M2351 random number generator validation using NIST statistical test suite

Application Note for 32-bit NuMicro® Family

Document Information

Abstract	Nuvoton secure microcontroller platform supports random number generator (RNG) for secure applications. This document provides the guide line for how to verify the randomness of the random number generated with Nuvoton RNG. The verification is based on NIST 800-22r1a.
Apply to	NuMicro® Cortex®- M2351 series

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

Table of Contents

1 Overview

Nuvoton secure microcontroller platform (NuSMP) provides random number generator (RNG) function. The RNG is implemented by TRNG (True Random Number Generator) and PRNG (Pseudorandom Number Generator). TRNG is used to provide an unpredictable and uncontrollable random number and it is implemented based on analog random source. The random number of TRNG will be used as a seed for PRNG. PRNG is a deterministic random number generator to provide better speed of random bits generating.

The RNG is important for the cryptographic to meet security requirements. Therefore, it is necessary to have some way to guaranty the quality of the random numbers of RNG.

In this document, the guideline to verify the randomness of the output of RNG is provided. The verification is based on NIST Statistical Test Suite SP 800-22rev1a.

2 Generate Random Number

In M2351, the random number could be generated by calling MKROM API. The relative sample code could be found at

bsp\SampleCode\MKROM\CRYPTOLibDemo\TRNG_Verify

Note: The latest BSP could be found at https://github.com/OpenNuvoton.

The relative APIs are:

*int32_t XTRNG_RandomInit(XTRNG_T *rng, uint32_t opt);*

Which is used to initial the random number generator hardware including TRNG and PRNG.

*int32_t XTRNG_Random(XTRNG_T *rng, uint8_t *p, uint32_t size);*

Which is used to generate random number.

2.1 Example Code

Because the verification of NIST SP800-22 needs huge of random bits, the example code is implemented with a loop to generate huge random bits. The generated random numbers will be print to console through UART.

```
int main(void)
{
    int32_t  i,j;
    XTRNG_T rng;
    uint32_t au32RandVal[8];
    uint8_t *pu8Buf;
    uint32_t u32Sum, u32Val;


    /* Unlock protected registers */
    SYS_UnlockReg();

    /* Init System, peripheral clock and multi-function I/O */
    SYS_Init();
```

```
    /* Init UART for printf */
    UART_Init();

    /* Initial TRNG */
    XTRNG_RandomInit(&rng, XTRNG_PRNG | XTRNG_LIRC32K);

    pu8Buf = (uint8_t *)au32RandVal;

    /* Waiting for press any key */
    printf("Press any key to start ...\n");
    getchar();

    printf("//-- START --//\n");

    /* Checksum is used to double check if data transfer ok */
    u32Sum = 0;

    /* 512000 bits * 10 blocks is required for NIST SP800-22 analysis */
    /* Total generated bits are 20001*32*8 = 5120256 bits */
    for (i = 0; i < 20001; i++)
    {
        XTRNG_Random(&rng, pu8Buf, 8*4);

        for(j=0;j<8;j++)
        {
            u32Val = au32RandVal[j];
            u32Sum += u32Val;
            printf("%08x", u32Val);
        }

        /* A short delay could let data trasnfer be more stable for such husge data counts
*/
        if((i&0xff) == 0)
        {
            CLK_SysTickDelay(1000);
        }

    }
    printf("\n//-- END --//\n");
    printf("Random Check Sum = %u\n", u32Sum);
```

```
    while(1) {__WFI();}
}
```

3 NIST SP800-22 test suite

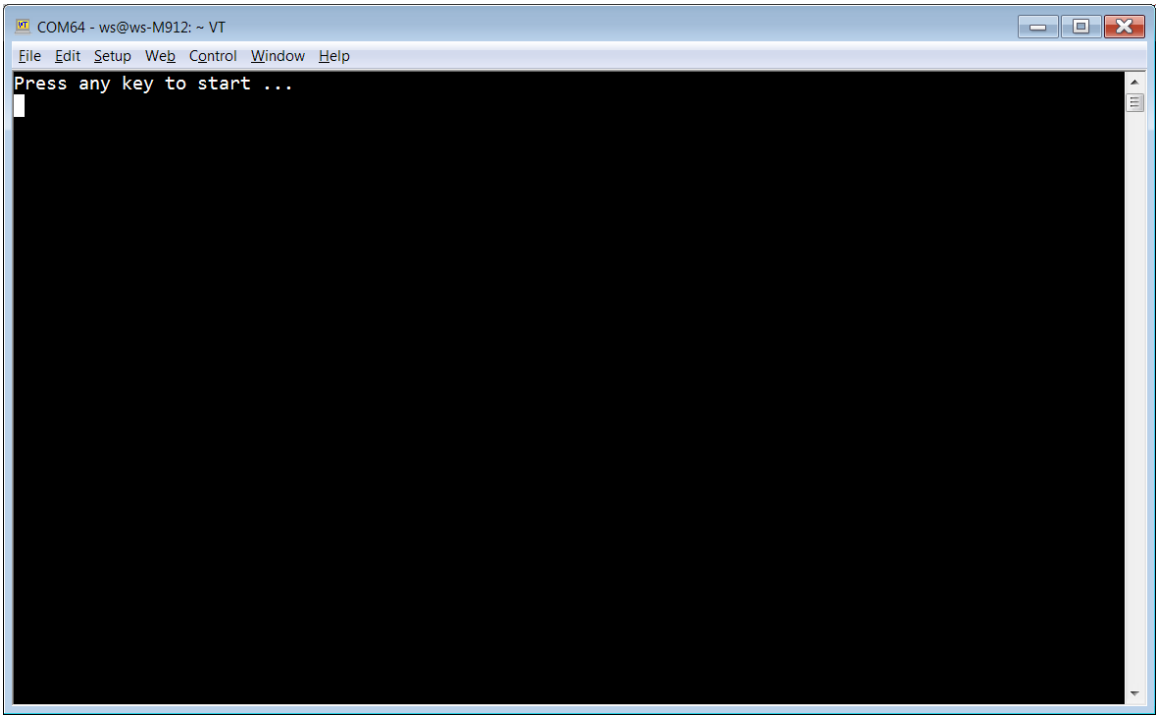
NIST SP800-22 is statistical test suite is a software tool developed by NIST to verify the quality of random generators for cryptographic applications. The download link is as:

https://csrc.nist.gov/projects/random-bit-generation/documentation-and-software

In this document, “sts-2.1.2” test suite is used.

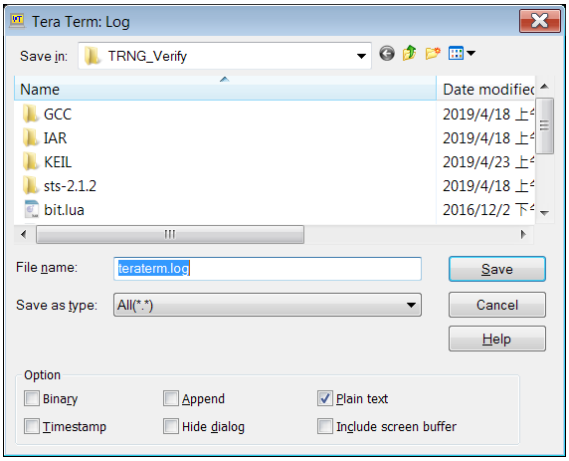
3.1 Firmware to Generate Random Number

To generate random number for analysis, the sample code “TRNG_Verify” must be download to M2351. When the sample code executing ok, the message of console would like bellow figure:



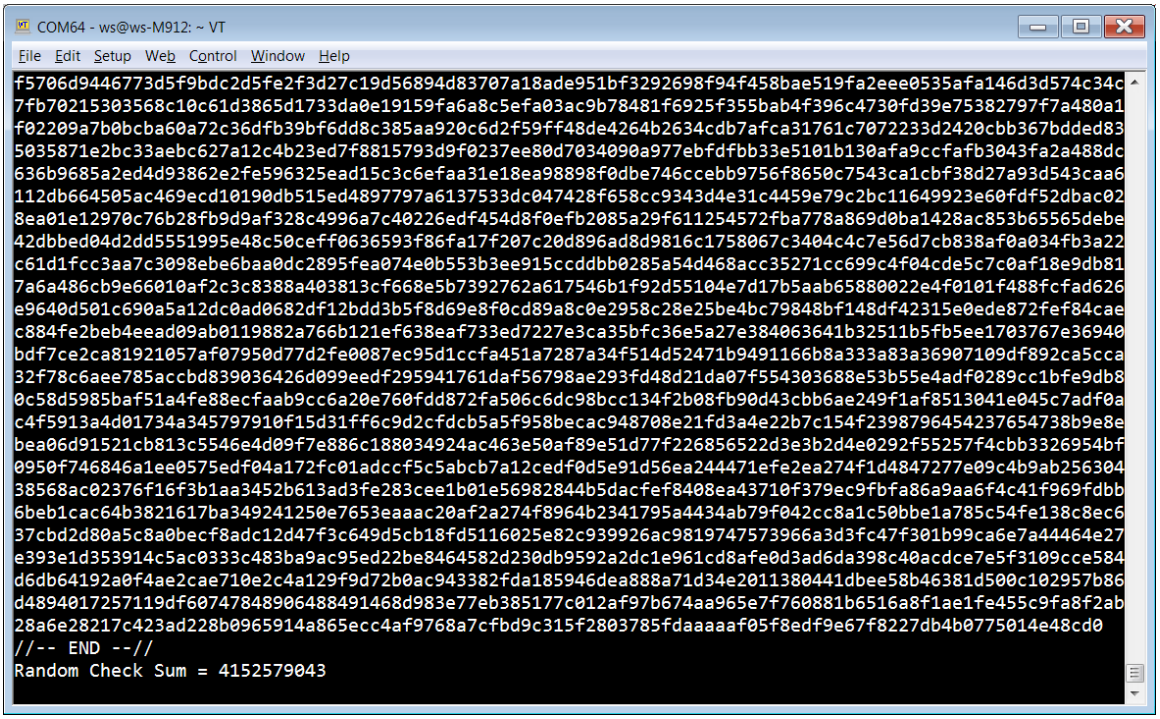
A message “Press any key to start …” is shown on screen. It means the firmware is ready to print out the random numbers which generated by RNG. User needs to enable log function of terminal to log the output random numbers. .

In this document, the terminal tool “Tera Term” is used. To enable log function, click File->Log…

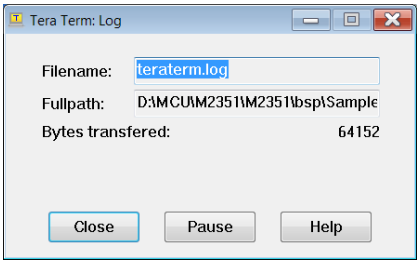


Set log file name and select “Plain text”.

Then press any key in console to start print random numbers in hex. Because it needs to log enough data for analysis tool, it takes more than one more minutes to print data. The final console screen would like this:



After the printing finished, close the log dialog.



Because the NIST analysis tool needs binary input file format, the log file needs to be converted to plan binary file. The convert tool could be found at the same folder of sample code. The command is:

conv.exe [log file] [output file]

For example:

conv teraterm.log out.bin

The convert tool will convert the log file to binary file and verify the checksum. If the checksum fail, it is necessary to re-log the number again. If It converts ok, it shows [PASSED!] on command consoles. Then we can run NIST analysis tool to verify the random numbers.

3.2 Analysis the Random Number

After capturing the log file of random data and convert to binary file, user can start to analyze the data by NIST analysis tool now. First, we need to download and unzip the tool. For example, the tool is downloaded and unzip to

\bsp\SampleCode\MKROM\CRYPTOLibDemo\TRNG_Verify\sts-2.1.2

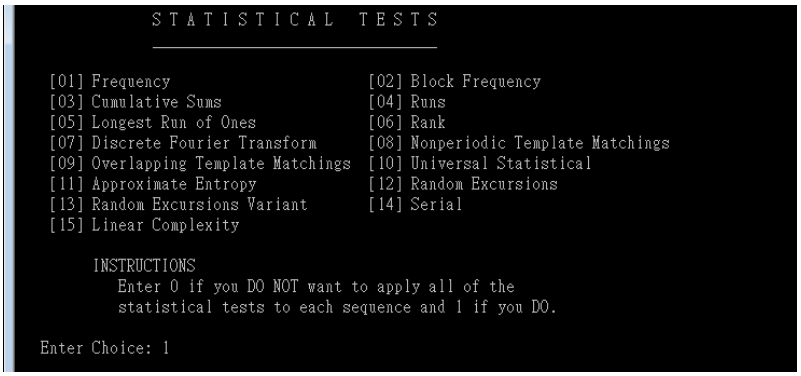
Then, execute the tool by command:

access.exe 512000

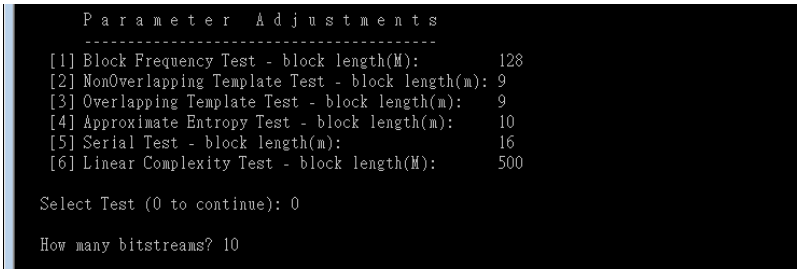
The parameter means 512000 bits per block.

Choice ‘0’ to select “Input File” and enter the file name which is prepared in previous section: “out.bin”. The random data captured in out.bin contains 10 blocks.

The next step is to choice ‘1’ to apply all statistical tests.



Enter bitstreams number 10 and it means 10 random data blocks.



Finally, set the input file format. Here “Binary” is selected and the tool will start to process the analysis.

After statistical testing done, the report could be found at

sts-2.1.2\experiments\AlgorithmTesting

4 Conclusion

This application note describes how to verify the randomness of M2351 RNG function including the firmware to generate the random number and the tool which used to analyze the generated random numbers. Total 15 NIST SP800-22 tests could be passed with M2351 RNG function under this verification. The final report could be found at appendix of this application note.

Appendix A Test Report

```
------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
------------------------------------------------------------------------------
   generator is <../out.bin>
------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
------------------------------------------------------------------------------
  3   0   0   1   0   1   1   0   3   1  0.213309     10/10      Frequency
  1   0   2   0   1   2   1   1   2   0  0.739918     10/10      BlockFrequency
  2   1   0   1   0   1   2   0   2   1  0.739918     10/10      CumulativeSums
  2   2   0   0   1   0   2   1   0   2  0.534146     10/10      CumulativeSums
  1   1   3   1   0   1   0   0   1   2  0.534146     10/10      Runs
  0   1   0   1   3   2   0   1   1   1  0.534146     10/10      LongestRun
  0   3   3   2   0   0   1   0   1   0  0.122325     10/10      Rank
  0   1   2   2   0   2   2   0   1   0  0.534146     10/10      FFT
  2   0   0   0   0   1   1   3   3   0  0.122325     10/10      NonOverlappingTemplate
  2   2   1   1   1   2   1   0   0   0  0.739918     10/10      NonOverlappingTemplate
  0   0   0   2   2   1   0   3   1   1  0.350485     10/10      NonOverlappingTemplate
  0   2   0   2   2   1   0   0   1   2  0.534146     10/10      NonOverlappingTemplate
  0   2   2   2   0   1   0   0   1   2  0.534146     10/10      NonOverlappingTemplate
  0   2   0   0   0   2   3   2   0   1  0.213309     10/10      NonOverlappingTemplate
  1   1   1   1   0   1   0   1   3   1  0.739918     10/10      NonOverlappingTemplate
  0   0   2   1   0   3   2   0   2   0  0.213309     10/10      NonOverlappingTemplate
  1   0   2   1   2   1   1   1   1   0  0.911413     10/10      NonOverlappingTemplate
  3   1   0   1   3   0   1   0   1   0  0.213309     10/10      NonOverlappingTemplate
  2   3   0   0   0   0   0   2   1   2  0.213309      9/10      NonOverlappingTemplate
  1   1   1   3   0   1   0   1   1   1  0.739918     10/10      NonOverlappingTemplate
  1   1   3   1   0   2   0   0   2   0  0.350485      9/10      NonOverlappingTemplate
  0   1   3   2   0   0   0   0   2   2  0.213309     10/10      NonOverlappingTemplate
  1   0   3   1   1   0   1   1   2   0  0.534146     10/10      NonOverlappingTemplate
  1   0   3   0   2   1   1   1   1   0  0.534146     10/10      NonOverlappingTemplate
  2   2   0   0   3   0   1   1   0   1  0.350485      9/10      NonOverlappingTemplate
```

2	1	2	1	1	0	0	0	1	2	0.739918	10/10	NonOverlappingTemplate
1	0	0	2	0	0	5	1	1	0	0.008879	10/10	NonOverlappingTemplate
2	1	0	0	2	2	0	2	0	1	0.534146	10/10	NonOverlappingTemplate
4	1	0	2	0	1	1	1	0	0	0.122325	10/10	NonOverlappingTemplate
4	0	1	0	0	3	0	0	1	1	0.035174	9/10	NonOverlappingTemplate
1	2	0	3	0	1	1	2	0	0	0.350485	10/10	NonOverlappingTemplate
1	0	2	0	2	0	1	1	3	0	0.350485	10/10	NonOverlappingTemplate
2	1	0	2	2	0	2	1	0	0	0.534146	9/10	NonOverlappingTemplate
1	1	1	0	4	0	1	1	0	1	0.213309	10/10	NonOverlappingTemplate
0	1	3	1	0	1	0	0	3	1	0.213309	10/10	NonOverlappingTemplate
1	1	0	0	1	0	3	3	1	0	0.213309	10/10	NonOverlappingTemplate
0	0	0	2	1	2	1	1	1	2	0.739918	10/10	NonOverlappingTemplate
2	2	1	0	0	0	0	0	1	4	0.066882	9/10	NonOverlappingTemplate
0	1	0	2	1	2	2	2	0	0	0.534146	10/10	NonOverlappingTemplate
0	0	1	2	2	1	0	1	2	1	0.739918	10/10	NonOverlappingTemplate
3	0	2	1	0	2	0	1	1	0	0.350485	10/10	NonOverlappingTemplate
2	1	1	0	1	0	2	1	1	1	0.911413	10/10	NonOverlappingTemplate
2	1	2	2	1	0	0	0	1	1	0.739918	10/10	NonOverlappingTemplate
0	1	2	1	0	1	0	2	1	2	0.739918	10/10	NonOverlappingTemplate
1	1	1	1	0	1	2	1	2	0	0.911413	10/10	NonOverlappingTemplate
1	0	2	3	1	1	0	1	0	1	0.534146	10/10	NonOverlappingTemplate
0	3	0	2	1	0	2	1	1	0	0.350485	10/10	NonOverlappingTemplate
0	2	2	1	2	0	1	1	0	1	0.739918	10/10	NonOverlappingTemplate
2	0	2	2	0	1	1	1	1	0	0.739918	9/10	NonOverlappingTemplate
0	1	1	2	1	1	2	1	0	1	0.911413	10/10	NonOverlappingTemplate
0	0	0	2	1	1	2	1	3	0	0.350485	10/10	NonOverlappingTemplate
0	0	1	0	0	2	2	0	3	2	0.213309	10/10	NonOverlappingTemplate
0	0	0	0	3	0	3	1	3	0	0.035174	10/10	NonOverlappingTemplate
1	0	0	0	1	1	2	3	1	1	0.534146	10/10	NonOverlappingTemplate
2	0	1	0	1	2	0	2	1	1	0.739918	9/10	NonOverlappingTemplate
1	1	1	0	4	2	1	0	0	0	0.122325	10/10	NonOverlappingTemplate
0	2	1	3	0	1	0	0	3	0	0.122325	10/10	NonOverlappingTemplate
0	0	1	4	1	1	1	1	1	0	0.213309	10/10	NonOverlappingTemplate
3	0	1	0	3	0	1	0	1	1	0.213309	9/10	NonOverlappingTemplate
3	1	1	2	0	0	1	0	0	2	0.350485	10/10	NonOverlappingTemplate
1	0	1	2	0	2	0	1	1	2	0.739918	10/10	NonOverlappingTemplate

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

1	0	0	1	0	0	1	0	3	4	0.035174	10/10	NonOverlappingTemplate
0	0	1	1	1	1	2	2	0	2	0.739918	10/10	NonOverlappingTemplate
4	1	0	0	2	2	0	0	0	1	0.066882	10/10	NonOverlappingTemplate
3	1	1	1	1	0	2	0	1	0	0.534146	10/10	NonOverlappingTemplate
2	3	0	3	0	0	1	1	0	0	0.122325	10/10	NonOverlappingTemplate
1	1	1	0	1	1	2	1	1	1	0.991468	10/10	NonOverlappingTemplate
0	1	1	4	1	0	0	1	1	1	0.213309	10/10	NonOverlappingTemplate
2	2	0	2	2	0	0	1	0	1	0.534146	10/10	NonOverlappingTemplate
2	1	2	2	0	2	1	0	0	0	0.534146	9/10	NonOverlappingTemplate
2	1	0	0	1	1	1	1	2	1	0.911413	10/10	NonOverlappingTemplate
2	3	1	1	0	0	0	1	1	1	0.534146	10/10	NonOverlappingTemplate
1	0	2	0	1	1	0	3	1	1	0.534146	10/10	NonOverlappingTemplate
2	1	0	0	1	3	2	0	1	0	0.350485	9/10	NonOverlappingTemplate
1	0	1	2	0	1	1	2	0	2	0.739918	9/10	NonOverlappingTemplate
1	2	1	1	0	2	1	0	1	1	0.911413	9/10	NonOverlappingTemplate
1	0	0	0	2	0	1	3	2	1	0.350485	10/10	NonOverlappingTemplate
2	2	0	1	1	0	0	2	1	1	0.739918	10/10	NonOverlappingTemplate
1	4	1	1	1	0	0	0	1	1	0.213309	9/10	NonOverlappingTemplate
1	0	1	0	1	1	1	1	1	3	0.739918	10/10	NonOverlappingTemplate
2	1	2	0	2	0	0	1	1	1	0.739918	10/10	NonOverlappingTemplate
0	1	1	1	0	1	2	1	2	1	0.911413	10/10	NonOverlappingTemplate
0	1	0	0	0	1	2	2	2	2	0.534146	10/10	NonOverlappingTemplate
2	0	1	2	2	0	0	2	0	1	0.534146	9/10	NonOverlappingTemplate
2	2	1	1	2	1	0	0	1	0	0.739918	10/10	NonOverlappingTemplate
0	1	2	0	0	0	2	0	1	4	0.066882	10/10	NonOverlappingTemplate
1	0	2	0	0	1	2	0	2	2	0.534146	10/10	NonOverlappingTemplate
3	2	0	1	3	0	1	0	0	0	0.122325	10/10	NonOverlappingTemplate
1	0	1	1	3	2	0	0	1	1	0.534146	10/10	NonOverlappingTemplate
0	2	1	1	1	1	1	2	0	1	0.911413	10/10	NonOverlappingTemplate
2	1	1	1	0	1	1	1	0	2	0.911413	10/10	NonOverlappingTemplate
3	2	0	0	1	2	1	1	0	0	0.350485	10/10	NonOverlappingTemplate
2	2	1	0	0	0	1	2	1	1	0.739918	10/10	NonOverlappingTemplate
1	1	1	1	4	0	1	0	0	1	0.213309	10/10	NonOverlappingTemplate
0	1	1	0	2	1	2	0	1	2	0.739918	10/10	NonOverlappingTemplate
0	1	1	2	1	0	1	1	0	3	0.534146	10/10	NonOverlappingTemplate
0	3	0	3	2	1	1	0	0	0	0.122325	10/10	NonOverlappingTemplate

4	2	0	1	2	0	0	1	0	0	0.066882	10/10	NonOverlappingTemplate
1	2	1	2	0	1	2	1	0	0	0.739918	10/10	NonOverlappingTemplate
1	1	1	0	2	0	0	0	2	3	0.350485	10/10	NonOverlappingTemplate
2	2	0	0	1	1	0	0	1	3	0.350485	10/10	NonOverlappingTemplate
1	2	1	1	1	1	1	2	0	0	0.911413	10/10	NonOverlappingTemplate
3	1	1	0	0	0	2	1	0	2	0.350485	10/10	NonOverlappingTemplate
1	2	1	0	1	2	2	0	0	1	0.739918	9/10	NonOverlappingTemplate
2	1	1	2	0	0	1	3	0	0	0.350485	10/10	NonOverlappingTemplate
0	1	4	1	0	1	0	1	2	0	0.122325	10/10	NonOverlappingTemplate
0	2	0	0	2	0	1	3	1	1	0.350485	10/10	NonOverlappingTemplate
0	2	2	1	1	0	0	2	2	0	0.534146	10/10	NonOverlappingTemplate
3	3	0	2	1	0	0	1	0	0	0.122325	10/10	NonOverlappingTemplate
0	1	2	0	1	0	0	2	4	0	0.066882	10/10	NonOverlappingTemplate
1	1	1	0	0	3	0	4	0	0	0.035174	10/10	NonOverlappingTemplate
2	0	1	1	2	1	1	1	0	1	0.911413	10/10	NonOverlappingTemplate
5	1	0	1	0	0	1	1	1	0	0.017912	10/10	NonOverlappingTemplate
0	0	2	1	1	2	2	0	1	1	0.739918	10/10	NonOverlappingTemplate
2	4	0	1	1	1	1	0	0	0	0.122325	9/10	NonOverlappingTemplate
2	1	0	1	1	1	1	0	1	2	0.911413	10/10	NonOverlappingTemplate
2	1	1	2	2	0	1	0	1	0	0.739918	10/10	NonOverlappingTemplate
2	1	4	0	0	1	1	0	1	0	0.122325	9/10	NonOverlappingTemplate
2	1	0	2	0	1	3	0	0	1	0.350485	9/10	NonOverlappingTemplate
0	2	1	0	2	2	1	1	0	1	0.739918	10/10	NonOverlappingTemplate
3	1	1	0	0	2	0	2	1	0	0.350485	10/10	OverlappingTemplate
1	1	1	0	1	3	2	1	0	0	0.534146	10/10	Universal
1	1	2	0	1	0	2	2	1	0	0.739918	10/10	ApproximateEntropy
0	1	1	0	0	0	0	1	1	0	----	4/4	RandomExcursions
0	1	1	1	0	0	1	0	0	0	----	4/4	RandomExcursions
0	1	0	0	0	0	2	1	0	0	----	4/4	RandomExcursions
0	0	0	1	2	0	0	0	1	0	----	4/4	RandomExcursions
0	0	0	0	0	0	0	1	1	2	----	4/4	RandomExcursions
0	1	0	0	0	1	0	1	1	0	----	4/4	RandomExcursions
1	2	0	0	0	0	1	0	0	0	----	4/4	RandomExcursions
0	1	1	1	0	1	0	0	0	0	----	4/4	RandomExcursions
0	1	0	1	0	0	2	0	0	0	----	4/4	RandomExcursionsVariant
0	1	1	0	0	1	0	0	1	0	----	4/4	RandomExcursionsVariant

0	1	1	0	0	1	0	0	1	0	----	4/4	RandomExcursionsVariant
0	1	1	0	0	0	1	0	0	1	----	4/4	RandomExcursionsVariant
0	2	0	0	0	0	0	0	1	1	----	4/4	RandomExcursionsVariant
1	0	1	0	0	0	0	0	1	1	----	4/4	RandomExcursionsVariant
0	0	1	0	0	1	1	0	1	0	----	4/4	RandomExcursionsVariant
0	0	0	0	2	0	0	1	1	0	----	4/4	RandomExcursionsVariant
0	0	0	1	1	0	0	0	2	0	----	4/4	RandomExcursionsVariant
0	0	1	0	1	0	0	0	1	1	----	4/4	RandomExcursionsVariant
1	0	0	0	0	2	0	0	1	0	----	4/4	RandomExcursionsVariant
1	2	0	0	0	0	0	0	1	0	----	4/4	RandomExcursionsVariant
0	1	2	0	1	0	0	0	0	0	----	4/4	RandomExcursionsVariant
0	0	1	1	1	0	0	1	0	0	----	4/4	RandomExcursionsVariant
0	0	0	1	1	0	1	1	0	0	----	4/4	RandomExcursionsVariant
0	0	0	1	0	1	0	2	0	0	----	4/4	RandomExcursionsVariant
0	0	0	0	1	0	1	0	2	0	----	4/4	RandomExcursionsVariant
0	0	0	0	2	0	0	0	2	0	----	4/4	RandomExcursionsVariant
2	1	0	0	2	0	0	2	1	2	0.534146	10/10	Serial
0	2	0	0	1	2	1	2	1	1	0.739918	10/10	Serial
0	1	1	3	0	3	1	0	0	1	0.213309	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 = 10 binary sequences.

The minimum pass rate for the random excursion (variant) test
is approximately = 3 for a sample size = 4 binary sequences.

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

- -

Revision History

Date	Revision	Description
2019.04.23	1.00	1. Initially issued.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.