

---

**STM32F2xx, STM32F4xx random number generation validation  
using NIST statistical test suite**

---

**Introduction**

Many standards have created requirements and references for the construction, the validation and the use of Random Number Generators (RNG), in order to pursue that the output they produce is indeed random.

The purpose of this application note is to provide guidelines to verify the randomness of numbers generated by an STM32F2xx and an STM32F4xx embedded random number generator peripheral. This verification is based on the National Institute of Standards and Technology (NIST) Statistical Test Suite (STS) SP 800-22, which was published and updated recently as SP800-22rev1a (April 2010).

This application note is structured as follows:

- A general introduction to STM32F2 and STM32F4 random number generator (see [Section 1](#))
- The NIST SP800-22b test suite (see [Section 2](#))
- The steps to run NIST SP800-22b test and analysis (see [Section 3](#))

This application note is provided with a Firmware package which contains 2 projects: STM32F2xx\_RngGenerationViaUART and STM32F4xx\_RngGenerationViaUART, based on STM32F2 and STM32F4 respectively, in order to generate random number to be tested using the NIST statistical test suite.

[Table 1](#) lists the microcontrollers concerned by this application note.

**Table 1. Applicable products**

Type	Applicable products
Microcontrollers	STM32F2xx STM32F4xx

# Contents

- 1        STM32 F2/F4 MCU random number generator ..... 4**
  - 1.1    Introduction ..... 4
  - 1.2    STM32 MCUs implementation description ..... 4
    - 1.2.1    True random number generator ..... 4
  
- 2        NIST SP800-22b test suite ..... 6**
  - 2.1    Introduction ..... 6
  - 2.2    NIST SP800-22b test suite description ..... 6
  
- 3        NIST SP800-22b test suite running and analyzing ..... 8**
  - 3.1    Firmware description ..... 8
    - 3.1.1    On the STM32F2/F4 side ..... 8
    - 3.1.2    On the NIST SP800-22b test suite side ..... 9
  - 3.2    NIST SP800-22b test suite steps ..... 9
    - First step: random number generator..... 9
    - Second step: NIST statistical test ..... 10
    - Third step: tests report ..... 14
  
- 4        Conclusion ..... 15**
  
- Appendix A    Additional information..... 16**
  
- Revision history ..... 23**

## List of figures

Figure 1.	Block diagram . . . . .	5
Figure 2.	Block diagram of deviation testing of a binary sequence from randomness based on NIST test suite . . . . .	9
Figure 3.	Main sts-2.1.1 screen . . . . .	10
Figure 4.	File input screen . . . . .	10
Figure 5.	Statistical test screen . . . . .	11
Figure 6.	Parameter adjustment screen. . . . .	11
Figure 7.	Bitstreams input . . . . .	12
Figure 8.	Input file format . . . . .	12
Figure 9.	Statistical testing in progress . . . . .	13
Figure 10.	Statistical testing complete . . . . .	13

# 1 STM32 F2/F4 MCU random number generator

## 1.1 Introduction

Random number generators (RNG) used for cryptographic applications typically produce sequences made of random 0 and 1 bits.

There are two basic classes of random number generators:

1. Deterministic random number generator or pseudo random number generator (PRNG):  
A deterministic RNG consisting of an algorithm that produces a sequence of bits from an initial value called a seed. To ensure forward unpredictability, care must be taken in obtaining seeds. The values produced by a PRNG are completely predictable if the seed and generation algorithm are known. Since in many cases the generation algorithm is publicly available, the seed must be kept secret and generated from a TRNG.
2. Non deterministic random number generator or True random number generator (TRNG):  
A nondeterministic RNG producing randomness which depends on some unpredictable physical source (i.e. the entropy source) outside of any human control.

The STM32 F2xx and STM32F4xx microcontrollers embed a True random number generator as described in [Section 1.2.1](#).

## 1.2 STM32 MCUs implementation description

### 1.2.1 True random number generator

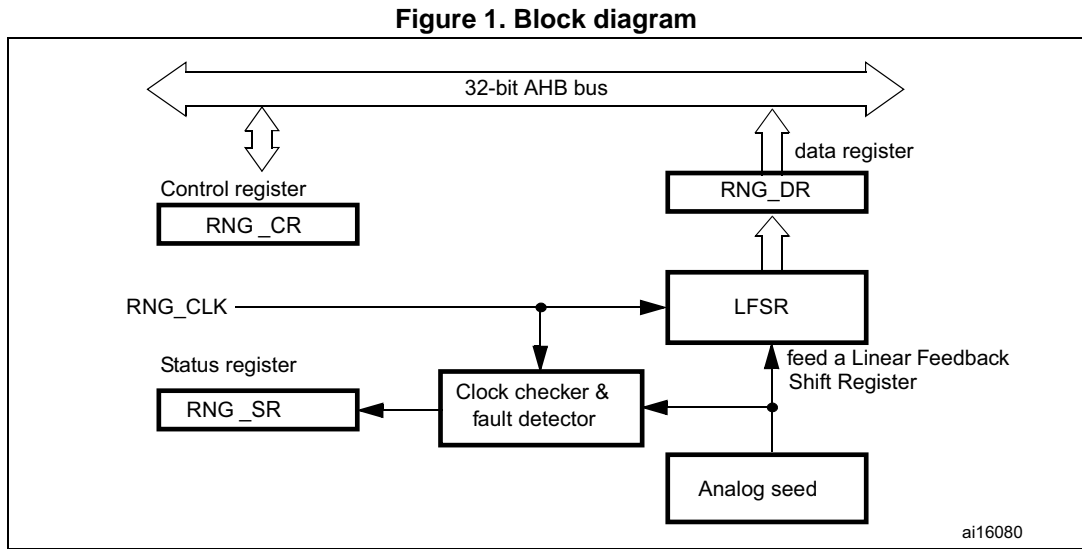
The True random number generator peripheral implemented on STM32 F2 series is the same as the one implemented on F4 series, and it is based on an analog circuit. This circuit generates a continuous analog noise that feed a Linear Feedback Shift Register (LFSR) in order to produce a 32-bit random number.

The analog circuit is made of several ring oscillators whose outputs are XORed.

The LFSR is clocked by a dedicated clock (PLL48CLK) at a constant frequency, so that the quality of the random number is independent of the HCLK frequency.

The contents of LFSR is transferred into the data register (RNG\_DR) when a significant number of seeds have been introduced into LFSR.

Figure 1 shows the TRNG block diagram on STM32F2/F4 series.



For more details about the True random number generator embedded in STM32F2/F4 series, refer to STM32F2 reference manual (RM0033) or STM32F4 reference manual (RM00329).

## 2 NIST SP800-22b test suite

### 2.1 Introduction

The NIST SP800-22b statistical test suite has been carried out using a statistical test suite (sts) developed by the National Institute of Standards and Technology (NIST) to probe the quality of random generators for cryptographic applications. A comprehensive description of the suite was presented in the paper entitled: “*A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications*”.

### 2.2 NIST SP800-22b test suite description

The NIST SP800-22b statistical test suite “sts-2.1.1” is a software package which was developed by the *National Institute of Standards and Technology*; it can be downloaded from the NIST web site at the following address:

[http://csrc.nist.gov/groups/ST/toolkit/rng/documentation\\_software.html](http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html)

The source code has been written in ANSI C. The NIST statistical test suite consists of 15 tests that were developed to test the randomness of a binary sequence. These tests focus on a variety of types of non-randomness that could exist in a sequence. From this point of view, the test suite can be classified as follows:

#### Frequency tests

- Frequency (Monobit) test:  
To measure the distribution of 0's and 1's in a sequence and to check if the result is similar to the one expected for a truly random sequence.
- Frequency test within a block:  
To check whether the frequency of 1's in an M-bit block is approximately  $M/2$ , as would be expected from the theory of randomness.
- Run tests:  
To assess if the expected total number of runs of 1's and 0's of various lengths is as expected for a random sequence.
- Test of the longest run of 1's in a block:  
To examine the long runs of 1's in a sequence.

#### Test of linearity

- Binary matrix rank test:  
To assess the distribution of the rank for 32x32 binary matrices.
- Linear complexity test:  
To determine the linear complexity of a finite sequence.

#### Test of correlation (by means of Fourier transform)

- Discrete Fourier transform (spectral) test:  
To assess the spectral frequency of a bit string via the spectral test based on the discrete Fourier transform. It is sensitive to the periodicity in the sequence.

**Test of finding some special strings**

- Non-overlapping template matching test:  
To assess the frequency of m-bit non-periodic patterns.
- Overlapping template matching test:  
To assess the frequency of m-bit periodic templates

**Entropy tests**

- Maurer's "Universal Statistical" test:  
To assess the compressibility of a binary sequence of L-bit blocks.
- Serial test:  
To assess the distribution of all  $2^m$  m-bit blocks.

*Note:* For  $m = 1$ , the serial test is equivalent to the Frequency test of [Section 2.2](#).

- Approximate entropy test:  
To assess the entropy of a bit string, comparing the frequency of all m-bit patterns against all (m+1)-bit patterns.

**Random walk tests**

- Cumulative sums (Cusums) test:  
To assess that the sum of partial sequences is not too large or too small; it is indicative of too many 0's or 1's.
- Random excursion test:  
To assess the distribution of states within a cycle of random walk.
- Random excursion variant test:  
To detect deviations from the expected number of visits to different states of the random walk.

Each of the above tests is based on a calculated test statistic value, which is a function of the testing sequence. The test statistic value is used to calculate a Pvalue where:

**Pvalue** is the probability that the perfect random number generator would have produced a sequence less random than the sequence that was tested.

For more details about the NIST statistical test suite, refer to the following NIST document available on the NIST web site:

*"A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications" Special Publication 800-22 Revision 1a.*

## 3 NIST SP800-22b test suite running and analyzing

### 3.1 Firmware description

To run the NIST statistical test suite as described in the previous section, two firmwares are needed, one on the STM32F2/F4 side and one on the NIST SP800-22b test suite side.

#### 3.1.1 On the STM32F2/F4 side

Within this application note, the firmware program is:

- STM32F2xx\_RngGenerationViaUART for STM32F2 series
- STM32F4xx\_RngGenerationViaUART for STM32F4 series

Each of them is provided with five different tool chains to allow the generation of random numbers using the STM32 Random number generator (RNG) peripheral and to send them to a workstation via the UART interface with the following settings:

- Baud rate = 38400 baud <sup>(a)</sup>
- Word length = 8 bits
- One stop bit
- No parity
- Hardware flow control disabled (RTS and CTS signals)

Each firmware program is used to generate 10 blocks of 64,000 bytes of random number, so the output file will contain 5,120,000 random bits to be tested with the NIST statistical test.

As recommended by the NIST statistical test suite, the output file format can:

1. be a sequence of ASCII 0's and 1's if the *FILE\_ASCII\_FORMAT* Private define is uncommented in the main.c file
2. have each byte in the data file containing 8 bits of data if the *FILE\_BINARY\_FORMAT* Private define is uncommented in the main.c file.

After startup, the program is waiting for the user to press a key button in order to start generating a random number and sending it to the USART.

Two status LEDs, LED 1 and LED 2, are used in this program as follows:

- After startup, LED 1 and LED 2 turn Off.
- LED 2 turns On and LED 1 turns Off when a key button is pressed to indicate that RNG is on-going.
- LED 2 turns Off and LED 1 turns On when RNG generation is complete.

For more details about the program description and settings, refer to the readme file inside the firmware package provided with this application note.

---

a. it is recommended to use 34800 baud in order to generate a baudrate with a high precision (the Error calculation for programmed baud rates is equal to 0% error). For more details about the Error calculation for programmed baud rates, refer to the Universal synchronous asynchronous receiver transmitter (USART) section in the STM32F2 reference manual (RM0033) or the STM32F4 reference manual (RM00329)."



*Note:* The USART configuration can be changed by the user via the `SendToWorkstation()` function in the `main.c` file.

The output values can be changed by the user by modifying the `Private define` in the `main.c` file:

```
#define NUMBER_OF_RANDOM_BITS_TO_GENERATE 512000
#define BLOCK_NUMBER 10
```

### 3.1.2 On the NIST SP800-22b test suite side

Downloaded on a workstation, the NIST statistical test suite package `sts-2.1.1` verifies the randomness of the output file of the random number generator from STM32 F2/F4 products.

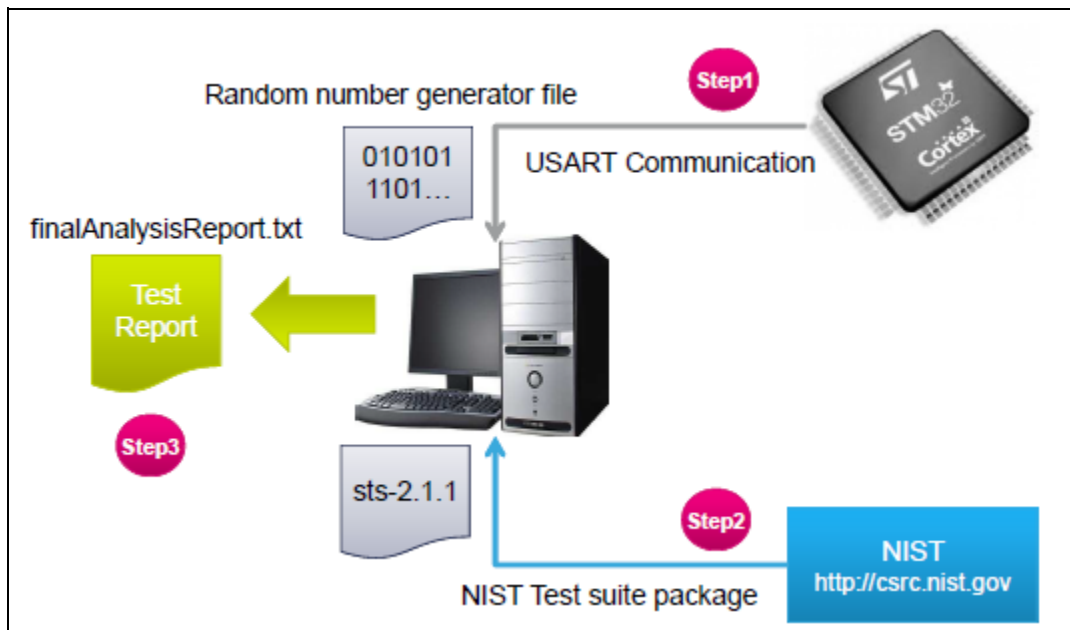
The generator file to be analyzed should be stored under the `data` folder (`sts-2.1.1\data`).

For more details about how the NIST statistical tests work, refer to section How to Get Started in “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications” Special Publication 800-22 Revision 1a.

## 3.2 NIST SP800-22b test suite steps

*Figure 2* describes the different steps needed to verify the randomness of an output number generated by STM32 F2/F4 using the NIST statistical test suite package `sts-2.1.1`.

**Figure 2. Block diagram of deviation testing of a binary sequence from randomness based on NIST test suite**



### 3.2.1 First step: random number generator

Connect STMicroelectronics evaluation board (STM322xG-EVAL” for STM32F2 series or STM324xG-EVAL” for STM32F4 series) with DB9 connector CN16 (USART3) to the workstation serial port via a null-modem female/female RS232 cable. Then, run STM32 RNG generation via the UART firmware using your preferred toolchain in order to generate a random number as described in the previous section. You can store data on the

workstation using a terminal emulation application such as HyperTerminal (a program that comes with Windows operating system, from Windows 98).

*Note: This example has been developed and tested with STM322xG-EVAL RevB for STM32 F2 series and STM324xG-EVAL RevC for STM32F4 series. It can be easily tailored for any other development board.*

### 3.2.2 Second step: NIST statistical test

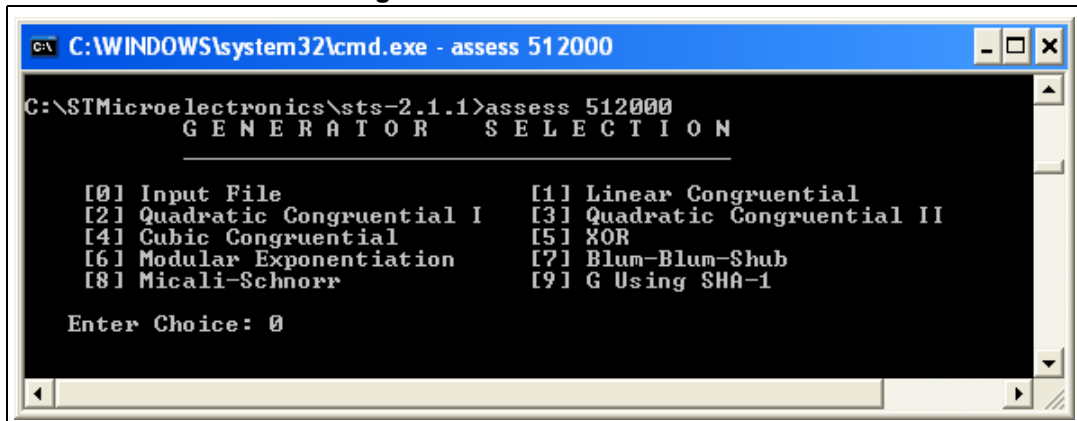
Compile sts-2.1.1 package as described in the NIST statistical test suite documentation in order to create an executable program using visual C++ compiler.

After running the NIST statistical test suite program, a series of menu prompts will be displayed in order to select the data to be analyzed and the statistical tests to be applied.

In this application note, the NIST statistical test suite is compiled under the name *assess.exe* and saved under *NIST\_Test\_Suite\_OutputExample* folder. As described in the previous section, the random number is defined as *512,000 bits per block*.

The first screen appears as in [Figure 3](#).

Figure 3. Main sts-2.1.1 screen



When value 0 is entered, the program requires to enter the file name and path of the random number to be tested.

The second screen appears as in [Figure 4](#).

Figure 4. File input screen

```

C:\WINDOWS\system32\cmd.exe - assess 512000
C:\STMicroelectronics\sts-2.1.1>assess 512000
  G E N E R A T O R   S E L E C T I O N
-----
[0] Input File                [1] Linear Congruential
[2] Quadratic Congruential I  [3] Quadratic Congruential II
[4] Cubic Congruential        [5] XOR
[6] Modular Exponentiation    [7] Blum-Blum-Shub
[8] Micali-Schnorr            [9] G Using SHA-1

Enter Choice: 0

User Prescribed Input File: data\ascii.bin

```

This application note provides you with an example of two files per series, generated with STM32F2 and STM32F4 random number generator with different file formats as recommended by NIST:

1. *ascii.bin*: sequence of ASCII 0's and 1's.
2. *binary.bin*: each byte in the data file contains 8 bits of data.

Then, the NIST statistical test suite displays 15 tests that can be run via the screen in [Figure 5](#).

Figure 5. Statistical test screen

```

C:\WINDOWS\system32\cmd.exe - assess 512000
C:\STMicroelectronics\sts-2.1.1>assess 512000
  S T A T I S T I C A L   T E S T S
-----
[01] Frequency                [02] Block Frequency
[03] Cumulative Sums          [04] Runs
[05] Longest Run of Ones     [06] Rank
[07] Discrete Fourier Transform [08] Nonperiodic Template Mat
[09] Overlapping Template Matchings [10] Universal Statistical
[11] Approximate Entropy      [12] Random Excursions
[13] Random Excursions Variant [14] Serial
[15] Linear Complexity

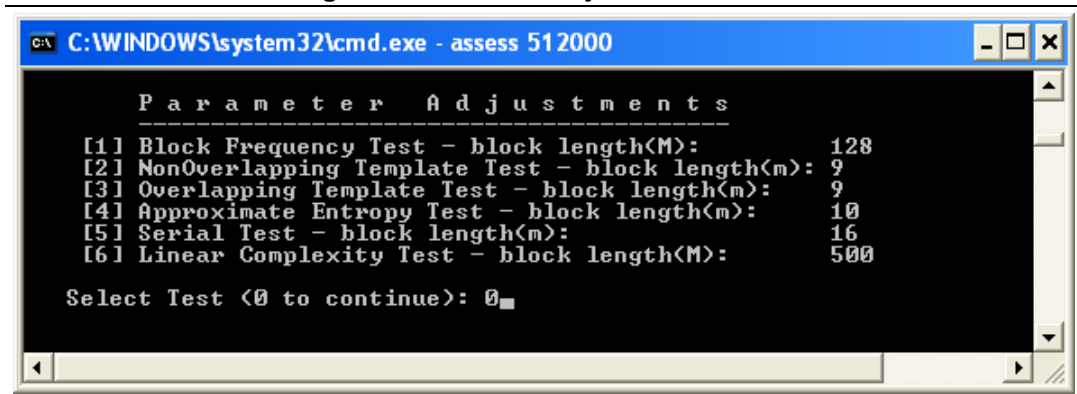
INSTRUCTIONS
Enter 0 if you DO NOT want to apply all of the
statistical tests to each sequence and 1 if you DO.

Enter Choice: 1_

```

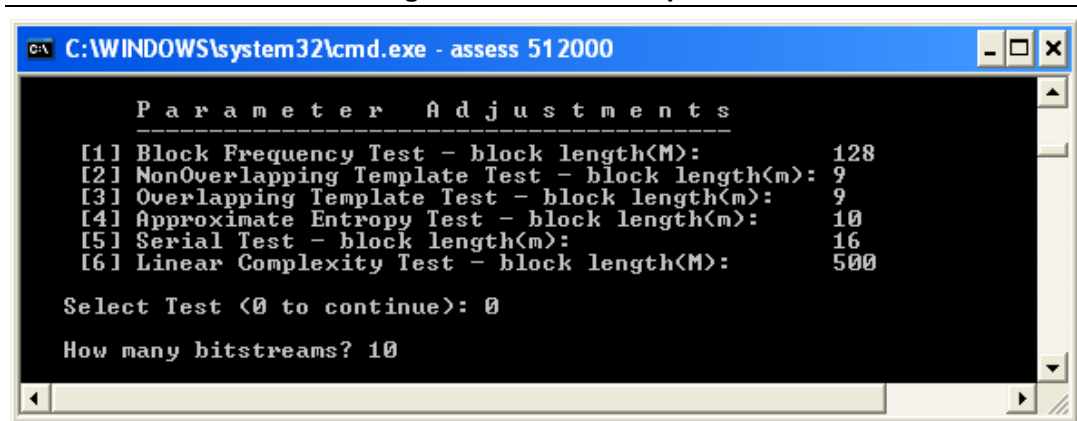
In this case, '1' has been selected to apply all of the statistical tests. [Figure 6](#) is displayed to change the parameter adjustments.

Figure 6. Parameter adjustment screen



In this example, the default settings are kept and value '0' is selected to go to the next step.

Figure 7. Bitstreams input



The NIST statistical test suite requires to put the number of bitstreams; here, '10' is entered.

You have selected 10 blocks of 512,000 bits which equal to 5,120,000 bits.

You must then specify whether the file consists of bits stored in ASCII format or hexadecimal strings stored in a binary format using the following screen.

Figure 8. Input file format

```

C:\WINDOWS\system32\cmd.exe - assess 512000

Parameter Adjustments
-----
[1] Block Frequency Test - block length(M):      128
[2] NonOverlapping Template Test - block length(m): 9
[3] Overlapping Template Test - block length(m):  9
[4] Approximate Entropy Test - block length(m):  10
[5] Serial Test - block length(m):                16
[6] Linear Complexity Test - block length(M):     500

Select Test <0 to continue>: 0

How many bitstreams? 10

Input File Format:
[0] ASCII - A sequence of ASCII 0's and 1's
[1] Binary - Each byte in data file contains 8 bits of data

Select input mode: 0

```

Value '0' is selected because the file is in ASCII format.

After entering all necessary inputs, the NIST statistical test suite starts analyzing the input file.

Figure 9. Statistical testing in progress

```

C:\WINDOWS\system32\cmd.exe - assess 512000

Parameter Adjustments
-----
[1] Block Frequency Test - block length(M):      128
[2] NonOverlapping Template Test - block length(m): 9
[3] Overlapping Template Test - block length(m):  9
[5] Serial Test - block length(m):                16
[6] Linear Complexity Test - block length(M):     500

Select Test <0 to continue>: 0

How many bitstreams? 10

Input File Format:
[0] ASCII - A sequence of ASCII 0's and 1's
[1] Binary - Each byte in data file contains 8 bits of data

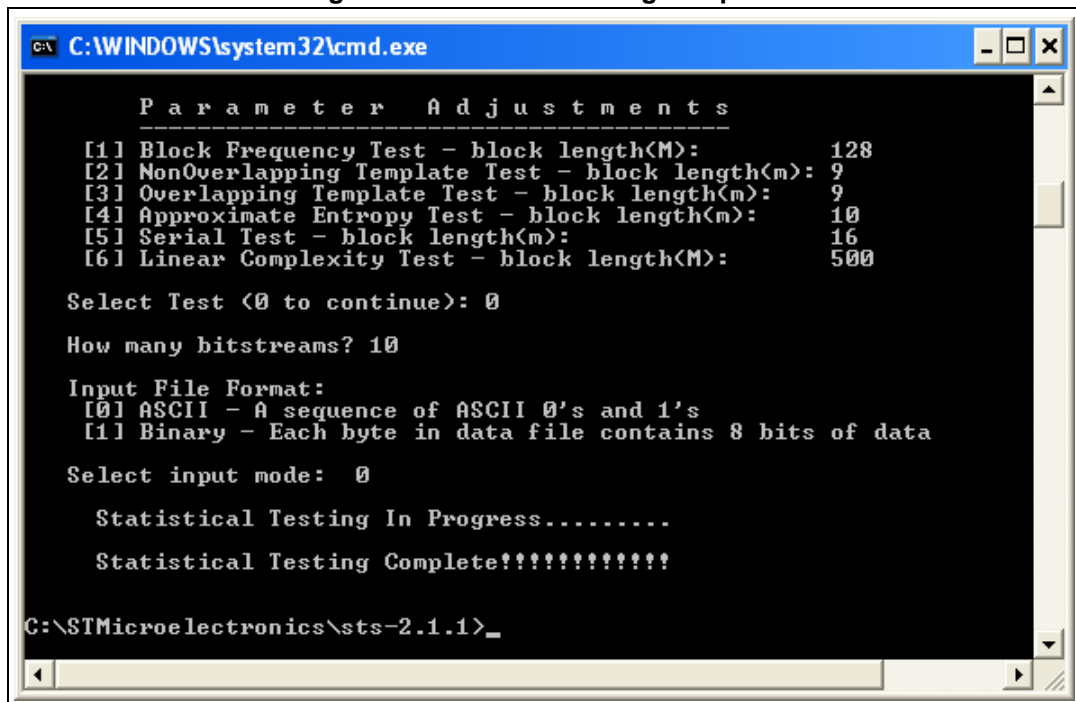
Select input mode: 0

Statistical Testing In Progress.....

```

When the testing process is complete, as shown in [Figure 10](#), the statistical test results can be found in sts-2.1.1\experiments\AlgorithmTesting.

Figure 10. Statistical testing complete



### 3.2.3 Third step: tests report

The NIST statistical tests provide an analytical routine to facilitate the interpretation of the results. A file named finalAnalysisReport is generated when the statistical testing is complete and saved under sts2.1.1\experiments\AlgorithmTesting. The report contains a summary of experimental results of 15 tests, as described in [Appendix A](#).

The NST statistical tests also provide a detailed report for each test, which is saved under sts-2.1.1\experiments\AlgorithmTesting\<Test suite name>.

*Note:* You can find two examples per series of complete NIST statistical test suite output reports under “NIST\_Test\_Suite\_OutputExample”.

1. Example of an *Ascii\_File\_Format*. This example has 2 folders:
  - **Input\_File** which contains the random number generator saved with the ascii format.
  - **Final\_Analysis\_Report** which contains the complete NIST statistical test suite output report based on this input file, the summary of experimental results and the report of each test.
2. Example of a *Binary\_File\_Format*. This example has 2 folders:
  - **Input\_File** which contains the random number generator saved with the binary format.
  - **Final\_Analysis\_Report** which contains the complete NIST statistical test suite output report based on this input file, the summary of experimental results and the report of each test.

## 4 Conclusion

This application note describes the main guidelines and steps to verify the randomness of numbers generated by the STM32F2/F4 Random Number Generator peripheral using NIST statistical test suite SP800-22rev1a, April 2010.

15 tests of NIST statistical test suite are passed in accordance with the minimum pass rate for each statistical test under the following conditions:

- **Environment**

STM32F2 series:

- STMicroelectronics evaluation board STM322xG-EVAL RevB
- STM32F207IG device

STM32F4 series:

- STMicroelectronics evaluation board STM324xG-EVAL RevB
- STM32F407IG device

Ambient temperature

- **Firmware**

NIST statistical test suite sts-2.1.1 with default parameter

STM32F2 series:

- STM32F2xx Standard Peripherals Library version V1.1.0/13-April-2012

STM32F4 series:

- STM32F4xx Standard Peripherals Library version V1.0.2/05-March-2012

- **Input file for each series**

- Binary format
- Ascii format
- Size: 5,120,000 bits

## Appendix A Additional information

The results are represented as a table with p rows and q columns.

- The number of rows, p, corresponds to the number of statistical tests applied.
- The number of columns, q = 13, is distributed as follows:
  - columns 1-10 correspond to the frequency of P-values<sup>10</sup>,
  - column 11 is the P-value that arises via the application of a chi-square test<sup>11</sup>,
  - column 12 is the proportion of binary sequences that passed,
  - column 13 is the corresponding statistical test.

An example is shown in [Table 2](#). For more details, refer to finalAnalysisReport file under sts-2.1.1\experiments\AlgorithmTesting).

**Table 2. Example**

```

-----
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING
SEQUENCES
-----

generator is <data/ascii.bin>
-----

C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 P-VALUE PROPORTION STATISTICAL TEST
-----
0 1 2 1 2 1 1 1 0 1 0.911413 10/10 Frequency
1 1 0 1 3 0 2 1 1 0 0.534146 10/10 BlockFrequency
0 1 3 3 0 1 0 2 0 0 0.122325 10/10 CumulativeSums
1 1 3 1 0 1 1 1 0 1 0.739918 10/10 CumulativeSums
2 0 2 2 1 1 0 1 0 1 0.739918 10/10 Runs
1 0 1 1 0 3 1 1 0 2 0.534146 9/10 LongestRun
1 2 1 0 2 1 1 0 0 2 0.739918 10/10 Rank
3 0 1 2 1 1 0 1 0 1 0.534146 9/10 FFT
1 1 1 0 0 2 1 2 0 2 0.739918 10/10 NonOverlappingTemplate
1 1 0 0 1 1 1 3 0 2 0.534146 10/10 NonOverlappingTemplate
0 2 1 0 4 0 2 0 0 1 0.066882 10/10 NonOverlappingTemplate
0 0 0 1 1 3 0 2 1 2 0.350485 10/10 NonOverlappingTemplate
0 1 2 2 1 1 1 2 0 0 0.739918 10/10 NonOverlappingTemplate
2 2 1 0 2 0 1 1 1 0 0.739918 10/10 NonOverlappingTemplate
1 0 2 2 1 1 1 0 1 1 0.911413 10/10 NonOverlappingTemplate
0 0 1 1 0 0 2 3 1 2 0.350485 10/10 NonOverlappingTemplate
    
```



1	1	1	0	4	0	0	0	2	1	0.122325	10/10	NonOverlappingTemplate
1	0	1	0	2	3	0	0	2	1	0.350485	10/10	NonOverlappingTemplate
0	1	0	2	2	2	2	0	1	0	0.534146	10/10	NonOverlappingTemplate
1	0	1	1	2	0	2	1	0	2	0.739918	10/10	NonOverlappingTemplate
0	1	2	0	0	0	3	2	1	1	0.350485	10/10	NonOverlappingTemplate
2	1	1	1	2	2	0	0	1	0	0.739918	10/10	NonOverlappingTemplate
1	1	2	3	1	0	1	0	1	0	0.534146	10/10	NonOverlappingTemplate
1	3	1	3	1	1	0	0	0	0	0.213309	10/10	NonOverlappingTemplate
1	1	2	0	2	2	0	0	1	1	0.739918	10/10	NonOverlappingTemplate
2	1	1	1	1	0	1	1	1	1	0.991468	10/10	NonOverlappingTemplate
2	0	0	0	1	4	0	1	0	2	0.066882	10/10	NonOverlappingTemplate
2	0	0	1	2	2	1	0	0	2	0.534146	10/10	NonOverlappingTemplate
0	2	1	1	0	3	1	1	0	1	0.534146	10/10	NonOverlappingTemplate
0	2	1	1	2	1	1	0	1	1	0.911413	10/10	NonOverlappingTemplate
0	0	1	0	4	0	2	1	0	2	0.066882	10/10	NonOverlappingTemplate
1	2	1	2	0	2	2	0	0	0	0.534146	10/10	NonOverlappingTemplate
3	0	1	1	0	1	3	1	0	0	0.213309	10/10	NonOverlappingTemplate
2	0	1	0	1	0	1	4	0	1	0.122325	10/10	NonOverlappingTemplate
0	1	1	0	1	1	0	1	4	1	0.213309	10/10	NonOverlappingTemplate
3	2	0	0	1	0	3	1	0	0	0.122325	10/10	NonOverlappingTemplate
0	3	0	2	1	1	1	1	0	1	0.534146	10/10	NonOverlappingTemplate
0	1	1	4	0	0	1	0	1	2	0.122325	10/10	NonOverlappingTemplate
0	0	2	0	1	2	1	4	0	0	0.066882	10/10	NonOverlappingTemplate
2	1	2	0	0	1	2	1	1	0	0.739918	10/10	NonOverlappingTemplate
1	1	2	2	0	0	0	1	2	1	0.739918	10/10	NonOverlappingTemplate
1	0	1	3	1	0	1	2	1	0	0.534146	10/10	NonOverlappingTemplate
1	3	2	1	1	0	0	1	0	1	0.534146	10/10	NonOverlappingTemplate
2	0	0	0	0	0	2	1	1	4	0.066882	10/10	NonOverlappingTemplate
0	1	1	0	1	0	1	2	1	3	0.534146	10/10	NonOverlappingTemplate
1	2	0	0	2	2	0	1	1	1	0.739918	10/10	NonOverlappingTemplate
1	3	0	2	0	0	0	2	2	0	0.213309	9/10	NonOverlappingTemplate
3	1	0	0	0	2	1	0	3	0	0.122325	10/10	NonOverlappingTemplate
1	0	2	0	3	0	1	1	1	1	0.534146	10/10	NonOverlappingTemplate
0	1	1	1	0	2	2	2	0	1	0.739918	10/10	NonOverlappingTemplate
2	1	0	0	3	1	1	0	2	0	0.350485	10/10	NonOverlappingTemplate

0	0	1	2	1	4	1	1	0	0	0.122325	10/10	NonOverlappingTemplate
1	1	0	1	1	3	2	0	1	0	0.534146	10/10	NonOverlappingTemplate
2	1	0	2	1	0	1	0	1	2	0.739918	10/10	NonOverlappingTemplate
0	0	2	3	1	1	0	1	1	1	0.534146	10/10	NonOverlappingTemplate
1	1	1	0	1	3	1	0	1	1	0.739918	9/10	NonOverlappingTemplate
0	1	4	0	0	1	2	1	1	0	0.122325	10/10	NonOverlappingTemplate
1	3	0	0	0	0	1	1	2	2	0.350485	10/10	NonOverlappingTemplate
2	1	1	1	1	0	1	0	2	1	0.911413	10/10	NonOverlappingTemplate
2	1	2	0	1	1	0	1	1	1	0.911413	9/10	NonOverlappingTemplate
1	0	1	0	2	2	1	2	0	1	0.739918	10/10	NonOverlappingTemplate
0	1	3	1	1	1	1	2	0	0	0.534146	10/10	NonOverlappingTemplate
4	0	0	0	2	1	2	0	0	1	0.066882	10/10	NonOverlappingTemplate
0	0	0	0	2	0	1	2	2	3	0.213309	10/10	NonOverlappingTemplate
2	2	0	0	3	1	0	1	0	1	0.350485	10/10	NonOverlappingTemplate
2	0	2	0	1	0	2	2	0	1	0.534146	9/10	NonOverlappingTemplate
1	0	1	0	0	1	3	0	3	1	0.213309	9/10	NonOverlappingTemplate
1	2	1	1	0	1	3	0	0	1	0.534146	10/10	NonOverlappingTemplate
0	1	2	0	1	0	1	2	0	3	0.350485	10/10	NonOverlappingTemplate
2	0	2	0	0	0	2	0	3	1	0.213309	10/10	NonOverlappingTemplate
2	1	2	1	0	1	0	2	0	1	0.739918	10/10	NonOverlappingTemplate
1	0	1	0	4	0	0	1	2	1	0.122325	10/10	NonOverlappingTemplate
0	0	0	2	1	1	3	2	1	0	0.350485	10/10	NonOverlappingTemplate
1	3	1	0	2	0	1	0	0	2	0.350485	9/10	NonOverlappingTemplate
0	0	1	0	0	3	1	2	3	0	0.122325	10/10	NonOverlappingTemplate
0	1	0	2	1	0	1	2	3	0	0.350485	10/10	NonOverlappingTemplate
0	0	0	2	4	2	1	1	0	0	0.066882	10/10	NonOverlappingTemplate
1	0	0	2	0	1	2	1	2	1	0.739918	9/10	NonOverlappingTemplate
0	0	2	0	1	2	0	0	1	4	0.066882	10/10	NonOverlappingTemplate
1	1	0	1	1	1	2	0	2	1	0.911413	10/10	NonOverlappingTemplate
0	0	0	5	2	0	1	2	0	0	0.004301	10/10	NonOverlappingTemplate
2	1	0	0	1	1	0	1	3	1	0.534146	10/10	NonOverlappingTemplate
1	1	1	0	0	2	1	2	0	2	0.739918	10/10	NonOverlappingTemplate
2	0	2	1	0	1	2	0	1	1	0.739918	10/10	NonOverlappingTemplate
0	1	3	0	1	0	1	2	1	1	0.534146	10/10	NonOverlappingTemplate
1	1	1	1	1	1	1	2	1	0	0.991468	10/10	NonOverlappingTemplate



0	0	3	2	2	2	0	1	0	0	0.213309	10/10	NonOverlappingTemplate
2	1	2	1	1	1	0	0	1	1	0.911413	10/10	NonOverlappingTemplate
0	1	1	1	0	1	2	1	2	1	0.911413	10/10	NonOverlappingTemplate
1	0	0	2	3	2	0	0	1	1	0.350485	10/10	NonOverlappingTemplate
2	2	1	0	0	1	1	3	0	0	0.350485	10/10	NonOverlappingTemplate
2	0	2	2	0	0	0	1	0	3	0.213309	10/10	NonOverlappingTemplate
1	1	1	2	0	0	2	2	0	1	0.739918	10/10	NonOverlappingTemplate
3	2	1	0	1	0	0	0	1	2	0.350485	9/10	NonOverlappingTemplate
2	0	1	1	2	1	1	0	1	1	0.911413	10/10	NonOverlappingTemplate
0	2	0	0	1	0	1	3	2	1	0.350485	10/10	NonOverlappingTemplate
1	0	3	1	0	0	1	2	1	1	0.534146	9/10	NonOverlappingTemplate
1	2	0	2	0	4	0	0	0	1	0.066882	10/10	NonOverlappingTemplate
1	0	1	0	2	0	1	1	4	0	0.122325	9/10	NonOverlappingTemplate
1	1	1	1	0	0	3	2	1	0	0.534146	10/10	NonOverlappingTemplate
2	0	1	0	0	1	1	1	1	3	0.534146	9/10	NonOverlappingTemplate
3	0	1	1	1	1	1	1	1	0	0.739918	10/10	NonOverlappingTemplate
1	0	0	1	1	1	1	3	0	2	0.534146	10/10	NonOverlappingTemplate
3	3	1	1	0	0	0	1	1	0	0.213309	10/10	NonOverlappingTemplate
1	0	0	1	1	0	1	1	4	1	0.213309	10/10	NonOverlappingTemplate
0	2	2	3	1	1	0	1	0	0	0.350485	10/10	NonOverlappingTemplate
2	1	0	3	0	0	2	0	1	1	0.350485	10/10	NonOverlappingTemplate
1	0	0	3	0	1	1	2	0	2	0.350485	9/10	NonOverlappingTemplate
1	2	0	1	0	0	3	2	0	1	0.350485	10/10	NonOverlappingTemplate
2	1	1	1	2	0	1	1	1	0	0.911413	10/10	NonOverlappingTemplate
0	0	1	2	1	1	1	3	0	1	0.534146	10/10	NonOverlappingTemplate
0	0	2	1	3	0	3	0	0	1	0.122325	10/10	NonOverlappingTemplate
0	2	2	1	0	0	2	1	1	1	0.739918	10/10	NonOverlappingTemplate
1	2	0	2	2	1	0	0	1	1	0.739918	10/10	NonOverlappingTemplate
1	1	2	0	2	1	3	0	0	0	0.350485	10/10	NonOverlappingTemplate
2	1	0	1	1	1	3	1	0	0	0.534146	9/10	NonOverlappingTemplate
2	4	0	1	1	1	0	0	0	1	0.122325	10/10	NonOverlappingTemplate
0	0	1	0	2	2	2	2	0	1	0.534146	10/10	NonOverlappingTemplate
2	1	2	0	1	1	1	1	0	1	0.911413	10/10	NonOverlappingTemplate
1	1	1	4	1	1	1	0	0	0	0.213309	10/10	NonOverlappingTemplate
1	0	0	2	0	1	2	1	3	0	0.350485	10/10	NonOverlappingTemplate

4	0	2	0	2	0	1	0	0	1	0.066882	10/10	NonOverlappingTemplate
0	1	2	2	0	0	0	2	2	1	0.534146	10/10	NonOverlappingTemplate
1	0	2	0	1	1	0	3	1	1	0.534146	10/10	NonOverlappingTemplate
4	1	1	0	0	0	0	1	1	2	0.122325	10/10	NonOverlappingTemplate
2	1	0	0	1	1	2	1	1	1	0.911413	10/10	NonOverlappingTemplate
0	1	1	1	0	0	1	4	1	1	0.213309	10/10	NonOverlappingTemplate
1	2	2	0	0	5	0	0	0	0	0.004301	10/10	NonOverlappingTemplate
0	2	2	2	0	2	0	1	1	0	0.534146	10/10	NonOverlappingTemplate
3	1	1	0	0	2	0	1	1	1	0.534146	10/10	NonOverlappingTemplate
2	1	0	1	1	1	0	2	1	1	0.911413	10/10	NonOverlappingTemplate
1	1	2	1	0	1	1	0	1	2	0.911413	10/10	NonOverlappingTemplate
0	1	2	3	0	2	0	2	0	0	0.213309	10/10	NonOverlappingTemplate
0	1	0	1	3	1	2	0	0	2	0.350485	10/10	NonOverlappingTemplate
3	0	0	3	1	0	1	0	1	1	0.213309	9/10	NonOverlappingTemplate
0	1	0	2	0	2	1	0	3	1	0.350485	10/10	NonOverlappingTemplate
1	1	3	2	0	1	0	1	0	1	0.534146	10/10	NonOverlappingTemplate
0	1	0	3	0	0	0	1	0	5	0.002043	10/10	NonOverlappingTemplate
2	1	0	1	0	0	3	1	1	1	0.534146	10/10	NonOverlappingTemplate
2	0	0	0	2	1	1	0	3	1	0.350485	10/10	NonOverlappingTemplate
0	2	1	1	2	2	0	0	0	2	0.534146	10/10	NonOverlappingTemplate
0	0	1	0	3	0	1	1	1	3	0.213309	10/10	NonOverlappingTemplate
1	0	2	2	0	0	0	2	3	0	0.213309	10/10	NonOverlappingTemplate
1	2	2	1	1	1	0	1	0	1	0.911413	10/10	NonOverlappingTemplate
1	1	1	1	4	0	0	1	0	1	0.213309	9/10	NonOverlappingTemplate
0	1	1	2	2	1	0	0	2	1	0.739918	10/10	NonOverlappingTemplate
2	1	0	0	2	0	1	2	1	1	0.739918	10/10	NonOverlappingTemplate
1	0	1	1	0	0	2	3	1	1	0.534146	10/10	NonOverlappingTemplate
2	1	0	0	1	2	2	0	0	2	0.534146	10/10	NonOverlappingTemplate
0	0	2	0	1	0	3	3	0	1	0.122325	10/10	NonOverlappingTemplate
1	1	1	2	0	3	1	0	0	1	0.534146	10/10	NonOverlappingTemplate
1	0	1	1	1	2	2	1	0	1	0.911413	10/10	NonOverlappingTemplate
0	1	2	0	1	1	2	0	1	2	0.739918	10/10	NonOverlappingTemplate
1	2	0	0	1	3	1	1	0	1	0.534146	10/10	NonOverlappingTemplate
2	1	1	0	1	0	0	2	0	3	0.350485	10/10	NonOverlappingTemplate
2	1	0	0	1	1	0	1	3	1	0.534146	10/10	NonOverlappingTemplate

2	0	1	0	1	2	1	0	2	1	0.739918	10/10	OverlappingTemplate
1	0	2	1	0	2	2	1	1	0	0.739918	10/10	Universal
1	1	0	0	2	0	2	3	1	0	0.350485	10/10	ApproximateEntropy
0	1	1	1	1	0	0	0	1	0	----	5/5	RandomExcursions
1	1	0	0	2	0	0	0	0	1	----	5/5	RandomExcursions
0	1	1	1	0	0	0	0	1	1	----	5/5	RandomExcursions
0	0	0	0	0	1	1	0	2	1	----	5/5	RandomExcursions
1	0	0	0	3	0	0	0	1	0	----	5/5	RandomExcursions
0	0	0	1	1	0	0	1	1	1	----	5/5	RandomExcursions
1	0	1	1	0	2	0	0	0	0	----	5/5	RandomExcursions
1	0	0	0	1	1	1	0	1	0	----	5/5	RandomExcursions
2	1	0	1	1	0	0	0	0	0	----	5/5	RandomExcursionsVariant
2	1	0	0	1	1	0	0	0	0	----	5/5	RandomExcursionsVariant
1	1	0	2	1	0	0	0	0	0	----	5/5	RandomExcursionsVariant
1	2	0	1	1	0	0	0	0	0	----	5/5	RandomExcursionsVariant
1	1	1	1	0	0	0	1	0	0	----	5/5	RandomExcursionsVariant
1	1	0	1	1	0	0	0	0	1	----	5/5	RandomExcursionsVariant
0	1	0	2	1	0	0	0	0	1	----	5/5	RandomExcursionsVariant
0	0	0	1	0	1	0	3	0	0	----	5/5	RandomExcursionsVariant
0	0	0	0	0	0	2	1	1	1	----	5/5	RandomExcursionsVariant
0	0	1	0	0	0	1	1	1	1	----	5/5	RandomExcursionsVariant
0	0	0	1	0	0	2	0	2	0	----	5/5	RandomExcursionsVariant
0	1	0	0	1	1	1	1	0	0	----	5/5	RandomExcursionsVariant
1	0	0	2	0	1	1	0	0	0	----	5/5	RandomExcursionsVariant
1	0	0	0	2	1	0	0	0	1	----	5/5	RandomExcursionsVariant
0	0	0	1	1	0	1	1	1	0	----	5/5	RandomExcursionsVariant
0	0	0	0	2	0	2	0	0	1	----	5/5	RandomExcursionsVariant
0	0	1	0	1	2	1	0	0	0	----	5/5	RandomExcursionsVariant
0	0	1	0	0	2	2	0	0	0	----	5/5	RandomExcursionsVariant
1	1	0	0	0	2	3	0	2	1	0.350485	10/10	Serial
0	2	1	0	3	1	0	1	1	1	0.534146	10/10	Serial
2	1	1	1	0	1	1	3	0	0	0.534146	10/10	LinearComplexity

-----  
 The minimum pass rate for each statistical test, with the exception of the random excursion (variant) test, is approximately 8 for a sample size of 10 binary sequences.

The minimum pass rate for the random excursion (variant) test is approximately 4 for a sample size of 5 binary sequences.

For further guidelines, construct a probability table using the MAPLE program provided in the addendum section of the documentation.

## Revision history

**Table 3. Document revision history**

Date	Revision	Changes
13-May-2013	1	Initial release.

**Please Read Carefully:**

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

**UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.**

**ST PRODUCTS ARE NOT AUTHORIZED FOR USE IN WEAPONS. NOR ARE ST PRODUCTS DESIGNED OR AUTHORIZED FOR USE IN: (A) SAFETY CRITICAL APPLICATIONS SUCH AS LIFE SUPPORTING, ACTIVE IMPLANTED DEVICES OR SYSTEMS WITH PRODUCT FUNCTIONAL SAFETY REQUIREMENTS; (B) AERONAUTIC APPLICATIONS; (C) AUTOMOTIVE APPLICATIONS OR ENVIRONMENTS, AND/OR (D) AEROSPACE APPLICATIONS OR ENVIRONMENTS. WHERE ST PRODUCTS ARE NOT DESIGNED FOR SUCH USE, THE PURCHASER SHALL USE PRODUCTS AT PURCHASER'S SOLE RISK, EVEN IF ST HAS BEEN INFORMED IN WRITING OF SUCH USAGE, UNLESS A PRODUCT IS EXPRESSLY DESIGNATED BY ST AS BEING INTENDED FOR "AUTOMOTIVE, AUTOMOTIVE SAFETY OR MEDICAL" INDUSTRY DOMAINS ACCORDING TO ST PRODUCT DESIGN SPECIFICATIONS. PRODUCTS FORMALLY ESCC, QML OR JAN QUALIFIED ARE DEEMED SUITABLE FOR USE IN AEROSPACE BY THE CORRESPONDING GOVERNMENTAL AGENCY.**

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2013 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

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