

## Objective

This code example demonstrates generating a One-Time Password (OTP) using the True Random Number generation feature of PSoC® 6 MCU cryptography block.

## Overview

This example demonstrates generating a One-Time-Password (OTP) of eight characters in length. Using the True Random Number generation feature of PSoC 6 MCU crypto block, a random number corresponding to each character of the OTP is generated. The generated random number is such that it corresponds to alpha-numeric and special characters of the ASCII code. The generated OTP is then displayed on a UART terminal emulator.

## Requirements

**Tool:** PSoC Creator™ 4.2; Peripheral Driver Library (PDL) 3.0.1

**Programming Language:** C (Arm® GCC 5.4.1 and Arm MDK 5.22)

**Associated Parts:** All PSoC 6 MCU parts

**Related Hardware:** CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit

## Hardware Setup

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

## Software Setup

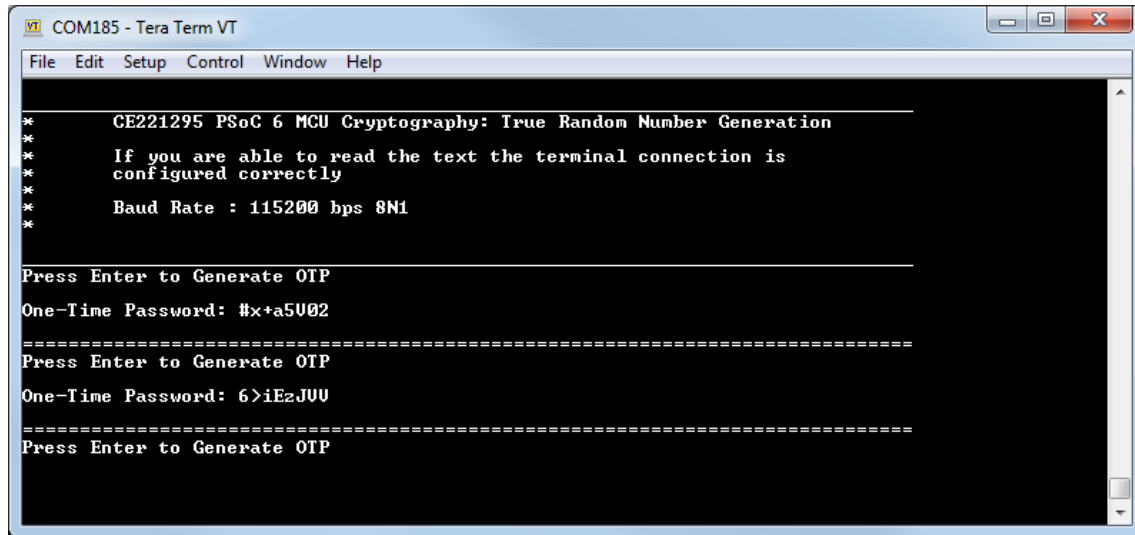
This example uses Tera Term as the UART terminal for displaying the generated OTP. Set the UART configuration settings as the same as that used by the UART SCB on PSoC 6 MCU.

## Operation

1. Plug the CY8CKIT-062-BLE kit board into your computer's USB port.
2. Build the project and program it into the PSoC 6 MCU device. Choose **Debug > Program**. For more information on device programming, see PSoC Creator Help. Flash for both CPUs is programmed in a single program operation.  
**Note:** If during the build process, if the PSoC Creator prompts you to replace *stdio\_user.h* file, **DO NOT** replace the file.
3. Open Tera Term and connect to the "KitProg2 USB-UART" bridge COM port. Set the baud rate as 115200 bps.
4. Press the **Enter** key to generate an OTP. The generated OTP will be displayed on the UART terminal. Note that you must press the **Enter** key every time when you need to generate an OTP.

Figure 1 shows a sample output as displayed on Tera Term UART Terminal.

Figure 1. Sample Output as Displayed on Tera Term



```

COM185 - Tera Term VT
File Edit Setup Control Window Help

*      CE221295 PSoC 6 MCU Cryptography: True Random Number Generation
*
*      If you are able to read the text the terminal connection is
*      configured correctly
*
*      Baud Rate : 115200 bps 8N1
*

Press Enter to Generate OTP
One-Time Password: #x+a5U02
=====
Press Enter to Generate OTP
One-Time Password: 6>iEzJU0
=====
Press Enter to Generate OTP
  
```

## Design and Implementation

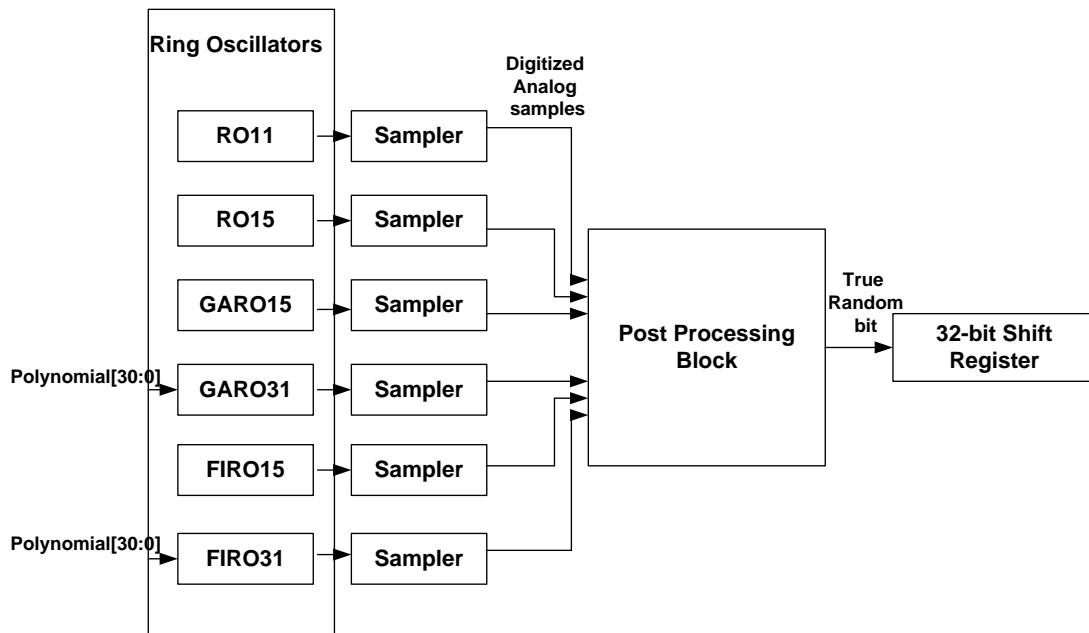
Random number generation is the generation of a sequence of numbers or symbols that cannot be predicted based on the previous knowledge of the generated sequence. Random number generators have applications in cryptography, statistical sampling, gambling and other areas where producing an unpredictable result is desirable.

A true random number is generated using a hardware random number generator that generates random numbers from a physical process. The true random number generator (TRNG) in PSoC 6 MCU generates true random numbers of programmable bit size ranging from 0–32 bits. The TRNG relies on up to six ring oscillators to provide physical noise sources namely:

- Two fixed ring oscillators consisting of 11 and 15 inverters (RO11 and RO15).
- A fixed Galois-based ring oscillator (GAR015) and a fixed Fibonacci-based ring oscillator (FIRO15) each consisting of 15 inverters.
- A flexible Galois-based (GAR031) and a flexible Fibonacci-based oscillator (FIRO31) consisting of 31 inverters with a programmable polynomial of up to order 31.

A ring oscillator consists of a series of inverters connected in a feedback loop to form a ring. Due to (temperature) sensitivity of the inverter delays, jitter is introduced on a ring's oscillating signal. The jittered oscillating signal is sampled to produce a digitized analog signal (DAS). This is done for all multiple ring oscillators. To increase entropy and to reduce bias in DAS bits, the DAS bits are further post-processed. Post-processing produces bit samples that are considered true random bit samples. The true random bit samples are shifted into a register to provide random values of up to 32 bits. [Figure 2](#) shows an overview of how generation of true random generation is implemented in PSoC 6 MCU.

Figure 2. True Random Number Generation in PSoC 6 MCU



Cryptography in PSoC 6 MCU is based on a Client-Server model. The firmware initializes and starts the Crypto server. The server runs only on the CM0+ core, and works with the crypto hardware. The Crypto client can run on either core. In this example, the client runs on the CM4 core. The firmware initializes and starts the client. The firmware then provides the configuration data required for generation of true random number and requests the crypto server to run the cryptographic operation.

In this example, an OTP of eight characters in length is generated. Using the true random number generator, a random number is generated corresponding to each character of the OTP. The generated random number is such that it corresponds to alpha-numeric and special characters of the ASCII code. The generated OTP is then displayed on a UART terminal emulator. Each time, the firmware waits for the user to press the “Enter” key to generate a new OTP.

Figure 3. Firmware Flowchart

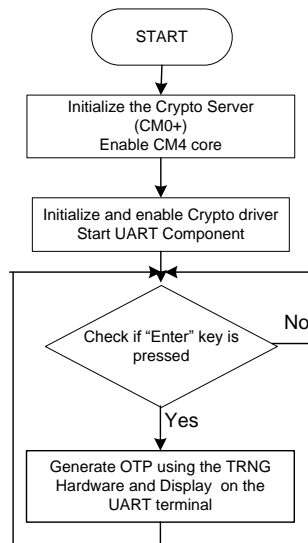
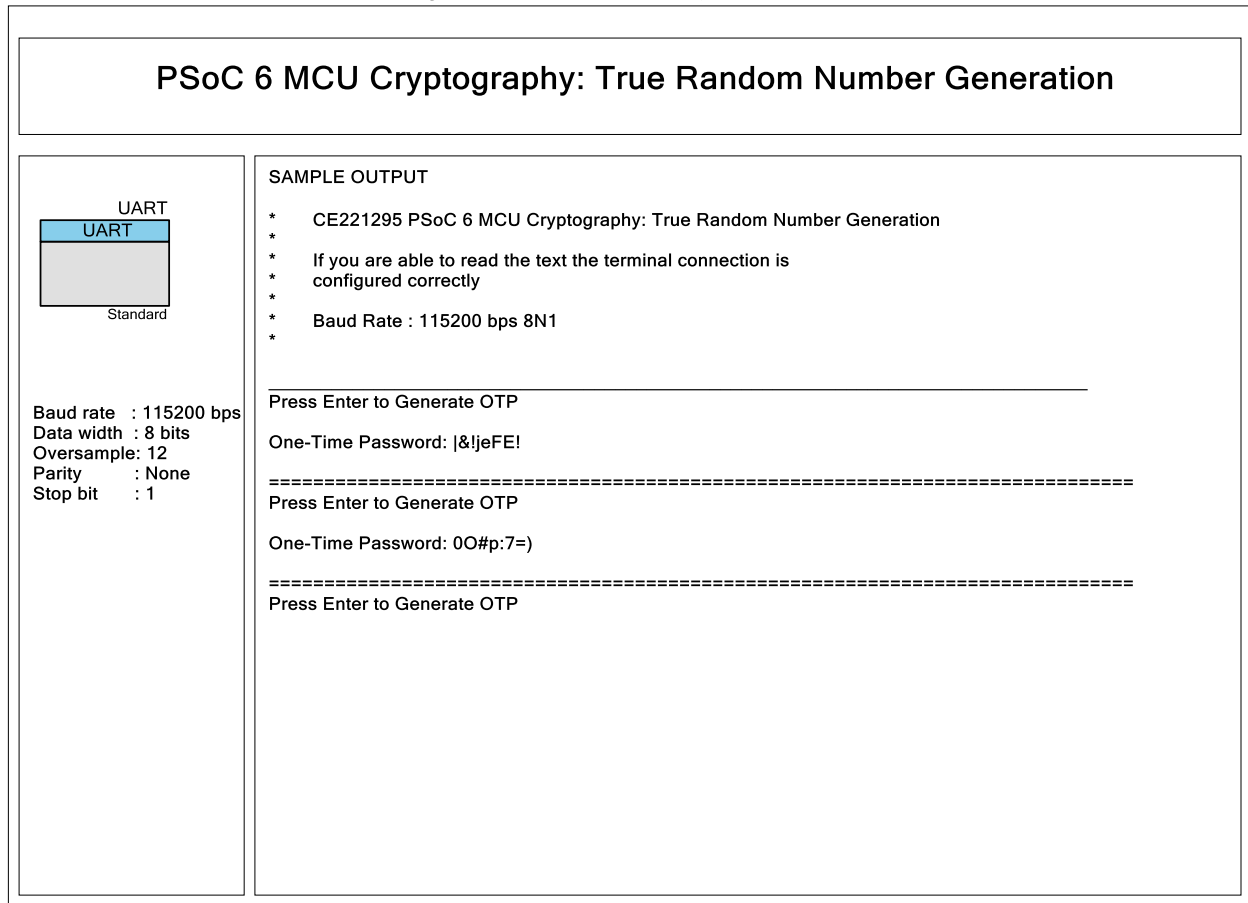


Figure 4. PSoC Creator Project Schematic



## Components and Settings

Table 1 lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 1. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
UART	UART	Facilitates printing on to a UART terminal	Default settings only

For information on the hardware resources used by a Component, see the Component datasheet.

In order to use the Crypto block of PSoC 6 MCU in your design, the Crypto driver must be enabled. To enable the drivers, check the crypto option under **Project > Build Settings > Peripheral Driver Library** as shown in Figure 5.

Figure 5. Enabling Crypto PDL Drivers

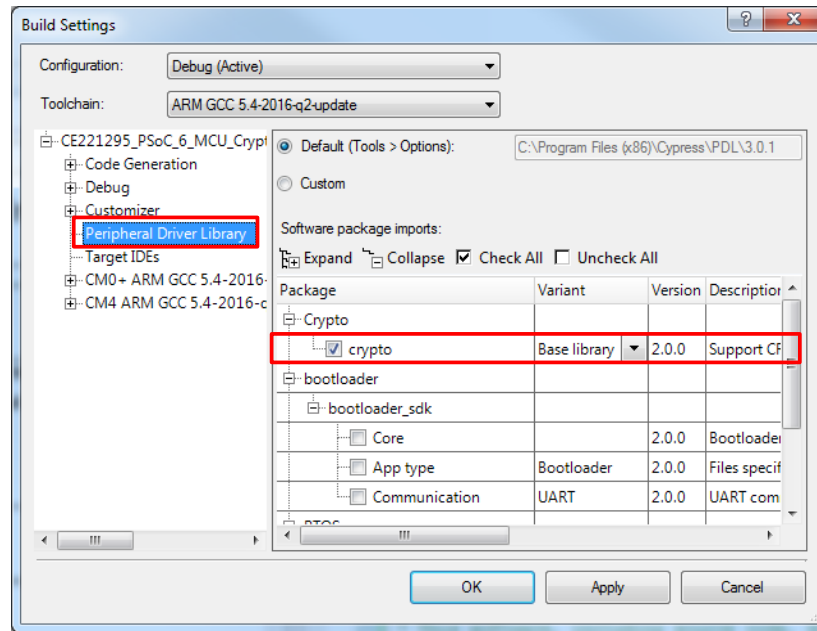




Figure 6 shows the pin assignment for the project done through the **Pins** tab in the **Design Wide Resources** window. This example uses the Kitprog2 USB-UART bridge to communicate with UART terminal emulator running on your PC. CY8CKIT-062-BLE Pioneer kit uses the pin **P5[0]** as UART Rx pin and pin **P5[1]** as UART Tx pin.

Figure 6. Device Pin Assignments

	Name	Port	Pin	Lock
	\UART: rx\	P5 [0]	L6	<input checked="" type="checkbox"/>
	\UART: tx\	P5 [1]	K6	<input checked="" type="checkbox"/>

## Reusing This Example

This example is designed for the CY8CKIT-062-BLE Pioneer Kit. To port the design to a different PSoC 6 MCU device and/or kit, change the target device using Device Selector and update the pin assignments in the Design-Wide Resources Pins settings as needed.

## Related Documents

Application Notes	
<a href="#">AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity</a>	Describes PSoC 6 MCU with BLE Connectivity devices and how to build your first PSoC Creator project
PSoC Creator Component Datasheets	
<a href="#">UART</a>	Supports standard UART interface
Device Documentation	
<a href="#">PSoC 6 MCU: PSoC 63 with BLE Datasheet</a>	<a href="#">PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual</a>
Development Kit Documentation	
<a href="#">CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit</a>	

## Document History

Document Title: CE221295 – PSoC 6 MCU Cryptography: True Random Number Generation

Document Number: 002-21295

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6006857	VKVK	12/27/2017	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	<a href="http://cypress.com/arm">cypress.com/arm</a>
Automotive	<a href="http://cypress.com/automotive">cypress.com/automotive</a>
Clocks & Buffers	<a href="http://cypress.com/clocks">cypress.com/clocks</a>
Interface	<a href="http://cypress.com/interface">cypress.com/interface</a>
Internet of Things	<a href="http://cypress.com/iot">cypress.com/iot</a>
Memory	<a href="http://cypress.com/memory">cypress.com/memory</a>
Microcontrollers	<a href="http://cypress.com/mcu">cypress.com/mcu</a>
PSoC	<a href="http://cypress.com/psoc">cypress.com/psoc</a>
Power Management ICs	<a href="http://cypress.com/pmic">cypress.com/pmic</a>
Touch Sensing	<a href="http://cypress.com/touch">cypress.com/touch</a>
USB Controllers	<a href="http://cypress.com/usb">cypress.com/usb</a>
Wireless Connectivity	<a href="http://cypress.com/wireless">cypress.com/wireless</a>

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

### 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](http://cypress.com/support)



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. 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](http://cypress.com). Other names and brands may be claimed as property of their respective owners.