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.

CE95915 - Implementing an RTC with PSoC® 4100/PSoC 4200 Devices

Objective

This code example demonstrates the implementation of a real-time clock (RTC) with PSoC® 4100 and PSoC 4200 devices.

Overview

This example project demonstrates the implementation of a real-time clock (RTC) in PSoC 4100 and PSoC 4200 devices using a 32.768-kHz crystal and PSoC internal resources.

An RTC provides time and date information – second, minute, hour, day of the week, day of the month, day of the year month, and year. The time and date information are updated every second based on a one-second interrupt generated from a 32.768-kHz crystal. The clock accuracy depends on the crystal; it is typically 20 ppm.

PSoC 4100 and PSoC 4200 devices do not have built-in watch crystal support. To overcome this limitation, an external crystal oscillator (ECO) circuit is created using PSoC 4's internal resources and external passive components. For details of ECO implementation with these devices and selection of the passive components, see the knowledge base article, Implementing a 32-kHz ECO Interface with PSoC® 4100/ PSoC 4200 – KBA95848.

PSoC Resources

Cypress provides a wealth of data at www.cypress.com to help you to select the right PSoC device for your design, and quickly and effectively integrate the device into your design. For a comprehensive list of resources, see KBA86521, How to Design with PSoC 3, PSoC 4, and PSoC 5LP. The following is an abbreviated list for PSoC 4:

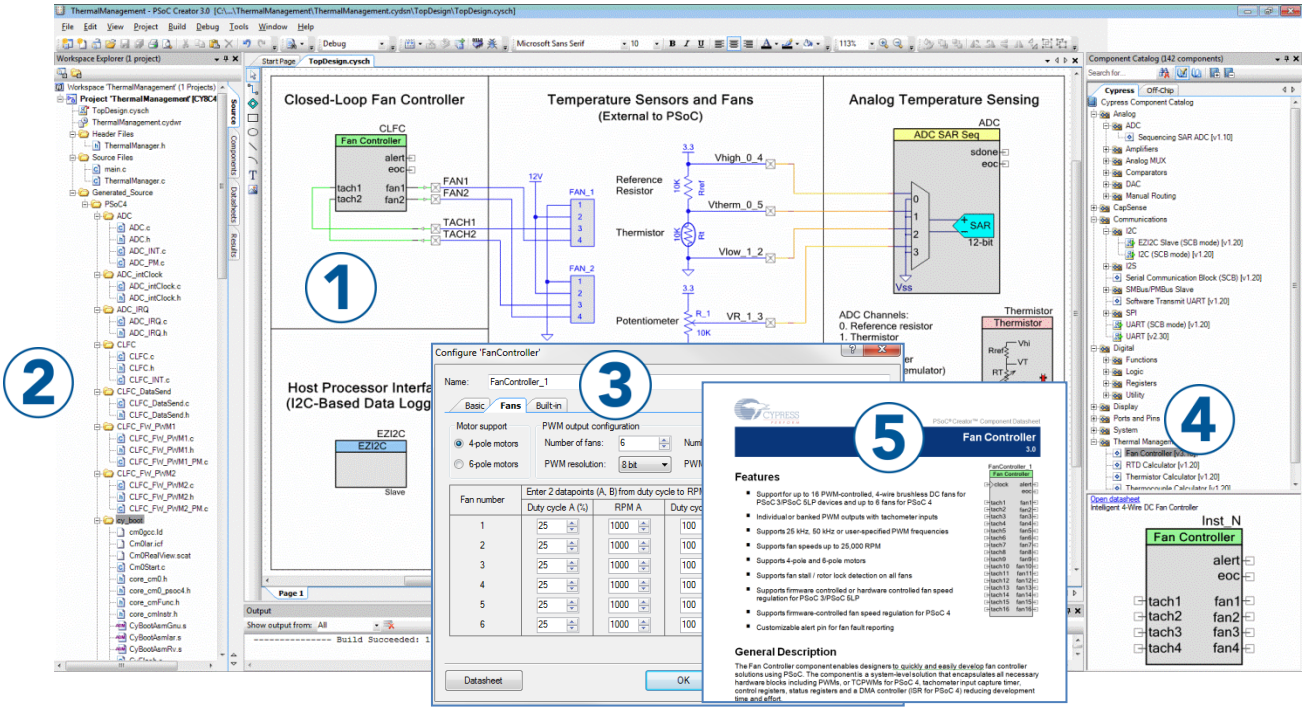
- **Overview:** PSoC Portfolio, PSoC Roadmap
- **Product Selectors:** PSoC 1, PSoC 3, PSoC 4, or PSoC 5LP. In addition, PSoC Creator includes a device selection tool.
- **Datasheets:** Describe and provide electrical specifications for the PSoC 3, PSoC 4, and PSoC 5LP device families.
- **CapSense Design Guides:** Learn how to design capacitive touch-sensing applications with the PSoC 3, PSoC 4, and PSoC 5LP families of devices.
- **Application Notes and Code Examples:** Cover a broad range of topics, from basic to advanced level. Many of the application notes include code examples.
- **Technical Reference Manuals (TRM):** Provide detailed descriptions of the architecture and registers in each of the PSoC 3, PSoC 4, and PSoC 5LP device families.
- **Development Kits:**
  - CY8CKIT-042 and CY8CKIT-040, PSoC 4 Pioneer kits, are easy-to-use and inexpensive development platforms. These kits include connectors for Arduino™ compatible shields and Digilent® Pmod™ daughter cards.
  - CY8CKIT-049 is a series of very low-cost prototyping platform for sampling PSoC 4 devices.

PSoC Creator

PSoC Creator is a free Windows-based Integrated Design Environment (IDE). It enables concurrent hardware and firmware design of systems based on PSoC 3, PSoC 4, and PSoC 5LP. See Figure 1 – with PSoC Creator, you can:

1. Drag and drop Components to build your hardware system design in the main design workspace
2. Codesign your application firmware with the PSoC hardware
3. Configure Components using configuration tools
4. Explore the library of 100+ Components
5. Review Component datasheets

Figure 1. PSoC Creator Features



Requirements

Tool: PSoC Creator 4.2

Programming Language: C (Arm® GCC 5.4-2016-q2-update and Arm MDK compilers)

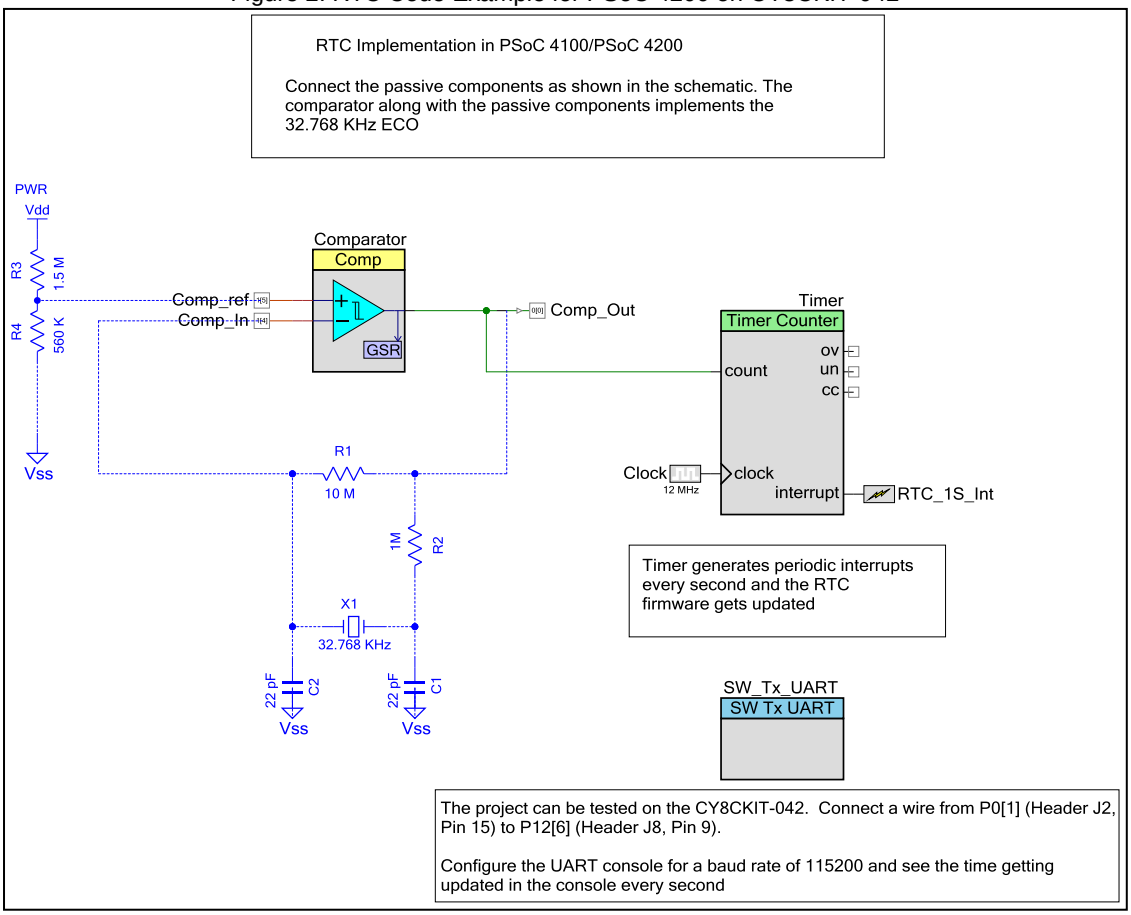
Associated Parts: All PSoC 4100 and PSoC 4200 parts

Related Hardware: CY8CKIT-042

Design

Figure 2 shows the code example design.

Figure 2. RTC Code Example for PSoC 4200 on CY8CKIT-042



The code example features the following:

- Implementation of a 32-768 kHz ECO using a comparator and passive components.
- Generation of a one-second interrupt
  - The Timer Counter Component is clocked using the generated 32.768-kHz signal. The timer period value is set to 32767, causing an interrupt to be generated every second. The interrupt service routine (ISR) code updates the RTC parameters.
- UART-based communication with a PC. Because a bit-bang software transmit UART is used, it takes some CPU processing bandwidth.

Code Design

In the initialization section, the RTC's structure members are initialized with time and date values. The one-second interrupt ISR updates the structure and sends the time and date via the UART. The main loop does nothing.

The flowchart in Figure 3 shows the overall code flow.

The RTC firmware updates the time, date, and other RTC parameters every second (inside the one-second ISR triggered by the timer) and displays the time. Figure 4 shows the general RTC firmware update algorithm.

Various event handlers (like RTC_EveryMinuteHandler, RTC_EveryHourHandler, etc.) provided in the project allow the user to read a particular parameter (like minute, hour, etc.) and process this data.

Figure 3. Functional Flow of Firmware

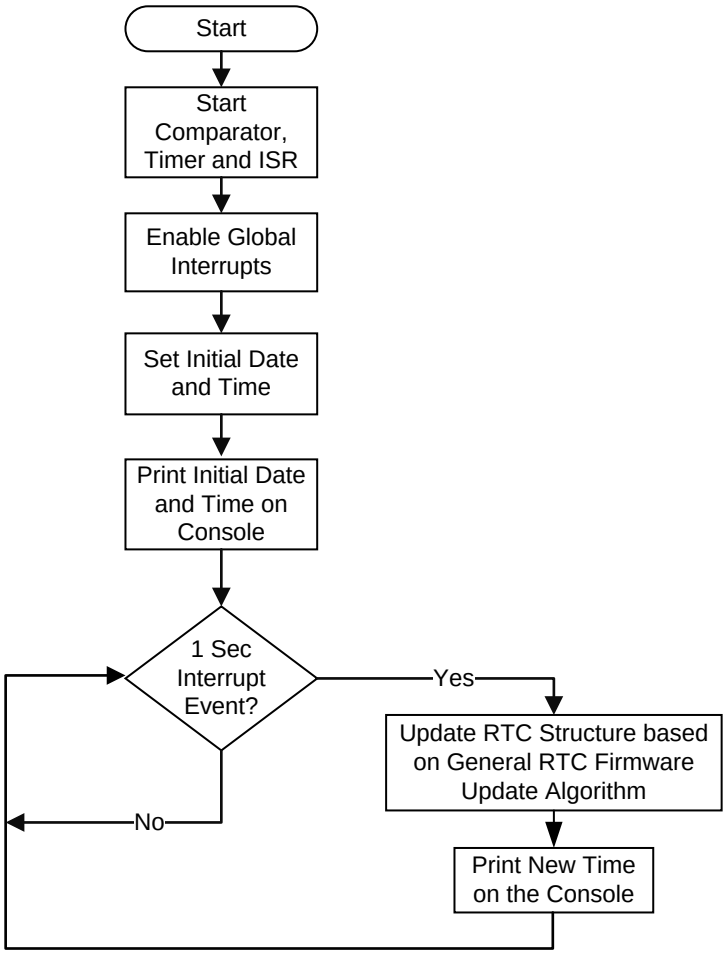
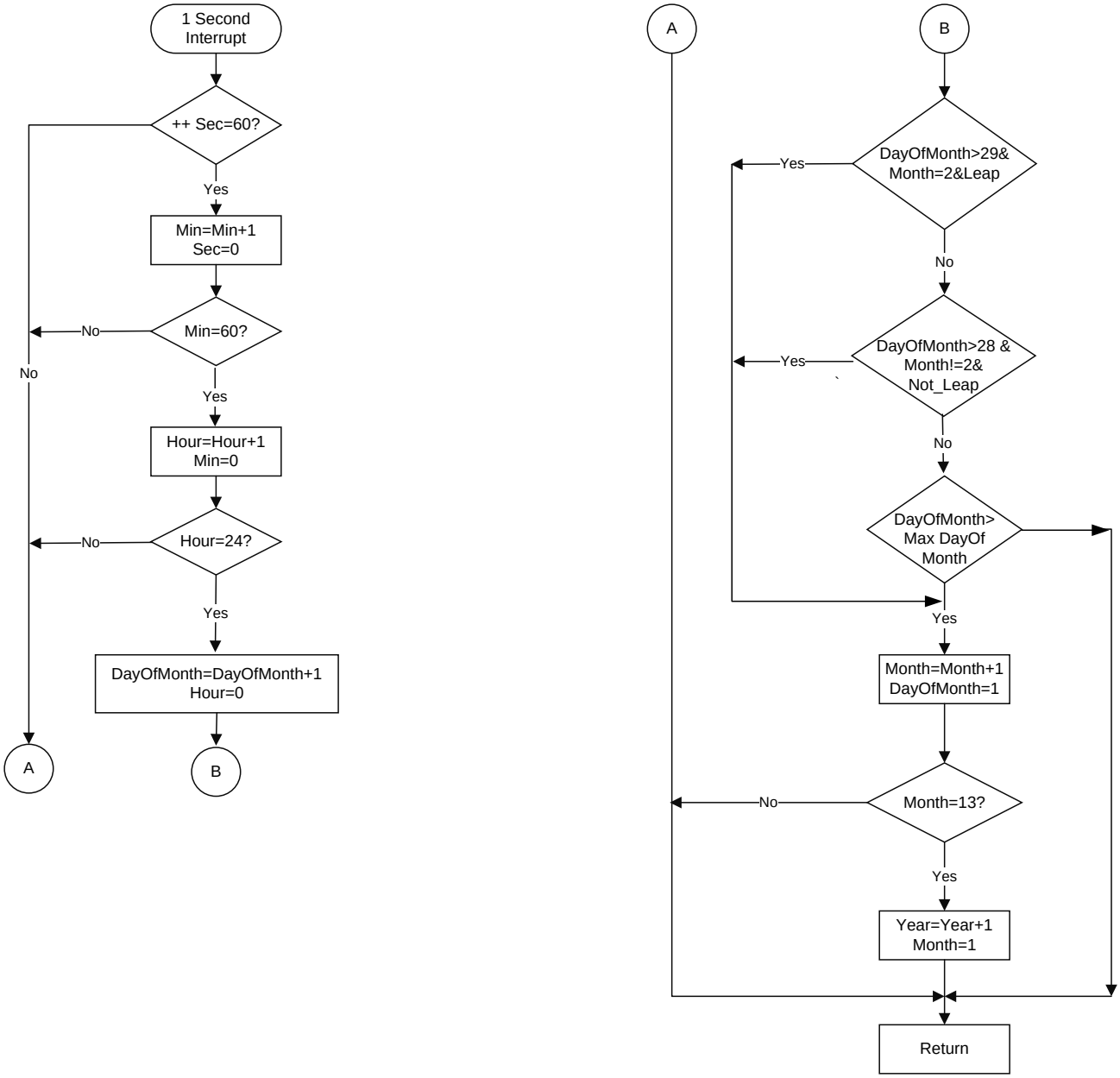


Figure 4. General RTC Firmware Update Algorithm



Design Considerations

- Off-chip connections are needed to implement the ECO Interface.
- This code example has been specifically designed for the CY8CKIT-042 PSoC 4 Pioneer Kit. In this kit, the on-board PSoC 5LP acts as a USB-UART bridge. The time can be displayed on a UART console. Any serial communication software, such as CoolTerm, Tera Term, or PuTTY, can be used to display the UART output. The project can also be modified to display the time and date on a Character LCD or I^2C LCD.
- The code example can be modified to run on CY8CKIT-049 4xxx Prototyping Kits. It can also be modified to run on CY8CKIT-001 by using the Character LCD on the kit to display the time and date.

Errata

None.

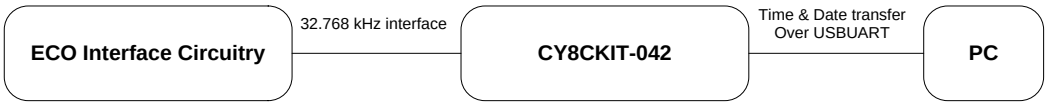
Hardware Setup

You need the following hardware:

- CY8CKIT-042 PSoC 4200 Pioneer Kit
- Resistors – 560 K (1), 1 M (1), 1.5 M (1), 10 M (1)
- Capacitors – 22 pF (2)
- Crystal – 32.768 kHz (1)
- Connecting Wire (1)
- USB-A to Mini-B cable

Figure 5 represents the hardware setup required to test the project. Connect the passive components (resistors, capacitors, and crystal) to the CY8CKIT-042 kit as Figure 2 shows. The CY8CKIT-042 kit connects to the PC through the USB-A to Mini-B cable.

Figure 5. Hardware Setup Block Diagram



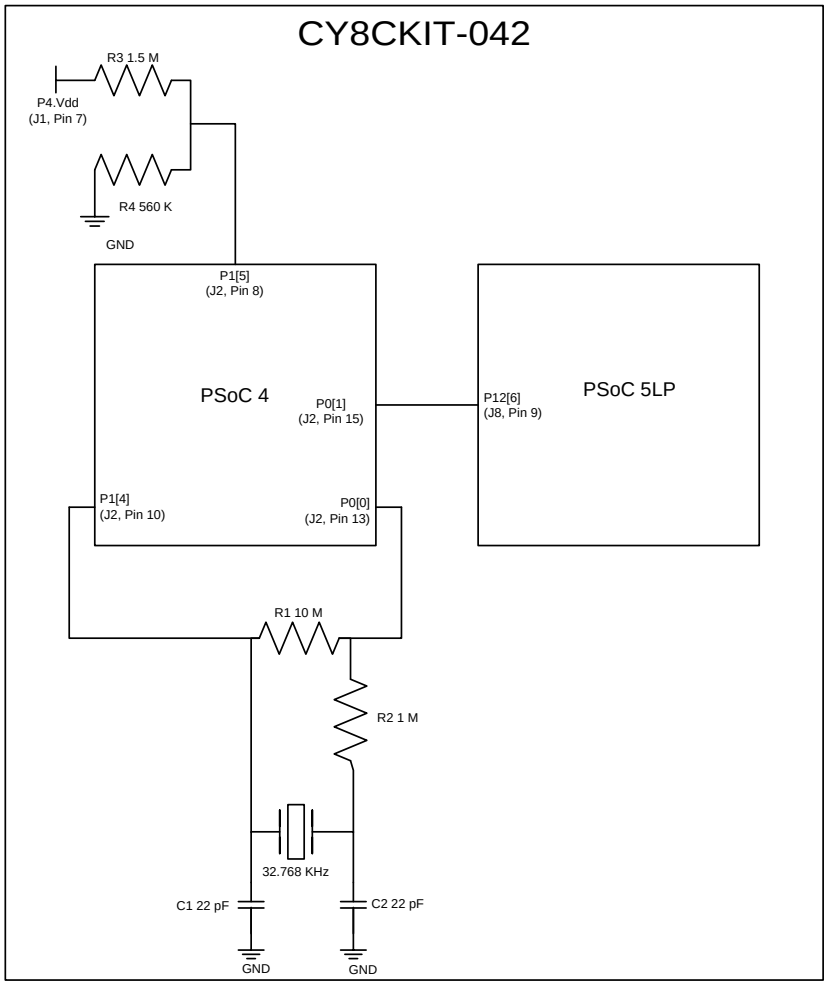
Make the following hardware connections on CY8CKIT-042:

1. Set Jumper J9 to either 3.3 V or 5 V.
2. Connect the 1.5-MΩ resistor between P4.Vdd (J1 pin 7) and P1[5] (J2 pin 8).
3. Connect the 560-kΩ resistor between GND (J1 pin 2) and P1[5] (J2 pin 8).
4. Connect the 10-MΩ resistor between P0[0] (J2 pin 13) and P1[4] (J2 pin 10).
5. Connect the 32.768-kHz crystal and 1-M resistor series combination in parallel to the 10 M resistor.
6. Connect two 22-pF capacitors between each terminal of the 32.768-kHz crystal and GND (J1 pin 2).

   Note: You can create an interface board containing the components listed. The board can then be connected to the CY8CKIT-042 kit.
7. Connect a wire from P0[1] (J2 pin 15) to P12[6] of PSoC5LP device (J8 pin 9).
8. Use the USB-A to Mini-B cable to connect the CY8CKIT-042 kit to the PC.

The connection diagram for the project is shown in Figure 6. The connections are shown with reference to CY8CKIT-042.

Figure 6. CY8CKIT-042 Connection Diagram



Software Setup

No special software setup is required for this project. All supported compilers can be used with any optimization.

At the PSoC Creator project's default CPU clock speed (24 MHz for PSoC 4200), the CPU has enough cycles to support the example. The RTC update ISR uses about 0.03% of the CPU's bandwidth, and the software transmit UART operating at 115,200 baud uses approximately 0.2% of the CPU bandwidth.

Components

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

Table 1. List of PSoC Creator Components Used by the Project

Component	Placement
Comparator	PSoC 4200 Opamp
Timer (TCPWM mode)	PSoC 4200 TCPWM
SW_Tx_UART	None – Software-controlled bit-bang UART does not use any hardware resources
Clock	None - no clock dividers are consumed
isr	1 interrupt
Pins	2 Analog Pins for the Comparator input , 1 Digital Output Pin for comparator output, 1 Digital Output Pin for the SW_Tx_UART

Parameter Settings

Table 2 lists the parameter settings for each of the PSoC Creator Components used in the example project. Only the parameters that vary from the default values are listed.

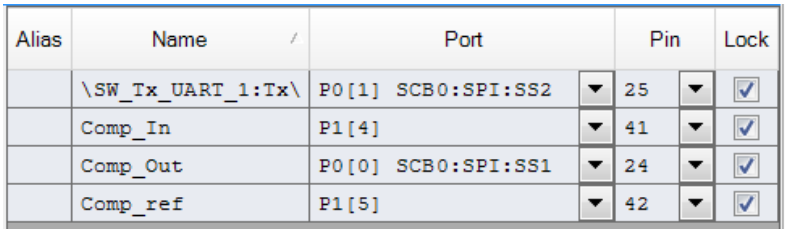
Table 2. List of PSoC Creator Component Parameter Settings for PSoC 4200 Example

Component	Non-Default Parameter Settings
Timer (TCPWM mode)	Configuration = Timer/Counter, count Present = Checked (Mode = Rising edge), Period = 32767
Clock	Frequency = 12 MHz

Design-Wide Resources

Figure 7 shows the pin assignments for the example project. No other design-wide resource needs to be changed from its default setting.

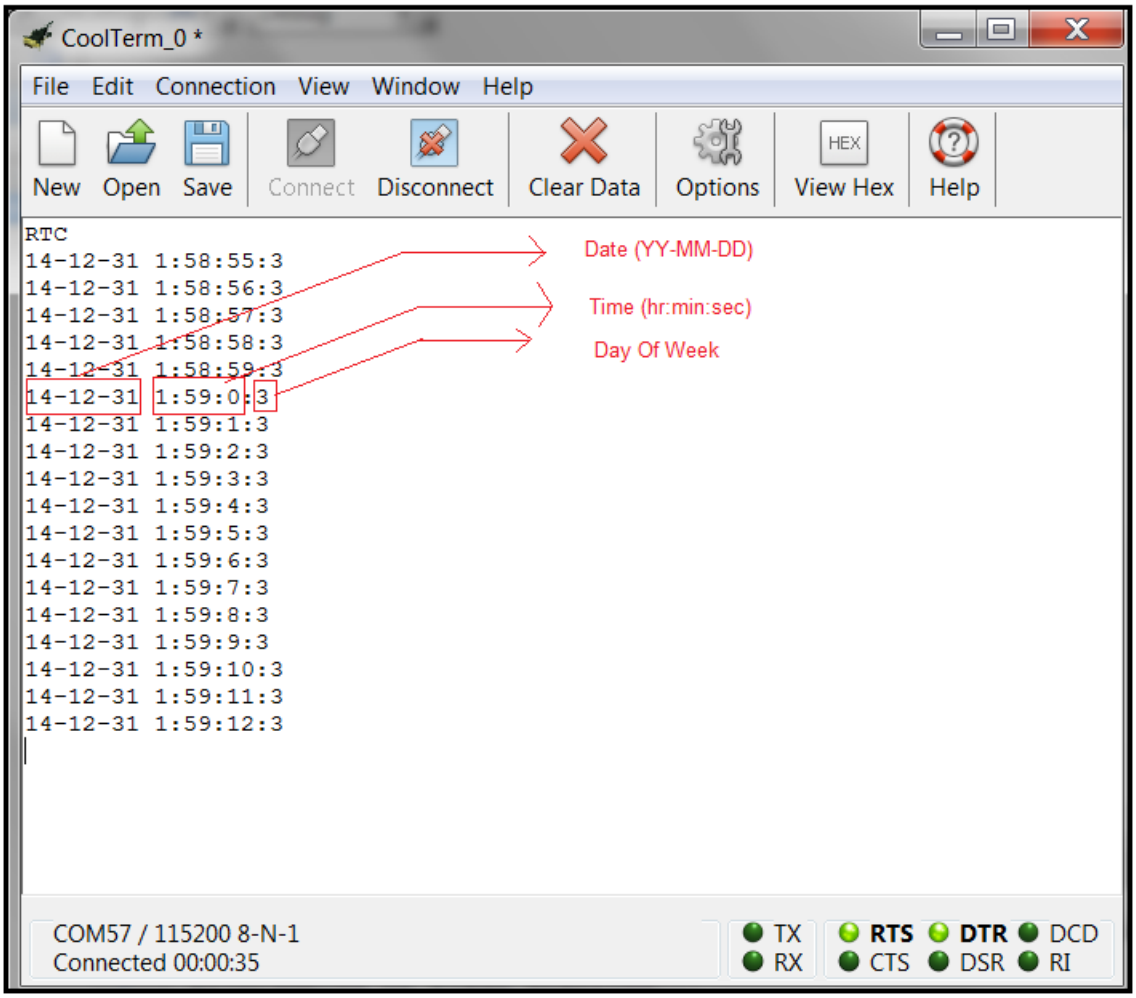
Figure 7. Pin Assignments for PSoC 4200 Example



Operation

1. Ensure that the necessary hardware connections are made.
2. Build the project in PSoC Creator.
3. Program the project into the CY8C4245AXI-483 device, which is the default device in the CY8CKIT-042 PSoC 4 Pioneer Kit.
4. Configure any UART console (like Hyperterminal, Coolterm, etc.) for a baud rate of 115,200.
5. Select the COM port that is enumerated as “KitProg USB-UART” in the Device Manager of the PC (Make sure that a wire is connected between P0[1] (J2 pin 15) and P12[6] (J8 pin 9) of the CY8CKIT-042 kit.
6. Connect the console and reset the device by pressing SW1 (Reset Switch). Verify that the time is updated once per second, as Figure 8 shows.

Figure 8. Time shown in the console



Upgrade Information

N/A

Related Documents

Application Notes		
AN79953	Getting Started with PSoC 4	Introduces user to PSoC® 4, an ARM® Cortex™-M0 based programmable system-on-chip.
AN90799	PSoC 4 Interrupts	Explains the interrupt architecture in PSoC 4 and its configuration in PSoC Creator IDE
Code Examples		
RTC Design	Example Project for RTC Implementation in PSoC 3 and PSoC 5LP using the available RTC Component	
Knowledge Base Articles		
Implementing a 32-kHz ECO Interface with PSoC® 4100/ PSoC 4200 – KBA95848	How can we implement a 32.768 kHz ECO interface with PSoC 4100 and PSoC 4200 series devices?	
PSoC Creator Component Datasheets		
PSoC 4 Voltage Comparator	Provides a hardware solution to compare two analog input voltages	
PSoC 4 Timer Counter Pulse Width Modulator (TCPWM)	Supports Timer/Counter, PWM, and Quadrature Decoder using the PSoC 4 TCPWM block	
Interrupt	Defines hardware-triggered interrupts	
Software Transmit UART	Provides an 8-bit RS-232 data-format compliant serial transmitter through software bit-banging	
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 4 Datasheets	PSoC 4 Technical Reference Manuals	
Development Kit (DVK) Documentation		
PSoC 4 Kits		

Document History

Document Title: CE95915 – Implementing an RTC with PSoC® 4100/PSoC 4200 Devices

Document Number: 001-95915

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	4676194	BMAH	03/03/2015	New code example
*A	5739811	AESATP12	05/26/2017	Updated logo and copyright.
*B	6078336	BMAH	03/02/2018	Updated Project to PSoC Creator 4.2

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 MCU

Cypress Developer Community

Community | 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. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. 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.