CE219322 – PSoC® 3 and PSoC 5LP Internal Main Oscillator Trim

Objective

These examples demonstrate how to trim the PSoC® 3 or PSoC 5LP internal main oscillator (IMO) to improve its accuracy.

Overview

The PSoC 3 and PSoC 5LP IMO accuracy specification in the device datasheets is ±1% at a nominal 3 MHz. The IMO frequency may drift over time and temperature; see device datasheet specifications. For applications that require high IMO accuracy, you may need to periodically adjust, or trim, the IMO frequency. This code example demonstrates how to do so.

To trim the IMO, a higher-accuracy clock source is required. In this example, a reference clock is provided from a 32.768-kHz crystal, through the PSoC kHz external crystal oscillator (kHzECO). The hardware portion of the design counts the number of edges of an IMO-based clock relative to the reference clock. Firmware compares the result to an expected value and adjusts the trim register up or down accordingly.

Two projects are included; one uses interrupt mode and the other uses polled mode.

Requirements

Tool: PSoC Creator™ 4.1

Programming Language: C: GCC 5.4-1026-q2-update or MDK/armcc for PSoC 5LP; DP 8051 Keil 9.5.1 for PSoC 3

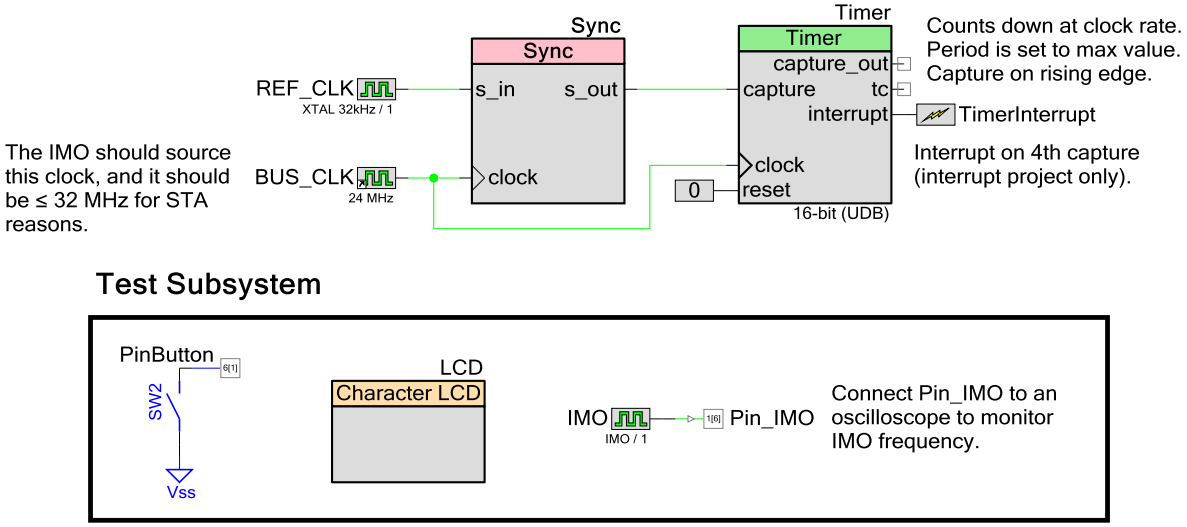
Associated Parts: All PSoC 3 and PSoC 5LP parts

Related Hardware: CY8CKIT-030, CY8CKIT-050

Design

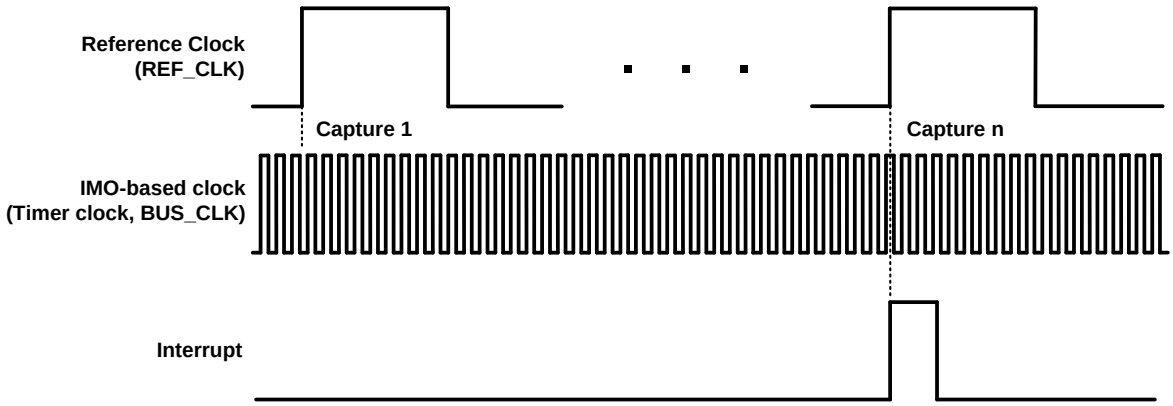
The PSoC hardware design uses a 16-bit Timer Component in PSoC Creator, as Figure 1 shows. The Sync Component synchronizes the two clocks to avoid static timing analysis (STA) warnings. For more information on STA, see AN81623, PSoC Digital Design Best Practices.

Figure 1. IMO Trim Design Schematic



The Timer Component is based on PSoC universal digital blocks (UDBs), as opposed to the fixed-function timer/counter/PWM (TCPWM) blocks that PSoC also has. The reason for this is that the UDB-based Timer can save multiple counter captures before generating an interrupt, as Figure 2 shows; a TCPWM block can save only one counter capture. Multiple captures are implemented for increased accuracy.

Figure 2. IMO Trim Design Timing Diagram



The firmware reads the counter capture values and computes the difference between the first and last capture values. (Intermediate capture values are discarded.) Because the Timer Component counts down, the first capture value is always greater than the last capture value (unless a counter wraparound occurs). As a result, the measured number of timer clocks is:

$$N_{TIMER_CLOCKS} = (Capture\ 1 – Capture\ n) / (n – 1)$$

Note that the above formula works even if the counter wraps around zero. However, counting always starts from the maximum value of 65535, so the chance of wraparound before the last capture is low.

The number of captures, n, can be chosen based on several factors, including accuracy. However, the dominant factor is that the UDBs have a 4-deep FIFO in which the counter capture values are saved, so it is convenient for n to be 4. In polled mode, the firmware monitors the Timer status for FIFO full.

The target number of timer clock edges is $F_{TIMER_CLOCK} / F_{REF_CLOCK}$. The calculated N_{TIMER_CLOCKS} is compared to the target, and an IMO trim register (IMO_TR1) is incremented or decremented accordingly to change the IMO frequency. For more information on the IMO trim register, see the PSoC 3 or PSoC 5LP *Registers Technical Reference Manual* (TRM).

A test subsystem displays the current values of IMO_TR1 and N_{TIMER_CLOCKS}. A new measurement and adjustment is made whenever a kit button is pressed and released. This results in a convergence (as close as possible) of the IMO frequency to the target value. A spare pin enables observation of the current IMO frequency using an oscilloscope or a frequency counter.

Design Considerations

This code example is designed to be used with the CY8CKIT-030 or CY8CKIT-050 kit, mainly because these kits have a 32.768-kHz crystal connected to the PSoC kHzECO pins. The design can be adapted to work in any system that provides a crystal or other accurate clock as a reference.

The firmware in this example mainly exists in two functions: main() and DoOneIMO_Trim(). The interrupt project also has an interrupt handler function TimerISR(). The functions are easy to copy and paste into other designs.

Software Setup

Both *main.c* files have statements that define the frequencies of the timer clock and the reference clock, as Code 1 shows. If needed, adjust these values for your design.

Code 1. Clock Frequency Definitions

```
/* Timer clock frequency in Hz */
#define  FTMRCLK  24000000
/* Reference clock frequency in Hz */
#define  FREF     32768
```

Components

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

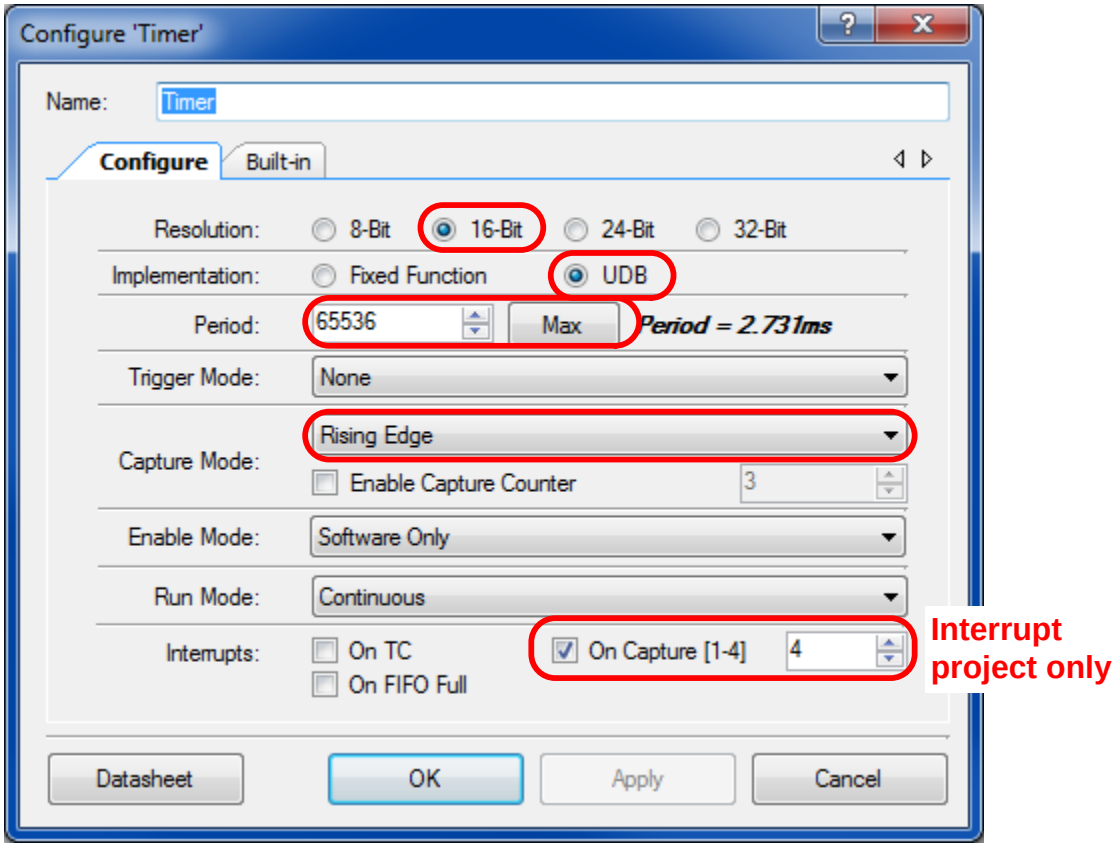
Table 1. List of PSoC Creator Components

Component	Version	Hardware Resources
Timer	2.70	2 UDBs
Sync	1.0	
Clock	2.20	2 digital clock dividers
Interrupt	1.70	1 interrupt
Test Subsystem		
LCD	2.20	7 pins
PinButton, Pin_IMO	2.20	2 pins

Parameter Settings

Figure 3 shows the changed settings for the Timer Component.

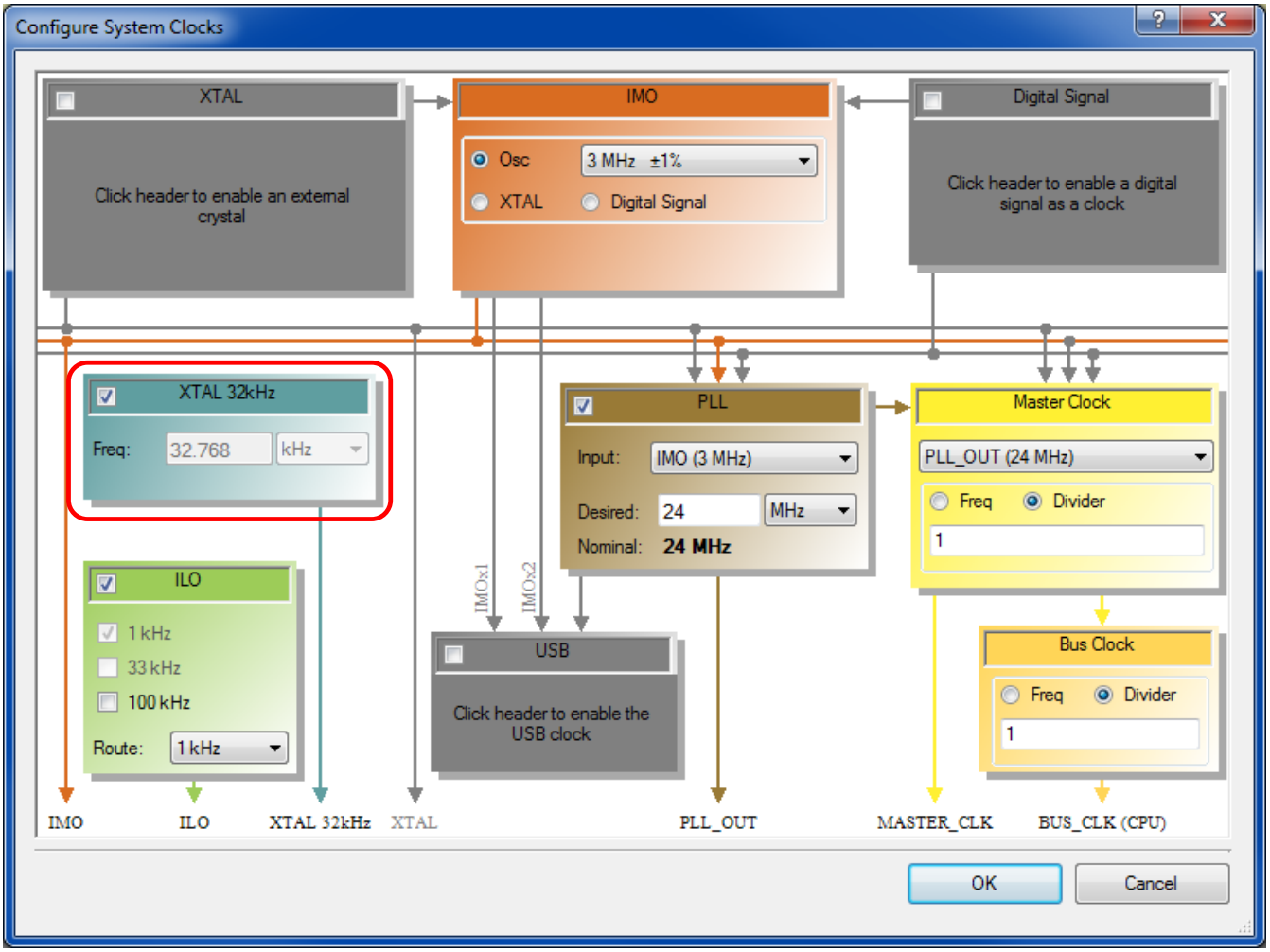
Figure 3. Timer Component Configuration



Design-Wide / Global Resources

XTAL 32kHz must be turned on, as Figure 4 shows. Also, confirm that the IMO setting is at the default shown.

Figure 4. System Clocks Configuration



Operation

For each project, do the following:

1. Using PSoC Creator, build the project. The project is configured for the PSoC device in CY8CKIT-050. If you are using CY8CKIT-030, change the device to that used in CY8CKIT-030: CY8C3866AXI-040. For more information on selecting a device and building a project, see PSoC Creator Help.
2. Connect the kit to your computer using the USB cable.
3. Program the project into the kit using either PSoC Creator or PSoC Programmer. For more information on device programming, see PSoC Creator Help. After programming, confirm that the LCD display is similar to Figure 5:

Figure 5. Start Trim Display



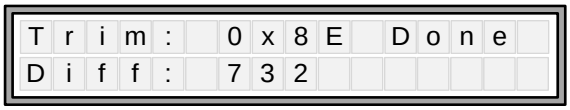
4. Optionally, connect an oscilloscope or frequency counter to kit connector P2 pin 2, and confirm that the IMO frequency is approximately 3 MHz.
5. Press and release the kit button SW2. Confirm that the LCD display is similar to Figure 6:

Figure 6. Trim Display



6. Continue pressing and releasing the kit button SW2. Confirm that the trim value converges, and the difference value converges to at or near the target. Confirm that eventually the LCD display is similar to Figure 7. If observing the IMO frequency (Step 4), confirm that the frequency changes in a corresponding fashion.

Figure 7. Trim Done Display



Related Documents

Table 2 lists all relevant application notes, code examples, PSoC Creator Component datasheets, and device and kit information.

Table 2. Related Documents

Application Notes		
AN60631	PSoC 3 and PSoC 5LP Clocking Resources	Describes PSoC clocking resources, including oscillators, clock sources, phase-locked loop (PLL), and clock distribution network.
AN54439	PSoC 3 and PSoC 5LP External Crystal Oscillators	Describes how to use an external crystal or ceramic resonator at 32.768 kHz or in the 4-25 MHz range with PSoC.
AN81623	PSoC Digital Design Best Practices	Describes best practices for digital design using PSoC Creator. In-depth discussion of static timing analysis.
Code Examples		
CE97601	Improving the Accuracy of the PSoC 4 Internal Main Oscillator	Similar to this code example, but for the PSoC 4 family.
PSoC Creator Component Datasheets		
Timer	Supports fixed block and UDB-based timers	
Sync	Synchronizes a signal to a clock	
Clock	Supports definition and configuration of local, system, and design-wide clocks	
Interrupt	Supports definition and configuration of hardware interrupts to the CPU	
Pins	Supports connection of hardware resources to physical pins	
Device Documentation		
PSoC 3 Datasheets	PSoC 3 Technical Reference Manuals	
PSoC 5LP Datasheets	PSoC 5LP Technical Reference Manuals	
Development Kit (DVK) Documentation		
PSoC 3 CY8CKIT-030, and PSoC 5LP CY8CKIT-050 Click the Kits tab.		

Document History

Document Title: CE219322 - PSoC® 3 and PSoC 5LP Internal Main Oscillator Trim

Document Number: 002-19322

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5726196	MKEA	06/02/2017	New code example

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, 2017. 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.