CE223541 – PSoC 6 MCU: Hello World using ModusToolbox

Objective

This example demonstrates UART communication and blinks an LED using a TCPWM Component, on the PSoC® 6 MCU, using ModusToolbox IDE.

Requirements

Tool: ModusToolbox™ 1.0; Cypress SDK

Programming Language: C

Associated Parts: All PSoC 6 MCU parts

Related Hardware: PSoC 6 WiFi-BT Pioneer Kit

Overview

This example uses the TCPWM in PWM mode to blink a green LED once per second. The UART resource prints a “Hello World! This is PSoC 6 MCU.” message.

Hardware Setup

This example uses the kit’s default configuration. Refer to the kit guide to ensure the kit is configured correctly.

Software Setup

This example uses a terminal emulator. Install one if you don’t have one. The instructions use Tera Term.

Operation

1. Open your terminal software and select the KitProg3 COM port. Set the serial port parameters to 8N1 and 115,200 baud.
2. Connect the Pioneer board to your PC using the provided USB cable through the USB connector.
3. Import the project into a new workspace.
4. Build the project. For a multi-core device, choose **Project** > **Build All**.
5. Program the PSoC 6 MCU device. Choose **Run** > **Run Configurations**. Expand **GDB OpenOCD Debugging**. Select the **<project> Program OpenOCD** item. Then click **Run**.
6. Observe the “Hello World! This is PSoC 6 MCU.” message on the UART terminal, and the green LED toggles every 500 milliseconds.

Design and Implementation

This example configures a TCPWM resource in PWM mode to blink the green LED at 1 Hz, and a UART resource to print “Hello World! This is PSoC 6 MCU.” message.

The TCPWM resource is connected to a clock operating at 1 kHz, with a compare value of 500. It outputs a signal to pin P1[1], which drives the green LED on the kit. The Serial Communication Block resource is configured as a UART at 115200 baud, 8N1. It is connected to a clock operating at approximately 926 kHz to generate the correct baud rate. The RX is on pin P5[0] and TX is on P5[1], to match the pin usage on the kit.

All code runs on the CM4 processor.

To see all the settings, review the *design.modus* file in the project.

Resources and Settings

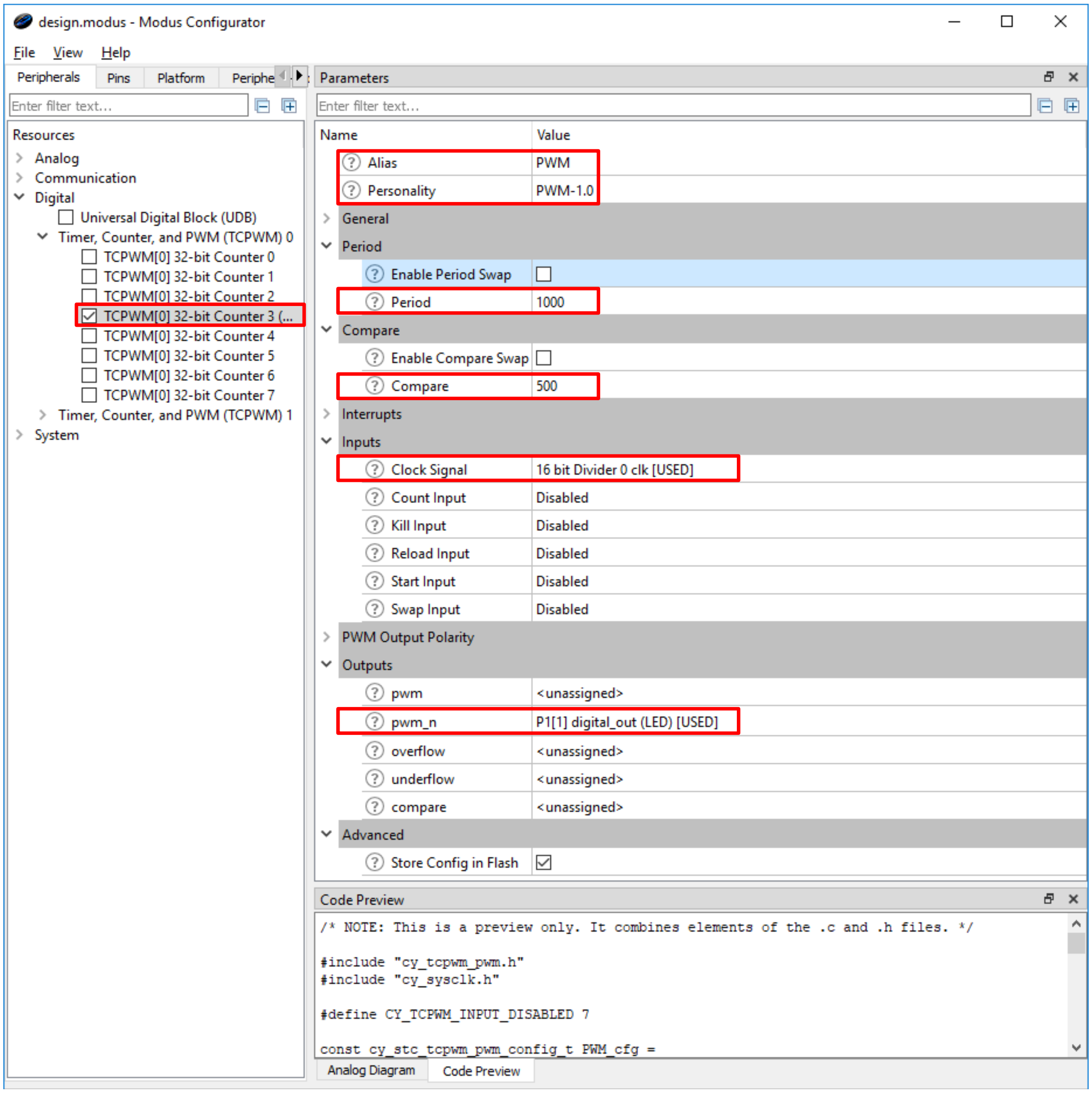
Table 1 lists the ModusToolbox resources used in this example, and how they are used in the design.

Table 1: ModusToolbox Resources

Resource	Alias	Purpose
Timer Counter (TCPWM)	PWM	Drives the green LED
SCB	UART	Prints a message to a terminal window
Digital Output Pin	LED	Provide visual feedback
	UART_TX	Used for UART transmit (Tx)
Digital Input Pin	UART_RX	Used for UART receive (Rx)

Figure 1 to Figure 4 show non-default configuration settings for the Resources.

Figure 1. PWM Configuration



Resources
- Analog
- Communication
- Digital
  - □ Universal Digital Block (UDB)
  - Timer, Counter, and PWM (TCPWM) 0
    - □ TCPWM[0] 32-bit Counter 0
    - □ TCPWM[0] 32-bit Counter 1
    - □ TCPWM[0] 32-bit Counter 2
    - ☑ TCPWM[0] 32-bit Counter 3 (...
    - □ TCPWM[0] 32-bit Counter 4
    - □ TCPWM[0] 32-bit Counter 5
    - □ TCPWM[0] 32-bit Counter 6
    - □ TCPWM[0] 32-bit Counter 7
  - Timer, Counter, and PWM (TCPWM) 1
- System

Name	Value
Alias	PWM
Personality	PWM-1.0
General	
Period	
Enable Period Swap	□
Period	1000
Compare	
Enable Compare Swap	□
Compare	500
Interrupts	
Inputs	
Clock Signal	16 bit Divider 0 clk [USED]
Count Input	Disabled
Kill Input	Disabled
Reload Input	Disabled
Start Input	Disabled
Swap Input	Disabled
PWM Output Polarity	
Outputs	
pwm	<unassigned>
pwm_n	P1[1] digital_out (LED) [USED]
overflow	<unassigned>
underflow	<unassigned>
compare	<unassigned>
Advanced	
Store Config in Flash	☑

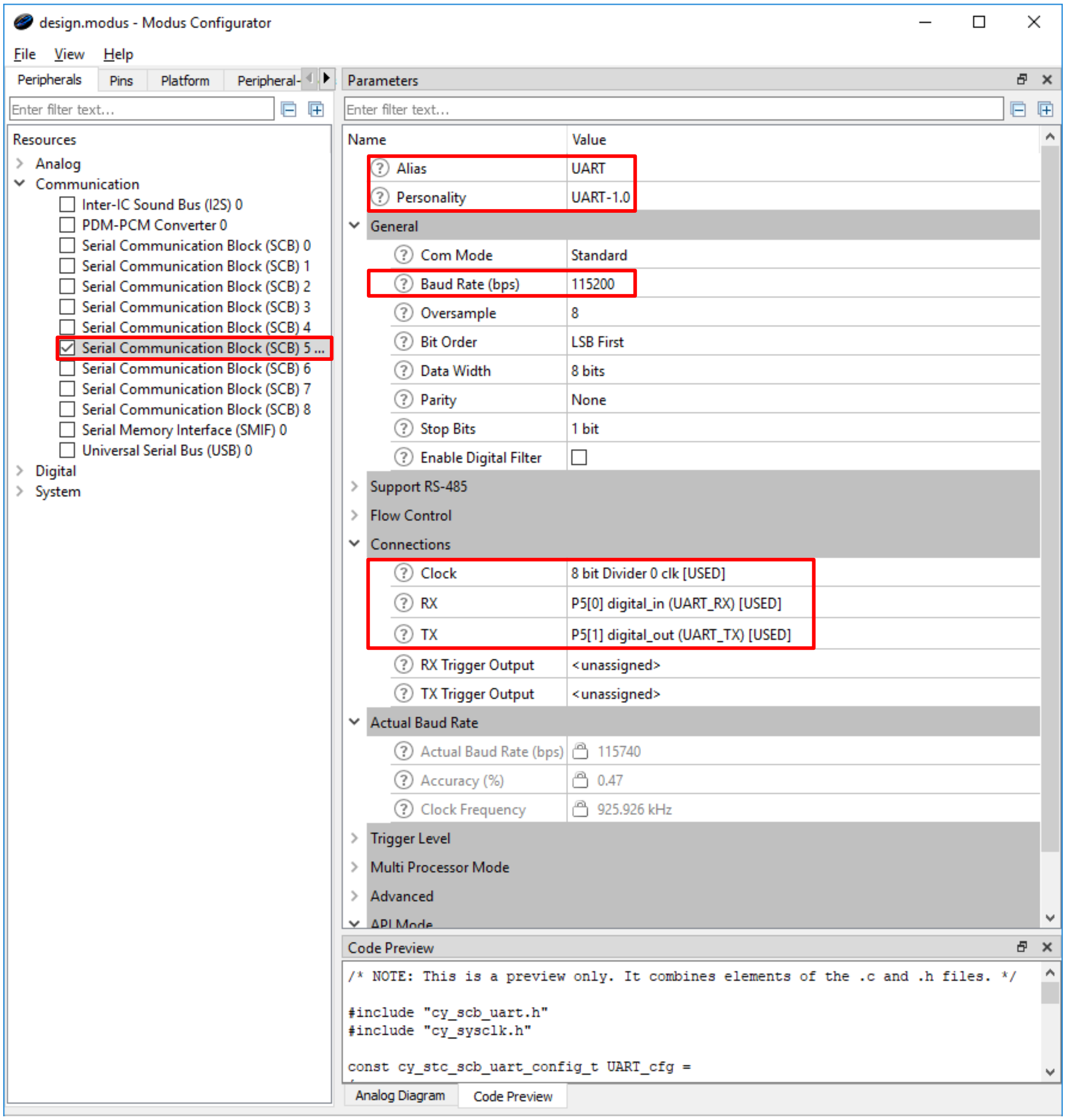
```
/* NOTE: This is a preview only. It combines elements of the .c and .h files. */

#include "cy_tcpwm_pwm.h"
#include "cy_sysclk.h"

#define CY_TCPWM_INPUT_DISABLED 7

const cy_stc_tcpwm_pwm_config_t PWM_cfg =
```

Figure 2. UART Configuration



Resources
- Analog
- Communication
  - ☐ Inter-IC Sound Bus (I2S) 0
  - ☐ PDM-PCM Converter 0
  - ☐ Serial Communication Block (SCB) 0
  - ☐ Serial Communication Block (SCB) 1
  - ☐ Serial Communication Block (SCB) 2
  - ☐ Serial Communication Block (SCB) 3
  - ☐ Serial Communication Block (SCB) 4
  - ☑ Serial Communication Block (SCB) 5 ...
  - ☐ Serial Communication Block (SCB) 6
  - ☐ Serial Communication Block (SCB) 7
  - ☐ Serial Communication Block (SCB) 8
  - ☐ Serial Memory Interface (SMIF) 0
  - ☐ Universal Serial Bus (USB) 0
- Digital
- System

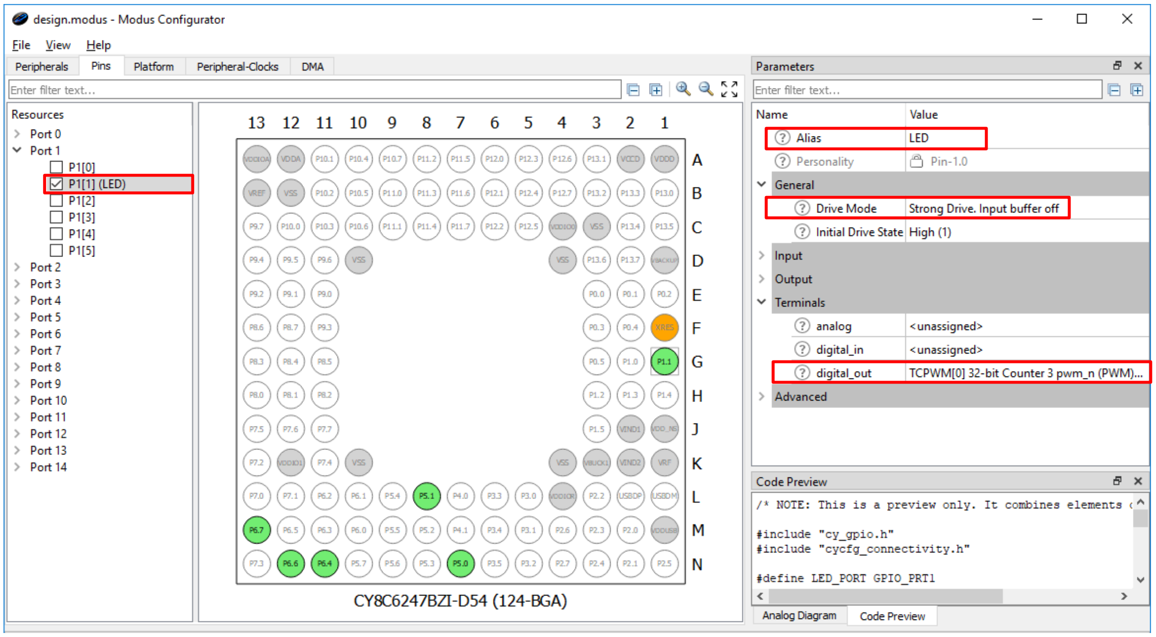
Name	Value
Alias	UART
Personality	UART-1.0
General	
Com Mode	Standard
Baud Rate (bps)	115200
Oversample	8
Bit Order	LSB First
Data Width	8 bits
Parity	None
Stop Bits	1 bit
Enable Digital Filter	☐
Support RS-485	
Flow Control	
Connections	
Clock	8 bit Divider 0 clk [USED]
RX	P5[0] digital_in (UART_RX) [USED]
TX	P5[1] digital_out (UART_TX) [USED]
RX Trigger Output	<unassigned>
TX Trigger Output	<unassigned>
Actual Baud Rate	
Actual Baud Rate (bps)	115740
Accuracy (%)	0.47
Clock Frequency	925.926 kHz
Trigger Level	
Multi Processor Mode	
Advanced	
API Mode	

```
/* NOTE: This is a preview only. It combines elements of the .c and .h files. */

#include "cy_scb_uart.h"
#include "cy_sysclk.h"

const cy_stc_scb_uart_config_t UART_cfg =
```

Figure 3. GPIO Pin configuration for LED


design.modus - Modus Configurator
File View Help
Peripherals Pins Platform Peripheral-Clocks DMA
Enter filter text...
13 12 11 10 9 8 7 6 5 4 3 2 1
A B C D E F G H J K L M N
CY8C6247BZI-D54 (124-BGA)
Analog Diagram Code Preview

Resources
- Port 0
- Port 1
  - □ P1[0]
  - ☑ P1[1] (LED)
  - □ P1[2]
  - □ P1[3]
  - □ P1[4]
  - □ P1[5]
- Port 2
- Port 3
- Port 4
- Port 5
- Port 6
- Port 7
- Port 8
- Port 9
- Port 10
- Port 11
- Port 12
- Port 13
- Port 14

Parameters

Name	Value
Alias	LED
Personality	Pin-1.0
General	
Drive Mode	Strong Drive. Input buffer off
Initial Drive State	High (1)
Input	
Output	
Terminals	
analog	<unassigned>
digital_in	<unassigned>
digital_out	TCPWM[0] 32-bit Counter 3 pwm_n (PWM)...
Advanced	

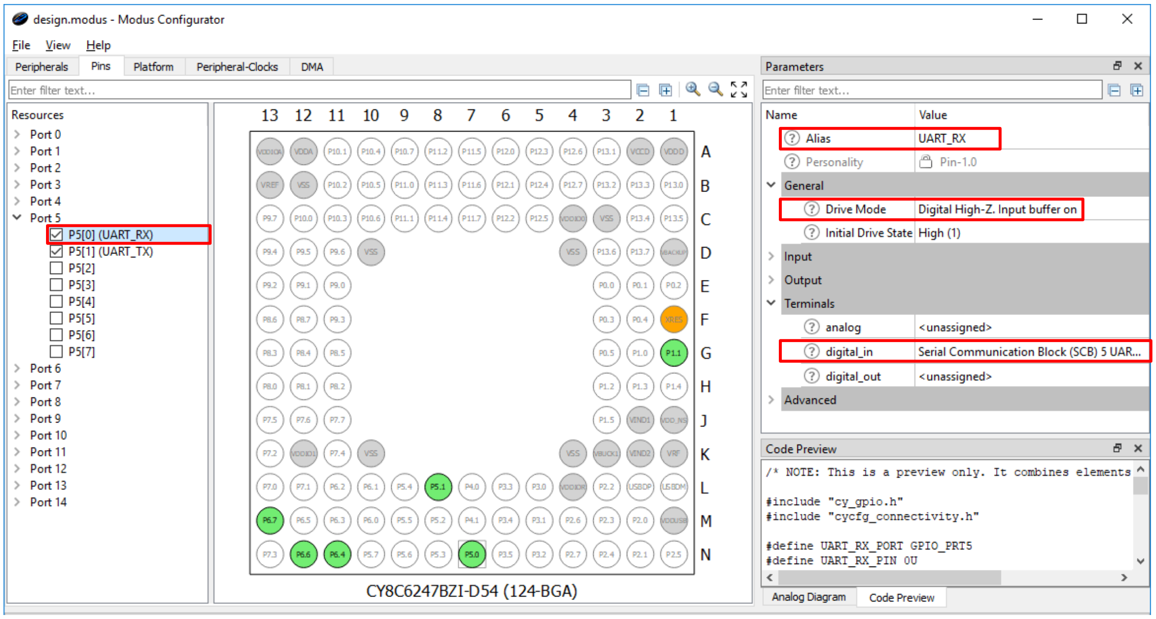
Code Preview

```
/* NOTE: This is a preview only. It combines elements (

#include "cy_gpio.h"
#include "cycfg_connectivity.h"

#define LED_PORT GPIO_PRT1
```

Figure 4. GPIO Pin Configuration for UART Rx


design.modus - Modus Configurator
File View Help
Peripherals Pins Platform Peripheral-Clocks DMA
Enter filter text...
13 12 11 10 9 8 7 6 5 4 3 2 1
A B C D E F G H J K L M N
CY8C6247BZI-D54 (124-BGA)
Analog Diagram Code Preview

Resources
- Port 0
- Port 1
- Port 2
- Port 3
- Port 4
- Port 5
  - ☑ P5[0] (UART_RX)
  - ☑ P5[1] (UART_TX)
  - □ P5[2]
  - □ P5[3]
  - □ P5[4]
  - □ P5[5]
  - □ P5[6]
  - □ P5[7]
- Port 6
- Port 7
- Port 8
- Port 9
- Port 10
- Port 11
- Port 12
- Port 13
- Port 14

Parameters

Name	Value
Alias	UART_RX
Personality	Pin-1.0
General	
Drive Mode	Digital High-Z. Input buffer on
Initial Drive State	High (1)
Input	
Output	
Terminals	
analog	<unassigned>
digital_in	Serial Communication Block (SCB) 5 UAR...
digital_out	<unassigned>
Advanced	

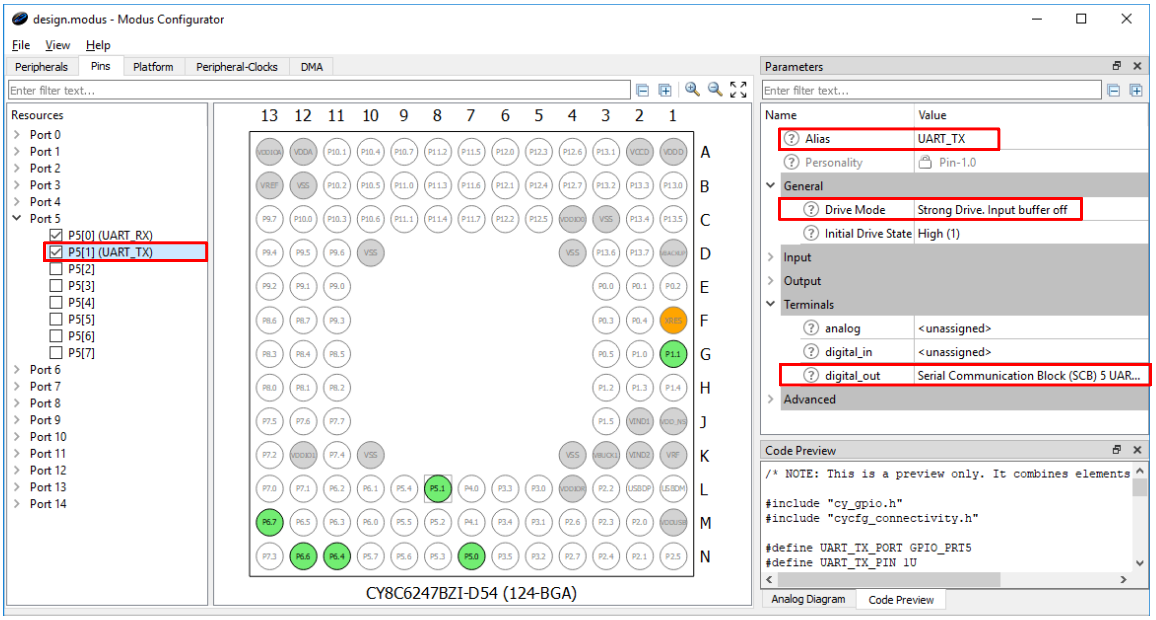
Code Preview

```
/* NOTE: This is a preview only. It combines elements

#include "cy_gpio.h"
#include "cycfg_connectivity.h"

#define UART_RX_PORT GPIO_PRT5
#define UART_RX_PIN 0U
```

Figure 5. GPIO Pin Configuration for UART Tx



☑ P5[0] (UART_RX)
☑ P5[1] (UART_TX)
□ P5[2]
□ P5[3]
□ P5[4]
□ P5[5]
□ P5[6]
□ P5[7]

Parameters

Name	Value
Alias	UART_TX
Personality	Pin-1.0
General	
Drive Mode	Strong Drive. Input buffer off
Initial Drive State	High (1)
Input	
Output	
Terminals	
analog	<unassigned>
digital_in	<unassigned>
digital_out	Serial Communication Block (SCB) 5 UAR...
Advanced	

Code Preview

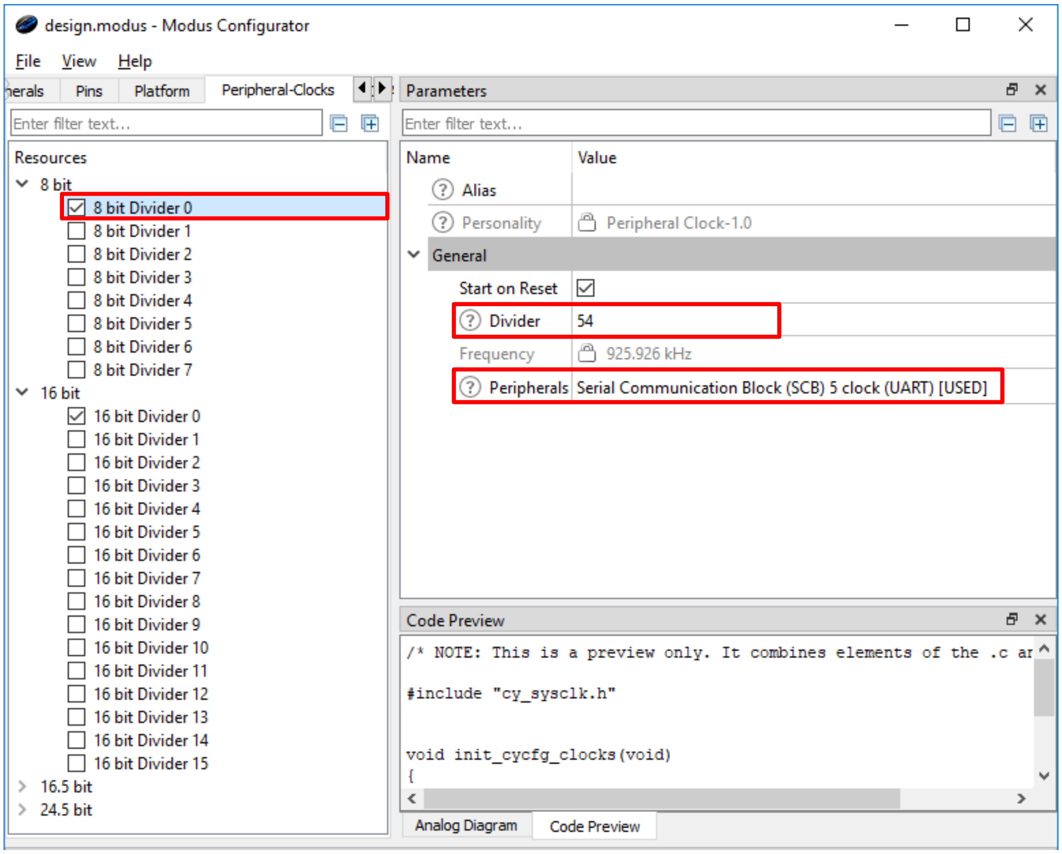
```
/* NOTE: This is a preview only. It combines elements

#include "cy_gpio.h"
#include "cycfg_connectivity.h"

#define UART_TX_PORT GPIO_PRT5
#define UART_TX_PIN 1U
```

Figure 6 and Figure 7 shows the Peripheral-Clock configuration for UART and TCPWM resources respectively.

Figure 6. Peripheral-Clock Configuration for UART


design.modus - Modus Configurator
File View Help
herals Pins Platform Peripheral-Clocks
Enter filter text...
Resources
Analog Diagram
Code Preview

- 8 bit
  - ☑ 8 bit Divider 0
  - □ 8 bit Divider 1
  - □ 8 bit Divider 2
  - □ 8 bit Divider 3
  - □ 8 bit Divider 4
  - □ 8 bit Divider 5
  - □ 8 bit Divider 6
  - □ 8 bit Divider 7
- 16 bit
  - ☑ 16 bit Divider 0
  - □ 16 bit Divider 1
  - □ 16 bit Divider 2
  - □ 16 bit Divider 3
  - □ 16 bit Divider 4
  - □ 16 bit Divider 5
  - □ 16 bit Divider 6
  - □ 16 bit Divider 7
  - □ 16 bit Divider 8
  - □ 16 bit Divider 9
  - □ 16 bit Divider 10
  - □ 16 bit Divider 11
  - □ 16 bit Divider 12
  - □ 16 bit Divider 13
  - □ 16 bit Divider 14
  - □ 16 bit Divider 15
- 16.5 bit
- 24.5 bit

Parameters

Name	Value
Alias	
Personality	Peripheral Clock-1.0
General	
Start on Reset	☑
Divider	54
Frequency	925.926 kHz
Peripherals	Serial Communication Block (SCB) 5 clock (UART) [USED]

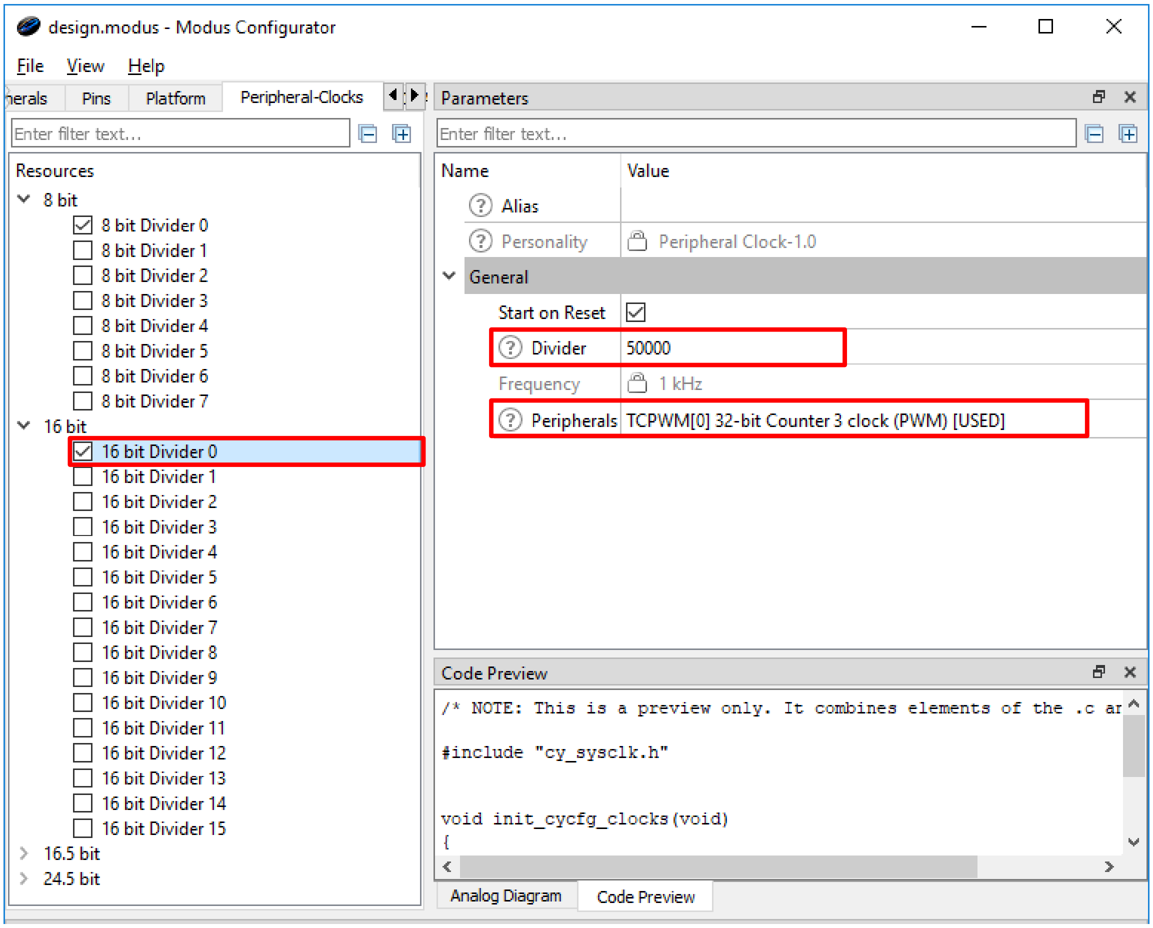
Code Preview

```
/* NOTE: This is a preview only. It combines elements of the .c ar

#include "cy_sysclk.h"


void init_cycfg_clocks(void)
{
```

Figure 7. Peripheral-Clock Configuration for PWM



Resources
- 8 bit
  - ☑ 8 bit Divider 0
  - □ 8 bit Divider 1
  - □ 8 bit Divider 2
  - □ 8 bit Divider 3
  - □ 8 bit Divider 4
  - □ 8 bit Divider 5
  - □ 8 bit Divider 6
  - □ 8 bit Divider 7
- 16 bit
  - ☑ 16 bit Divider 0
  - □ 16 bit Divider 1
  - □ 16 bit Divider 2
  - □ 16 bit Divider 3
  - □ 16 bit Divider 4
  - □ 16 bit Divider 5
  - □ 16 bit Divider 6
  - □ 16 bit Divider 7
  - □ 16 bit Divider 8
  - □ 16 bit Divider 9
  - □ 16 bit Divider 10
  - □ 16 bit Divider 11
  - □ 16 bit Divider 12
  - □ 16 bit Divider 13
  - □ 16 bit Divider 14
  - □ 16 bit Divider 15
- 16.5 bit
- 24.5 bit

Name	Value
Alias	
Personality	Peripheral Clock-1.0
General	
Start on Reset	☑
Divider	50000
Frequency	1 kHz
Peripherals	TCPWM[0] 32-bit Counter 3 clock (PWM) [USED]

```
/* NOTE: This is a preview only. It combines elements of the .c ar

#include "cy_sysclk.h"


void init_cycfg_clocks(void)
{
```

Table 2 shows the pin assignments for the project.

Table 2. Pin Assignments

Name	Port
LED	P1[1]
UART_RX	P5[0]
UART_TX	P5[1]

Debugging

You can debug the example to step through the code. If you are unfamiliar with how to start a debug session on the PSoC 6 MCU with Modus IDE, see KBA224621 in the Cypress community.

Reusing This Example

This example is designed for the PSoC 6-WIFI-BT Pioneer Kit.

In some cases, a resource used by a code example (for example, an IP block) is not supported on another device. In that case the example will not work. If you build the code targeted at such a device, you will get errors. See the device datasheet for information on which resources the device supports.

Related Documents

For a comprehensive list of PSoC 6 MCU resources, see KBA223067 in the Cypress community.

Application Notes	
AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 MCU with BLE Connectivity devices and how to build your first PSoC Creator project
AN215656 – PSoC 6 MCU: Dual-CPU System Design	Describes the dual-CPU architecture in PSoC 6 MCU, and shows how to build a simple dual-CPU design
Code Examples	
CE212736 – PSoC 6 MCU BLE Find Me	CE218133 – PSoC 6 MCU E-Ink Display with CapSense
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	
CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	
Tool Documentation	
ModusToolbox	Join the Early Access Program for links to the latest tools
Cypress SDK	Installed with ModusToolbox

Cypress Resources

Cypress provides a wealth of data at www.cypress.com to help you to select the right device, and quickly and effectively integrate the device into your design.

For the PSoC 6 MCU devices, see KBA223067 in the Cypress community for a comprehensive list of PSoC 6 MCU resources.

Document History

Document Title: CE223541 – PSoC 6 MCU: Hello World using ModusToolbox

Document Number: 002-23541

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	nnnnnnn	SNVN	07/19/2018	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 MCU

Cypress Developer Community

Community 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

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.