
3. Build the trim project, and program it into a CY8CKIT-042 kit.
4. Make a wire connection between the two kits – from the reference clock kit's output pin to the target kit's input pin.
   a. Using an oscilloscope or frequency counter on the PSoC 4 kit, confirm that the reference clock frequency, available on the RefClk_Pin test pin (see Figure 2) is the expected value, e.g., 183.105 Hz.
   b. (optional) Monitor the value of the target HFCLK, available on the IMO_Pin test pin (see Figure 2).
5. Bring up a UART terminal program such as HyperTerminal. Make a connection to the COMx port produced by the USB connection to the CY8CKIT-042 kit. Set the baud rate and other parameters to the appropriate values.
6. Reset the CY8CKIT-042 kit by pressing the SW1 button on the kit board. Confirm that something similar to the following is displayed on the terminal program:

   ```
   Starting IMO trim operation
   Target = 32767
   Trim value = 0x6b, capture value = 32811
   Trim value = 0x6a, capture value = 32737
   Done, trim = 0x6a
   ```

   Repeating this step may show small variances in the capture values – this is acceptable.
7. Press one of the buttons on the reference clock kit. Confirm that the period and frequency change on the LCD display. Repeat step 6 and confirm that different data is displayed on the terminal program:

   ```
   Starting IMO trim operation
   Target = 32767
   Trim value = 0x6b, capture value = 32488
   Trim value = 0x6c, capture value = 32586
   Trim value = 0x6d, capture value = 32657
   Trim value = 0x6e, capture value = 32737
   Trim value = 0x6f, capture value = 32813
   Trim value = 0x6e, capture value = 32739
   Done, trim = 0x6e
   ```

8. Note that adjusting the reference clock too far from the norm causes UART data corruption:

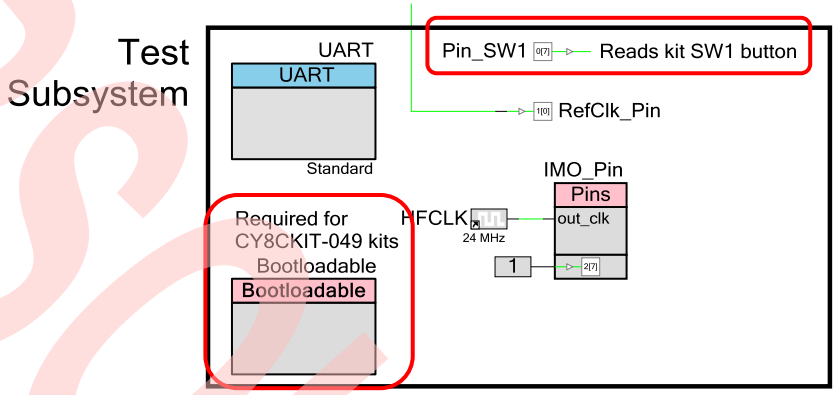
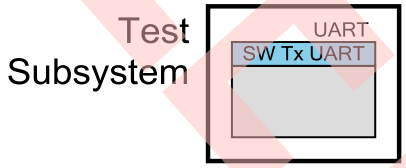
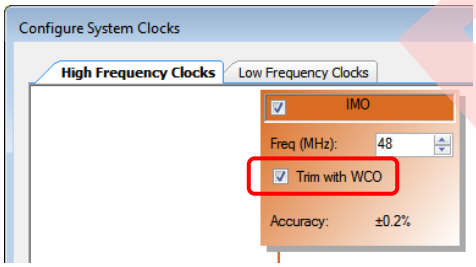
   ```
   Trim value = 0x58, capture value = 34958
   Trim value = 0x57, capture value = 34888
   Ôòéí öáìõå = 0ø56, ãapture value = 34810
   Trií valuå = 0x55, caðôure öalue = 34730
   Tòií váluå =`0ø54, ãapôuòå öalõe = 34646
   Tòií váluå =`0ø54, ãapôuòå öalõe = 34646
   Ôòéí öáìõå`} 0øu2,`ãápôõòe`öálõå = stty6
   ```

OBSOLETE

Porting to Other PSoC 4 Devices and Kits

This code example is designed for the PSoC 4200 device and associated CY8CKIT-042 kit (see Figure 2). The design is easily portable to other PSoC 4 devices and kits. However, some minor design changes must be made, typically in the Test Subsystem, due to differences in the devices and kits. Table 2 shows the specific changes that need to be made.

Table 2. Project Changes for Other PSoC 4 Devices and Kits

Device / Kit and Description	Project Changes
PSoC 4200 / CY8CKIT-049-42xx PSoC 4100 / CY8CKIT-049-41xx • Reset pin is not available on kit board. • The PSoC 4100 / 4200 devices in these kits have a factory-installed bootloader. Install applications using this bootloader, as opposed to programming through JTAG / SWD. See the kit guide for details.	• Add a pin to read the kit button. Change the code to do a software reset when the button is pressed. • Add a Bootloadable Component. Link it to the bootloader *.hex* file that is provided with the kit. See the Bootloadable Component datasheet and kit guide for details. 
PSoC 4000 / CY8CKIT-040 • Signal and clock routing to pins is not available. • Hardware UART is not available. • P3.0 is shared with SWD on a device pin. This pin is hard-wired on the kit board to the USB-UART Bridge Rx input.	• Delete the test pins from the Test Subsystem. The Test Subsystem now has only a UART. • Replace the UART Component with a SW_Tx_UART Component. See the kit guide for UART setup instructions.  • In the DWR window, System tab, change the Debug Select setting from SWD to GPIO. This enables P3.0 to be used as a GPIO.
PSoC 4200M / CY8CKIT-044 PSoC 4100M • Watch crystal oscillator (WCO) is included, as well as IMO trim to WCO.	• This code example may not be needed if your design includes a watch crystal. 

Related Documents

Table 3 lists all relevant application notes, code examples, knowledge base articles, device datasheets, and Component datasheets.

Table 3. Related Documents

Application Notes		
AN80248	PSoC 3, PSoC 5LP Improving the Accuracy of Internal Oscillators	Two Components developed for this purpose greatly simplify the process of calibrating the ILO and IMO with respect to a reference time base.
Code Examples		
CE95329	Compensation of ILO Trimming with PSoC 4	Demonstrates the compensation operation of the ILO Trim Component for PSoC 4
CE95328	1 kHz ILO Trimming with PSoC 3/5LP	Demonstrates 1-kHz ILO trimming for PSoC 3 and PSoC 5LP
PSoC Creator Component Datasheets		
PSoC 4 TCPWM	PSoC 4 counter, timer, PWM (TCPWM)	
Clock	Creates local clocks, and allows connection to system and design-wide clocks	
Pins	Controls interface with physical I/O port pins	
Device Documentation		
PSoC 3 Datasheets	PSoC 3 Technical Reference Manuals	
PSoC 4 Datasheets	PSoC 4 Technical Reference Manuals	
PSoC 5LP Datasheets	PSoC 5LP Technical Reference Manuals	
Development Kit (DVK) Documentation		
PSoC 3 and PSoC 5LP Kits		
PSoC 4 Kits		

OBSOLETE

Document History

Document Title: CE97601 - Improving the Accuracy of the PSoC® 4 Internal Main Oscillator

Document Number: 001-97601

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	4810682	MKEA	06/30/2015	New code example.
*A	5742432	AESATMP9	05/19/2017	Updated logo and copyright.
*B	6313288	MKEA	09/18/2018	Obsolete document. Completing Sunset Review.

OBSOLETE

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

©Cypress Semiconductor Corporation, 2015-2018. 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.

OBSOLETE