

# Arm® TRNG

Revision: r0p0

## Characterization Application Note

The Arm logo, consisting of the word "arm" in a bold, lowercase, sans-serif font.

## Arm® TRNG

### Characterization Application Note

Copyright © 2015, 2016, 2018–2020 Arm Limited or its affiliates. All rights reserved.

#### Release Information

#### Document History

Issue	Date	Confidentiality	Change
00	27 July 2015	Confidential	First official release (v1.0).
01	08 October 2015	Confidential	Second release (v1.1).
02	22 November 2015	Confidential	Third release (v1.2).
03	27 January 2016	Confidential	Fourth release (v1.3).
04	17 May 2016	Confidential	Fifth release (v1.4).
05	08 November 2016	Confidential	Sixth release.
06	09 January 2018	Non-Confidential	Seventh release.
07	06 June 2018	Non-Confidential	Eighth release.
08	06 September 2018	Non-Confidential	Ninth release.
09	16 January 2019	Non-Confidential	Tenth release.
10	05 February 2020	Non-Confidential	Eleventh release.

#### Non-Confidential Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. **No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.**

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED “AS IS”. ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if

there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm's trademark usage guidelines at <http://www.arm.com/company/policies/trademarks>.

Copyright © 2015, 2016, 2018–2020 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

### **Confidentiality Status**

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

### **Product Status**

The information in this document is Final, that is for a developed product.

### **Web Address**

[www.arm.com](http://www.arm.com)

# Contents

## Arm® TRNG Characterization Application Note

### **Preface**

About this book .....	6
Additional reading .....	8

### **Chapter 1**

#### **Overview of Arm® TRNG**

1.1 TRNG characterization .....	1-10
1.2 Compliance .....	1-11

### **Chapter 2**

#### **TRNG characterization procedure**

2.1 Characterization procedure overview .....	2-13
2.2 Characterization test program .....	2-14
2.3 Characterization test conditions .....	2-15
2.4 Base iteration .....	2-17
2.5 First characterization iteration .....	2-18
2.6 Second characterization iteration .....	2-20
2.7 Restart tests iteration .....	2-22

### **Appendix A**

#### **CC\_TST\_TRNG output**

A.1 CC_TST_TRNG output format .....	Appx-A-24
-------------------------------------	-----------

### **Appendix B**

#### **Revisions**

B.1 Revision history .....	Appx-B-26
----------------------------	-----------

# Preface

This preface introduces the *Arm® TRNG Characterization Application Note*.

It contains the following:

- *About this book* on page 6.
- *Additional reading* on page 8.

## About this book

This book describes the characterization process for the Arm True Random Number Generator (TRNG).

## Using this book

This book is organized into the following chapters:

### **Chapter 1 Overview of Arm® TRNG**

This chapter provides an overview of the Arm TRNG and its characterization.

### **Chapter 2 TRNG characterization procedure**

This chapter provides the detailed Arm characterization procedure.

### **Appendix A CC\_TST\_TRNG output**

This appendix describes the format of CC\_TST\_TRNG output.

### **Appendix B Revisions**

This appendix describes the technical changes between released issues of this book.

## Glossary

The Arm® Glossary is a list of terms used in Arm documentation, together with definitions for those terms. The Arm Glossary does not contain terms that are industry standard unless the Arm meaning differs from the generally accepted meaning.

See the [Arm® Glossary](#) for more information.

## Typographic conventions

### *italic*

Introduces special terminology, denotes cross-references, and citations.

### **bold**

Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

### `monospace`

Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

### monospace

Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

### `monospace italic`

Denotes arguments to monospace text where the argument is to be replaced by a specific value.

### `monospace bold`

Denotes language keywords when used outside example code.

### <and>

Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

### SMALL CAPITALS

Used in body text for a few terms that have specific technical meanings, that are defined in the *Arm® Glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

## Feedback

## Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

## Feedback on content

If you have comments on content then send an e-mail to [errata@arm.com](mailto:errata@arm.com). Give:

- The title *Arm TRNG Characterization Application Note*.
- The number 100685\_0000\_10\_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

Arm also welcomes general suggestions for additions and improvements.

---

### Note

Arm tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

---

## Other information

- [Arm® Developer](#).
- [Arm® Information Center](#).
- [Arm® Technical Support Knowledge Articles](#).
- [Technical Support](#).
- [Arm® Glossary](#).

## Additional reading

This book contains information that is specific to this product. See the following documents for other relevant information.

**Table 1 Other Publications**

Document ID	Document name
ANSI X9.31-1988	Public Key Cryptography Using Reversible Algorithms for the Financial Services Industry (rDSA)
BSI AIS-31	Functionality Classes and Evaluation Methodology for True Random Number Generators
FIPS Publication 140-2	Security Requirements for Cryptographic Modules
NIST SP 800-90A	Recommendation for Random Number Generation Using Deterministic Random Bit Generators – App C.
NIST SP 800-90B	Recommendation for the Entropy Sources Used for Random Bit Generation



# Chapter 1

## Overview of Arm® TRNG

This chapter provides an overview of the Arm TRNG and its characterization.

Arm TRNG collects entropy from a physical entropy source (a component capable of generating an unpredictable or random output bit stream). The collected entropy is used to seed the cryptographic random bits generator with a secure initial state.

————— **Note** —————

Usually, the physical process used for collecting entropy is an inverter timing jitter that is collected from a dedicated on-chip free-running ring oscillator.

The TRNG can be used in one of two modes, each requiring a different driver:

- The operating mode of the Arm implementation of the FE TRNG driver is compliant with the *BSI AIS-31 Functionality Classes and Evaluation Methodology for True Random Number Generators* standard, as a true random number generator that outputs full-entropy bits at a relatively low rate.
- The operating mode of the Arm implementation of the 800-90B TRNG driver is compliant with the *NIST SP 800-90B Recommendation for the Entropy Sources Used for Random Bit Generation* standard, as a true random number generator.

It contains the following sections:

- [1.1 TRNG characterization on page 1-10.](#)
- [1.2 Compliance on page 1-11.](#)

## 1.1 TRNG characterization

Arm True Random Number Generator (TRNG) configuration parameters specify the settings of the internal ring-oscillator lengths, and the output sampling rate. The parameters are device-specific.

Each silicon process has different noise and jitter characteristics. The specific SoC layout affects these characteristics. Therefore, the TRNG behavior must be characterized on the actual silicon of the device to determine the most suitable parameters. Characterizing in this way ensures that the TRNG output has maximal entropy.

Characterization must be performed during the initial post-silicon testing of the device, or whenever substantial changes are made. For example, after changes to process or respins.

## 1.2 Compliance

Arm TRNG complies with the following specifications:

**Table 1-1 Arm TRNG compliance**

Document ID	Document name	Compliance
BSI AIS-31	Functionality Classes and Evaluation Methodology for True Random Number Generators	Compliant in an implementation using FETRNG driver with class PTG.2.
NIST SP 800-90B	Recommendation for the Entropy Sources Used for Random Bit Generation	Compliant with section 4.4 Approved Continuous Health Tests.

# Chapter 2

## TRNG characterization procedure

This chapter provides the detailed Arm characterization procedure.

It contains the following sections:

- [2.1 Characterization procedure overview on page 2-13.](#)
- [2.2 Characterization test program on page 2-14.](#)
- [2.3 Characterization test conditions on page 2-15.](#)
- [2.4 Base iteration on page 2-17.](#)
- [2.5 First characterization iteration on page 2-18.](#)
- [2.6 Second characterization iteration on page 2-20.](#)
- [2.7 Restart tests iteration on page 2-22.](#)

## 2.1 Characterization procedure overview

The characterization procedure requires two iterations between the partner (you) and Arm.

In each iteration, you perform a series of characterization tests, and send the resulting data to Arm. Arm analyzes the results, and returns the best Arm settings to you.

————— **Note** —————

If you use the 800-90B mode, it is your responsibility to analyze the results of the second iteration (including the restart tests), using the tool provided on the NIST website. For more information, see [2.6 Second characterization iteration on page 2-20](#).

**Table 2-1 Characterization high-level procedure steps**

Step	Owner	Execution	Comments
Prepare characterization test program.	Partner	Prepare a characterization test program that uses the CC_TST_TRNG routine that Arm supplies.	See <a href="#">2.2 Characterization test program on page 2-14</a> and <a href="#">2.3 Characterization test conditions on page 2-15</a> .
Base iteration: Find the minimal sample count.	Partner	Run the preliminary test to establish the minimal (base) value of the sample count.	See <a href="#">2.4 Base iteration on page 2-17</a> .
First iteration: Run first set of characterization tests.	Partner	Run a series of characterization tests under multiple test conditions.	See <a href="#">2.5 First characterization iteration on page 2-18</a> . Send results to Arm.
First iteration: Analyze first characterization test results.	Arm	Runs a set of statistical tests on the characterization output data.	Sends the partner the TRNG configuration parameters, and a set of corners to run the tests on.
Second iteration: Run second set of characterization tests.	Partner	1. Use CC_TST_TRNG with the configuration parameters that were received in the first iteration. 2. Run the characterization tests in worst-case conditions, as provided by Arm.	See <a href="#">2.6 Second characterization iteration on page 2-20</a> . Partners using AIS-31 mode, must send the results to Arm.
Second iteration: Analyze second characterization test results.	Arm (AIS-31 mode)	Runs a set of statistical tests on the characterization output data. Use this analysis to generate the mass production Arm TRNG configuration parameters.	Sends the mass production TRNG configuration parameters to the partner. ————— <b>Note</b> ————— If the results are not good enough, repeat the second iteration. —————
	Partner (800-90B mode)	Analyze the results using NIST tools.	————— <b>Note</b> ————— If the results are not good enough, repeat the second iteration. —————
Restart tests iteration	Partner (800-90B mode)	Re-run tests using data from an entropy source that produces outputs for real-world use.	See <a href="#">2.7 Restart tests iteration on page 2-22</a> .

## 2.2 Characterization test program

You must implement the characterization test program, using the CC\_TST\_TRNG API that Arm supplies.

The CC\_TST\_TRNG API is included in <prefix>-TRNG\_Characterization.c, where <prefix> represents the product-specific part number and version. For the macros used in following sections, refer to <prefix>-TRNG\_Characterization.c.

---

### Note

---

Usually, for a memory-constrained IoT scenario, the collected raw data is redirected to a peripheral, to reduce memory usage. For a device without an available peripheral, you can utilize the max delay between blocks to design a private protocol to collect the data. The max delay between blocks follows this formula:  $\text{max\_delay} = 4 * (192 * \text{SAMPLE\_CNT} / \text{RNG\_CLK})$ .

---

### Example of API

```
/*
 * Collect TRNG output for characterization
 *
 * @regBaseAddress: The base address of TRNG registers in system memory map
 * @TRNGMode:       Base iteration - TRNG_MODE_FAST
 *                 First iteration - TRNG_MODE_FAST
 *                 Second iteration - TRNG_MODE_FE compliant with BSI AIS-31
 *                 TRNG_MODE_80090B compliant with NIST SP 800-90B
 *
 * @roscLength:     Ring oscillator length (0 to 3)
 * @sampleCount:    Ring oscillator sample counter value
 * @buffSize:       Total buffer size covered header, sample_bits and footer.
 *                 Must be between 52 and 2^24 bytes.
 *                 buffsize >= header(16 bytes)+sample_bits/8 (bytes)+footer(12 bytes).
 *                 For example, if 100Mbits of data need to be collected, the minimal
 *                 buffSize should be 12,500,000+28=12,500,028 bytes. To be safe, the
 *                 buffSize can be 12,501,000 bytes.
 *
 * @callbackFunc:   Callback function to process the real output data
 *                 This callbackFunc is called many times in CC_TST_TRNG.
 *                 First, it is called after the 16-byte header is generated.
 *                 Then, it is called repeatedly when the 24-byte EHR registers are
 *                 full.
 *                 Finally, it is called after the 12-byte footer is generated.
 * @return          0 on success. Non-zero value on failure.
 */
int CC_TST_TRNG( unsigned long regBaseAddress,
                 uint32_t TRNGMode,
                 uint32_t roscLength,
                 uint32_t sampleCount,
                 uint32_t buffSize,
                 callback_TRNG callbackFunc);
```

### Definition of callback\_TRNG

```
/*
 * Callback function type define
 * The partner must implement this function according the system resource.
 * This function is called by CC_TST_TRNG() to process the data generated
 * or collected by CC_TST_TRNG().
 * @outputSize      The size of the output data
 * @outputBuffer    The buffer of the data
 */
typedef void (*callback_TRNG)(uint32_t outputSize, uint8_t *outputBuffer);
```

---

### Note

---

Partners must implement the callback function according to the system resource.

---

## 2.3 Characterization test conditions

Each characterization test is executed by running the characterization test program under a combination of conditions.

These tests are described in [Table 2-2 Characterization test conditions on page 2-15](#).

It is critical that for each test:

- If using FE TRNG driver:
  - All output bits must be collected using a single contiguous execution of the test.
  - If any bits are dropped and not captured in the output file, you must rerun the test as the statistical analysis of the output is meaningless.
  - If the system does not have sufficient memory to collect all required bits in a single run, you can split each test into multiple runs. For example 100 consecutive runs, each collecting 1Mbits.
- If using 800-90B TRNG driver:
  - The output bits can be non-contiguous, and concatenation of several smaller sets of consecutive samples (generated using the same noise source) is allowed.
  - Smaller sets must contain at least 1000 samples.
  - The concatenated dataset must contain at least 1,000,000 samples.
- For both drivers, All the resulting bits are saved in the output file without any gaps.

If any bits are dropped and not captured in the output file, the test must be rerun as the statistical analysis of the output is meaningless.

**Table 2-2 Characterization test conditions**

Configuration variable	Operating conditions	Filename values
Ring oscillator length.	The four configurable lengths that Arm TRNG allows.	R0, R1, R2, R3. R0 is the shortest length and R3 is the longest.  ————— <b>Note</b> ————— In some chips, the ring oscillator might not properly work when configured to the shorter lengths. —————
Voltage.	High, typical, low.	VH, VT, VL.
Temperature.	High, typical, low.	TH, TT, TL.
CMOS process corner.	Typical, fast/fast, fast/slow, slow/fast, slow/slow.	CT, CFF, CFS, CSF, CSS.

This section contains the following subsection:

- [2.3.1 Output-file names on page 2-15](#).

### 2.3.1 Output-file names

Output-files are named according to a standard format.

The output file for each characterization test is named according to the Filename Values column in [Table 2-2 Characterization test conditions on page 2-15](#). Output filename format is `trng_samples_R*_S*_V*_T*_C*.bin`.

For example, for the following characterization test:

- The longest ring oscillator length.
- Sample counter value of five.
- High voltage.

- Low temperature.
- Fast or slow CMOS corner.

The filename is `trng_samples_R3_S5_VH_TL_CFS.bin`.



## 2.4 Base iteration

The base iteration is used to find the minimum sample counter value for which Arm TRNG operates properly, under typical operating conditions.

The typical operating conditions are:

- The second-longest ring-oscillator.
- Typical voltage.
- Typical temperature.
- Typical process corner.

The minimum sample counter value is found by calling `CC_TST_TRNG` with increasing values of `sampleCount` (starting with 1) until it exits successfully.

At least 200Kbits (2.0e5 bits) must be collected for the base iteration and the first characterization iteration.

---

### Note

---

In many systems, the test succeeds immediately (with `sampleCount=1`), which is an expected and even desirable result.

---

### Sample code for data type, macro, and callback function definition

```
typedef unsigned char uint8_t;
typedef unsigned int uint32_t;
/*
 * Define the buffer size for the base iteration.
 * At least 200 Kbits(25 KBytes) must be collected
 * every time EHR registers output 192 bits (24Bytes) value.
 * Therefore, set the collected data as 25008 bytes (25008/24 = 1042).
 * The full buffer size is 25008+28(header+footer)=25036 bytes.
 */
#define BASE_ITERATION_BUF_LEN 25036 //0x61CC
static uint8_t data_buf[BASE_ITERATION_BUF_LEN];
static uint32_t buffer_index = 0;

static void callback_BaseIteration(uint32_t outputSize, uint8_t *outputBuffer)
{
    if (NULL == outputBuffer)
    {
        printf("invalid input parameter!\r\n");
    }

    memcpy(data_buf+buffer_index, outputBuffer, outputSize);
    buffer_index += outputSize;
}
```

### Sample code for finding minimum sample counter value

The minimum sample counter value is based on the macros that are defined in the [Sample code for data type, macro, and callback function definition on page 2-17](#), and the following `callback_BaseIteration` value.

```
int ret = 0;
uint32_t i = 0;
uint32_t sampleCounter = 0;

for (i=1; ;i++)
{
    ret = CC_TST_TRNG(g_CcBaseAddr,          // regBaseAddress
                     0,                      // TRNGMode
                     2,                      // roscLength
                     i,                      // sampleCount
                     BASE_ITERATION_BUF_LEN, // buffSize
                     callback_BaseIteration); // callback

    buffer_index = 0;

    if (ret == 0)
        break;
}
sampleCounter = i;
```

## 2.5 First characterization iteration

Perform this procedure for each of the characterization test conditions combinations.

### Procedure

1. Set up the test operating conditions.
2. Run CC\_TST\_TRNG under these conditions with each of the following sampleCount values:
  - First set:  $S_{SMP1}$  = The minimal sampleCount found in [2.4 Base iteration on page 2-17](#).
  - Second set:  $S_{SMP2}$  = Ceiling( $1.5 * (S_{SMP1}+1)-1$ ).
  - Third set:  $S_{SMP3}$  = Ceiling( $1.5 * (S_{SMP2}+1)-1$ ).

For example, if the minimum value of sampleCount is 8, then run CC\_TST\_TRNG under each of the 180 conditions with:

- SMP1=8 in the first set.
- SMP2=13 in the second set.
- SMP3=20 in the third set.

Overall, you must run a total of 540 characterization tests: (3 (sample counter values) \* 4 (ring oscillator configurations) \* 3 (voltages) \* 3 (temperatures) \* 5 (process corners).

You must export all of the data that is collected as part of the first characterization iteration from the device, using printData.

The estimated total time for the characterization procedure is between 16-18 hours: 5 chips \* 3 temperature conditions \* (30 minutes to set each chip in each temperature condition + [30-40] minutes for running all oscillator, voltage, and sample count combinations in this temperature).

3. Save the results of each test run in the relevant output file that is named according to [2.3.1 Output-file names on page 2-15](#).
  - A set of four sample count values – one value for each ring oscillator length.
  - A definition of the worst-case corner (voltage/temperature/process) to be used for the second characterization iteration.
  - When the first characterization iteration is complete, you can expect to have 540 files that are named according to the different test conditions.

### Example 2-1 Sample code for the first charcterization iteration

In this example, SMP1=1, SMP2=2 and SMP3=4 and callback\_FirstIteration is used as the callback\_BaseIteration for the base iteration.

```
void first_iteration_test()
{
    uint32_t TRNGMode = 0;
    uint32_t roscLength = 0;
    uint32_t sampleCounter = 1;
    int ret = 0;

    for (roscLength=0; roscLength<4; roscLength++)
    {
        sampleCounter = 1;
        ret = CC_TST_TRNG(g_CcBaseAddr, TRNGMode, roscLength, sampleCounter,
FIRST_ITERATION_BUF_LEN, callback_FirstIteration);
        printData(ret, roscLength, sampleCounter);
        buffer_index = 0;
        memset(data_buf, 0, sizeof(data_buf));

        sampleCounter = 2;
        ret = CC_TST_TRNG(g_CcBaseAddr, TRNGMode, roscLength, sampleCounter,
FIRST_ITERATION_BUF_LEN, callback_FirstIteration);
        printData(ret, roscLength, sampleCounter);
        buffer_index = 0;
        memset(data_buf, 0, sizeof(data_buf));

        sampleCounter = 4;
    }
}
```

```
ret = CC_TST_TRNG(g_CcBaseAddr, TRNGMode, roscLength, sampleCounter,  
FIRST_ITERATION_BUF_LEN, callback_FirstIteration);  
printData(ret, roscLength, sampleCounter);  
buffer_index = 0;  
memset(data_buf, 0, sizeof(data_buf));  
}  
}
```

---

## Next Steps

1. You must send all of the collected files that you exported from the device to Arm.
2. Arm analyzes these files and responds with the feedback of the analysis.

## 2.6 Second characterization iteration

You must run a second set of characterization tests to determine the TRNG configuration parameters.

### Procedure

1. Run four characterization tests for each of the corners that Arm identifies as result of the first iteration.

Call CC\_TST\_TRNG with each of the four ring oscillator lengths (each with its corresponding sample count).

Depending on the intended TRNG driver, each call to CC\_TST\_TRNG must be as follows:

- FE TRNG driver: Call CC\_TST\_TRNG with TRNGMode=1 and collect 100Mbit (12.5MB).
  - All output bits must be collected using a single contiguous execution of the test.
  - All resulting bits must be saved in the output file without any gaps.
  - If any bits are dropped and not captured in the output file, you must rerun the test as the statistical analysis of the output is meaningless.
  - If the system does not have sufficient memory to collect all required bits in a single run, you can split each test into multiple runs. For example 100 consecutive runs, each collecting 1Mbits.
- 800-90B TRNG driver: CC\_TST\_TRNG with TRNGMode=2 and collect 1Mbit.
  - The output bits can be non-contiguous, and concatenation of several smaller sets of consecutive samples (generated using the same noise source) is allowed.
  - Smaller sets must contain at least 1000 samples.
  - The concatenated dataset must contain at least 1,000,000 samples.
  - All resulting bits must be saved in the output file without any gaps.

The same output file naming rules apply for both drivers. For more information, see [2.3.1 Output-file names on page 2-15](#).

2. Process the resulting output files as follows:
  - If you ran the characterization in AIS-31 mode, then you must send the resulting output files to Arm for statistical analysis.
  - If you ran the characterization in 800-90 mode, then you must proceed and analyze the results using NIST tools, as published in the NIST site.

**Results:** In either case, the returned results confirm or refute the TRNG configuration that is used for mass production. After the [2.7 Restart tests iteration on page 2-22](#), these configuration values must be updated in the relevant TRNG driver files.

#### ————— **Note** —————

If there are errors, you must repeat this iteration, or parts of it (some of the ring oscillator lengths) until a full set is obtained.

### Example 2-2 Sample code of the second characterization test that is based on the FE TRNG driver

#### Example 2-2 Second characterization test sample code

The following sample code includes the macro and callback function definitions.

```
/*
 * define the buffer size for second iteration
 * e.g. at least 100 Mbits(12.5 MBytes) need to be collected
 * every time EHR registers output 192bits(24Bytes) value
 * so set the collected data as 1250016bytes (1250016/24 = 52084)
 * the whole buffer size is 1250016+28(header+footer)=1250044bytes
 */
```

```
#define SECOND_ITERATION_BUFF_LEN      (12500044)

static void callback_SecondIteration(uint32_t outputSize, uint8_t *outputBuffer)
{
    int i = 0;
    if (NULL == outputBuffer)
    {
        printf("invalid input parameter!\r\n");
    }
    for (i=0; i<outputSize; i++)
    {
        printf("%02x", *(outputBuffer+i));
    }
}
```

The following sample code is based on callback\_SecondIteration.

```
void second_iteration_test()
{
    uint32_t TRNGMode = 1;
    uint32_t roscLength = 0;
    uint32_t sampleCounter = 1;
    int ret = 0;
    uint32_t i = 0;

    for (roscLength=0; roscLength<4; roscLength++)
    {
        /*
        * The following line must be updated for your board.
        * It is based on Musca-B1 CFS test boards.
        * Only roscLength 0 and 2 have worst condition on CFS board
        */
        if ((roscLength == 1) || (roscLength == 3))
            continue;
        /*
        * The sample counter values in the following switch statement must
        * be updated according the real cases. The following are the values
        * for the tested Musca-B1 board.
        */
        switch (roscLength)
        {
            case 0:
                sampleCounter = 360;
                break;
            case 1:
                sampleCounter = 420;
                break;
            case 2:
                sampleCounter = 600;
                break;
            case 3:
                sampleCounter = 620;
                break;
        }
        printf("\r\nroscLength=%d, sampleCount=%d\r\n", roscLength, sampleCounter);
        ret = CC_TST_TRNG(g_CcBaseAddr, TRNGMode, roscLength, sampleCounter,
SECOND_ITERATION_BUFF_LEN, callback_SecondIteration);
        printf("\r\nret = %d\r\n", ret);
    }
}
```

## 2.7 Restart tests iteration

To verify the values that are used in production, perform the following procedure:

### Prerequisites

- This iteration only applies to the 800-90B TRNG driver.
- Set TRNGMode to 0.
- You can collect the data when the entropy source is ready to produce outputs for real-world use (usually after start-up tests).

### Procedure

1. Run 1000 power cycles of the device, and in each cycle collect 1000 bits.
2. Create a 1000 by 1000 matrix according to NIST guidance, in which each column is based on the collected bits:
  - Column 1 is based on the collected bits from run number 1.
  - Column 2 is based on the collected bits from run number 2.
  - ...
  - Column 1000 is based on the collected bits from run number 1000.
3. Analyze the matrix, using the NIST tools, to confirm or refute the TRNG configuration that was calculated in the second iteration.

————— **Note** —————

The NIST tools are published on the NIST site.

### Next Steps

After verification, update these configuration values in the relevant files of the TRNG driver that you are using.

# Appendix A

## CC\_TST\_TRNG output

This appendix describes the format of CC\_TST\_TRNG output.

It contains the following section:

- [\*A.1 CC\\_TST\\_TRNG output format on page Appx-A-24.\*](#)

## A.1 CC\_TST\_TRNG output format

When CC\_TST\_TRNG is run, in addition to collecting samples, it stores some metadata in its output buffer. This buffer is described in terms of 32-bit little-endian words.

The following table lists the value of each word:

**Table A-1 CC\_TST\_TRNG output format**

Buffer offset (32-bit words)	Value
0	Signature value: 0xAABBCCDD.
1	<ul style="list-style-type: none"> <li>Bits [23:0] - <code>buffSize</code>.</li> <li>Bits [25:24] - <code>roscLength</code>.</li> <li>Bits [31:30] - <code>TRNGMode</code>.</li> </ul>
2	<code>sampleCount</code>
3	Signature value: 0xAABBCCDD.
4..N-1	Collected samples. Each 32-bit word contains the first collected sample in bit 0, and the last collected sample in bit 31.
N	Signature value: 0xDDCCBBAA.
N+1	Error flags <ul style="list-style-type: none"> <li>Bit [0] - Samples were lost during collection.</li> <li>Bit [1] - Autocorrelation error occurred.</li> <li>Bit [2] - CRNGT error that is detected and recovered.</li> <li>Bit [3] - Input that is stuck at same level for 32 bits.</li> </ul>
N+2	Signature value: 0xDDCCBBAA.

### Note

The value of "N" is derived from the buffer size, and is calculated to fit the entire data in the supplied buffer (as per the `buffSize` argument).

When parsing the output, it is important to test that the signature value is correct.



# Appendix B

## Revisions

This appendix describes the technical changes between released issues of this book.

It contains the following section:

- [B.1 Revision history](#) on page [Appx-B-26](#).

## B.1 Revision history

**Table B-1 Issue 00 (v1.0)**

Change	Location
First release.	-

**Table B-2 Differences between issue 00 (v1.0) and issue 01 (v1.1)**

Change	Location
Rebranded template to Arm logo and colors.	Entire document.
Product renamed Arm TrustZone® TRNG.	Entire document.
DX_TST_TRNG renamed CC_TST_TRNG.	Entire document.
Added the following standards: <ul style="list-style-type: none"> <li>BSI AIS-31: <i>Functionality Classes and Evaluation Methodology for True Random Number Generators</i></li> <li>NIST SP 800-90A <i>Recommendation for Random Number Generation Using Deterministic Random Bit Generators – App C</i>.</li> </ul>	<a href="#">Referenced Documents</a>
Added STRNG driver operating mode.	<a href="#">ArmTrustZone TRNG Overview</a>
Added STRNG driver to TRNGMode values in CC_TST_TRNG API code block.	<a href="#">First Characterization Test Program. on page 2-18</a>
Added STRNG driver.	<a href="#">Second Characterization Test Program on page 2-20</a>
Added STRNG driver to TRNGMode values in the <i>Reading Arm TrustZone TRNG Configuration Parameters</i> code block.	<a href="#">Arm TrustZone TRNG Configuration Parameters</a>

**Table B-3 Differences between issue 01 (v1.1) and issue 02 (v1.2)**

Change	Location
Renamed TRNG driver modes.	Entire document
Minor rephrasing.	<a href="#">Arm TrustZone TRNG Overview</a>

**Table B-4 Differences between issue 02 (v1.2) and issue 03 (v1.3)**

Change	Location
Minor correction.	<a href="#">Referenced Documents</a>
Rewritten.	<a href="#">Introduction</a>
Added chapter.	<a href="#">Chapter 1 Overview of Arm® TRNG on page 1-9</a>
Renamed from <i>ARM TrustZone TRNG Overview</i> (added under <i>Overview</i> ) and partially rewritten.	<a href="#">Arm TrustZone TRNG</a>
Renamed from <i>Characterization Overview</i> , and partially rewritten.	<a href="#">RNG Characterization on page 1-10</a>
Merged <i>Characterization High-Level Procedure</i> into it and rewritten.	<a href="#">Chapter 2 TRNG characterization procedure on page 2-12</a>
Renamed from <i>Setting the Output Files</i> and partially rewritten.	<a href="#">2.3.1 Output-file names on page 2-15</a>

**Table B-4 Differences between issue 02 (v1.2) and issue 03 (v1.3) (continued)**

Change	Location
Added the section.	<a href="#">2.2 Characterization test program on page 2-14</a>
Rewritten.	<a href="#">2.3 Characterization test conditions on page 2-15</a>
Added the section, including the <i>Finding Minimum Sample Counter Value</i> code block moved from <i>First Characterization Iteration</i> .	<a href="#">2.4 Base iteration on page 2-17</a>
Removed the section, and moved all its subsections under <i>Characterization Procedure</i> .	<i>Characterization Detailed Procedure</i>
Renamed from <i>First Characterization Test Program</i> , replaced first paragraph and removed step 1; partially rewritten.	<a href="#">2.5 First characterization iteration on page 2-18</a>
Renamed from <i>Second Characterization Test Program</i> , and rewritten.	<a href="#">2.6 Second characterization iteration on page 2-20</a>
Removed the section.	<i>Arm TrustZone TRNG Configuration Parameters</i>

**Table B-5 Differences between issue 03 (v1.3) and issue 04 (v1.4)**

Change	Location
Renamed from <i>Introduction</i> and restructured.	<a href="#">Preface on page 0</a>

**Table B-6 Differences between issue 04 (v1.4) and issue 05**

Change	Location
Template changes for the current releases.	Entire Document

**Table B-7 Differences between issue 05 and issue 06**

Change	Location
Document reclassified as Non-Confidential.	Entire document
Minor rephrasing in multiple sections. No technical changes.	Entire document
Split content into a new <a href="#">2.1 Characterization procedure overview on page 2-13</a> section. No technical changes.	<a href="#">Chapter 2 TRNG characterization procedure on page 2-12</a>
Split content into a new <a href="#">A.1 CC_TST_TRNG output format on page Appx-A-24</a> section. No technical changes.	<a href="#">Appendix A CC_TST_TRNG output on page Appx-A-23</a>

**Table B-8 Differences between issue 06 and issue 07**

Change	Location
<i>Compliance</i> section added.	<a href="#">1.2 Compliance on page 1-11</a>
Additional information provided in <i>Overview</i> .	<a href="#">Chapter 1 Overview of Arm® TRNG on page 1-9</a>
<ul style="list-style-type: none"> <li>Note added before the table.</li> <li>Restart tests iteration added as an additional step.</li> </ul>	<a href="#">2.1 Characterization procedure overview on page 2-13</a>
Changes made to the procedure.	<a href="#">2.6 Second characterization iteration on page 2-20</a>
<i>Restart tests iteration</i> section added.	<a href="#">2.7 Restart tests iteration on page 2-22</a>

**Table B-9 Differences between issue 07 and issue 08**

Change	Location
Updated execution of "Second iteration: Run second set of characterization tests."	<a href="#">2.1 Characterization procedure overview on page 2-13</a>
Changes made to the expected outcome of the procedure.	<a href="#">2.5 First characterization iteration on page 2-18</a>

**Table B-10 Differences between issue 08 and issue 09**

Change	Location
Removed TrustZone from product name	Entire document
Updated.	<a href="#">2.2 Characterization test program on page 2-14</a>

**Table B-11 Differences between issue 09 and issue 10**

Change	Location
Clarified that "Restart tests iteration" only applies to 800-90B mode.	<a href="#">2.1 Characterization procedure overview on page 2-13</a>
<ul style="list-style-type: none"> <li>Added paragraph re. location of the CC_TST_TRNGAPI.</li> <li>Updated API code example.</li> <li>Added Definition of callback_TRNG code example and corresponding note.</li> </ul>	<a href="#">2.2 Characterization test program on page 2-14</a>
Updated collection conditions for each TRNG driver.	<a href="#">2.3 Characterization test conditions on page 2-15</a>
<ul style="list-style-type: none"> <li>Defined the amount of data that must be collected for the base iteration and the first characterization iteration.</li> <li>Added the <a href="#">Sample code for data type, macro, and callback function definition on page 2-17</a> example code.</li> <li>Replaced the finding minimum sample counter value example code with <a href="#">Sample code for finding minimum sample counter value on page 2-17</a>.</li> </ul>	<a href="#">2.4 Base iteration on page 2-17</a>
<ul style="list-style-type: none"> <li>Added sample code and updated first characterization iteration description.</li> <li>Added steps to be taken after the first characterization iteration, and before the second characterization iteration.</li> </ul>	<a href="#">2.5 First characterization iteration on page 2-18</a>
Added sample codes.	<a href="#">2.6 Second characterization iteration on page 2-20</a>
Updated prerequisites: <ul style="list-style-type: none"> <li>Only applies to 800-90B.</li> <li>TRNGMode must be set to 0.</li> </ul>	<a href="#">2.7 Restart tests iteration on page 2-22</a>
Buffer offset 1, updated TRNGMode bits from [31:31] to [31:30].	<a href="#">A.1 CC_TST_TRNG output format on page Appx-A-24</a>