AN4230
应用笔记

使用 NIST 统计测试套件验证 STM32 微控制器随机数生成

前言

很多标准都规定了构造要求和参考、随机数发生器 (RNG) 的验证和使用，以便检验其生成的输出是否是真正的随机数。

本应用笔记中包含的指南用于检验所选 STM32 微控制器中嵌入的随机数发生器外设生成的数字的随机性。本次验证是基于美国国家标准技术局 (NIST) 统计测试套件 (STS) SP 800-22rev1a （2010 年 4 月）进行的。

本文档结构如下：

- STM32 微控制器随机数发生器概述 （请参见*第 1 节*）
- NIST SP800-22b 测试套件 （请参见*第 2 节*）
- 运行 NIST SP800-22b 测试和分析需要执行的步骤 （请参见*第 3 节*）

表 1. 适用产品

类型	产品系列
微控制器	STM32F2 系列、 STM32F4 系列、 STM32F7 系列、 STM32L0 系列、 TM32L4 系列。

目录

图片索引

1 STM32 微控制器随机数发生器

1.1 前言

为加密应用程序使用的随机数发生器 (RNG) 通常会生成由随机的 0 或 1 位组成的序列。

随机数发生器基本上分为两类，分别是：

1. 确定性随机数发生器或伪随机数发生器 (PRNG)：
   确定性 RNG 包含的算法会通过名为种子的初始值生成位序列。为确保向前不可预测性，获取种子时必须多加留意。如果已知种子和生成算法， PRNG 生成的数值是完全可预测的。由于很多情况下生成算法是公开可用的，因此种子必须保密，并通过 TRNG 来生成。
2. 非确定性随机数发生器或真随机数发生器 (TRNG)：
   非确定性 RNG 生成的随机性取决于一些不受人为控制的不可预测物理源 （即熵源）。

在 STM32 微控制器上采用的 RNG 硬件外设以及在*第 1.2.1 节*中介绍的 RNG 硬件外设属于真随机数发生器。

1.2 STM32 微控制器实施说明

1.2.1 真随机数发生器

下表列出的是嵌入了 RNG 硬件外设的 STM32 微控制器线列：

表 2. 嵌入 RNG 硬件外设的 STM32 线列

STM32 系列	STM32 线列
STM32F2	STM32F2x5 STM32F2x7
STM32F4	STM32F405/415 STM32F407/417 STM32F410 STM32F427/437 STM32F429/439 STM32F469/479
STM32F7	STM32F7x5 STM32F7x6
STM32L0	STM32L05x STM32L06x STM32L072/073
STM32L4	STM32L4x6

STM32 微控制器上使用的真随机数发生器外设基于模拟电路。该电路生成的连续模拟噪声将用于 RNG 处理，以生成 32 位随机数。

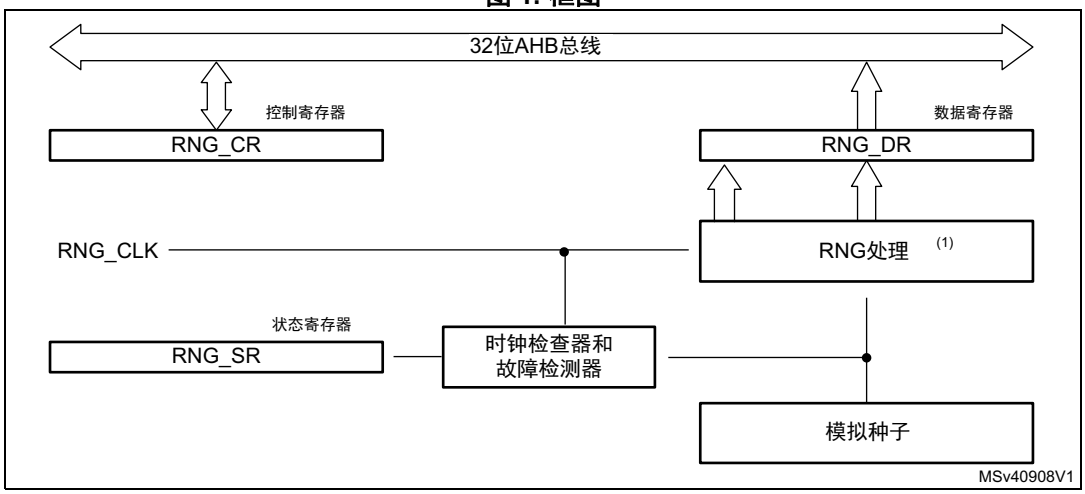
该模拟电路由几个环形振荡器组成，振荡器的输出进行异或运算。

RNG 处理由采用恒定频率的专用时钟计时，对于微控制器的子设备，还可以使用频率值不同的时钟来计时。

有关 RNG 外设的详细信息，请参见 STM32 微控制器参考手册。

图 1 给出了 STM32 微控制器的 TRNG 框图。

图 1. 框图



2 NIST SP800-22b 测试套件

2.1 前言

NIST SP800-22b 统计测试套件已使用由*国家标准技术局 (*NIST) 开发的统计数据测量套件 (sts) 实现，可检验用于加密应用的随机数发生器的质量。标题为 “*A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications*”” 的文章对该套件进行了全面介绍。

2.2 NIST SP800-22b 测试套件说明

NIST SP800-22b 统计测试套件 “sts-2.1.1” 是由*国家标准技术局*开发的软件包，可在 NIST 网站上下载：请在 http://csrc.nist.gov 上搜索 “*NIST Statistical Test Suite download*”。

源代码使用 ANSI C 编写。 NIST 统计测试套件由 15 种测试组成，用于测试二进制序列的随机性。这些测试主要针对序列中可能存在的各类非随机性问题。从这一角度，可将测试套件分为以下几类：

频率测试

- 频率 （单比特）测试：
  衡量 0 和 1 在序列中的分布情况，并检查结果是否与真随机数序列的预期结果相似。
- 块中的频率测试：
  检查 M 位块中 1 的频率是否近似为预期通过随机性原理得到 M/2。
- 运行测试：
  评估不同长度的 1 和 0 的预期运行总数是否是随机序列的预期结果。
- 测试块中运行最长的 “1”：
  检查序列中的长运行 “1”：

线性测试

- 二进制矩阵秩测试：
  评估 32x32 二进制矩阵秩的分布。
- 线性复杂度测试：
  确定有限序列的线性复杂度。

相关性测试 （通过傅里叶变换）

- 离散傅里叶变换 （频谱）测试：
  通过基于离散傅里叶变换的频谱测试评估位串的谱频率。此测试易受序列中的周期性影响。

查找特殊字符串测试

- 非重叠模板匹配测试：
  评估 m 位非周期性组合的频率。
- 重叠模板匹配测试：
  评估 m 位周期性模板的频率。

熵测试

- Maurer“ 通用统计 ” 测试：
  评估 L 位块二进制序列的压缩率。
- 连续测试：
  评估所有 2^m m 位块的分布。

注： *对于 m = 1 的情况，连续测试相当于第 2.2 节的频率测试。*

- 近似熵测试：
  评估位串的熵，将所有 m 位组合的频率与所有 (m+1) 位组合的频率进行对比。

随机游走测试

- 累积和测试：
  评估部分序列的和是否过大或过小；用于指示过多的 0 或 1。
- 随机偏移测试：
  评估随机游走周期内的状态分布。
- 随机偏移变化测试：
  检测与达到不同随机游走状态的预期次数的偏差。

上述测试中，每项测试都基于计算出的测试统计值，而测试统计值是测试序列的函数。测试统计值用于计算 Pvalue，其中：

Pvalue 是完美随机数发生器生成的序列随机性小于受测试序列的概率。

有关 NIST 统计测试套件的更多详细信息，请参见 NIST 网站上提供的以下 NIST 文档：

特别出版物“A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications”， 800-22 版本 1a。

3 NIST SP800-22b 测试套件运行和分析

3.1 固件说明

要按照上一节中的说明运行 NIST 统计测试套件，需要使用两个固件，一个固件位于 STM32 微控制器端，另一个位于 NIST SP800-22b 测试套件端。

3.1.1 在 STM32 微控制器端

会根据要求提供固件包，更多详细信息，请联系您当地的意法半导体销售代表。

程序允许使用 STM32 微控制器随机数发生器 (RNG) 外设生成随机数，并允许在工作站上重新获取生成的随机数，以便使用 NIST 统计测试套件进行测试。

会使用每个固件程序生成 10 个由 64000 字节随机数组成的块，因此输出文件将包含 5120000 个随机位，供 NIST 统计测试套件进行测试。

根据 NIST 统计测试套件的建议，输出文件格式应为：

1. 如果私有定义 *FILE_ASCII_FORMAT* 在 main.c 文件中未被注释掉，则为 ASCII 0 和 1 组成的序列
2. 如果私有定义 *FILE_BINARY_FORMAT* 在 main.c 文件中未被注释掉，则为随机字节二进制文件。

更多关于程序说明和设置的详细信息，请参见固件包中的自述文件。

注： *用户可通过 main.c 文件中的 SendToWorkstation() 函数更改 USART 配置。*

用户可通过修改 main.c 文件中的私有定义更改输出值。
#define NUMBER_OF_RANDOM_BITS_TO_GENERATE 512000
#define BLOCK_NUMBER 10

3.1.2 在 NIST SP800-22b 测试套件端

下载到工作站上的 NIST 统计测试套件包 sts-2.1.1 会检验 STM32 微控制器随机数发生器输出文件的随机性。

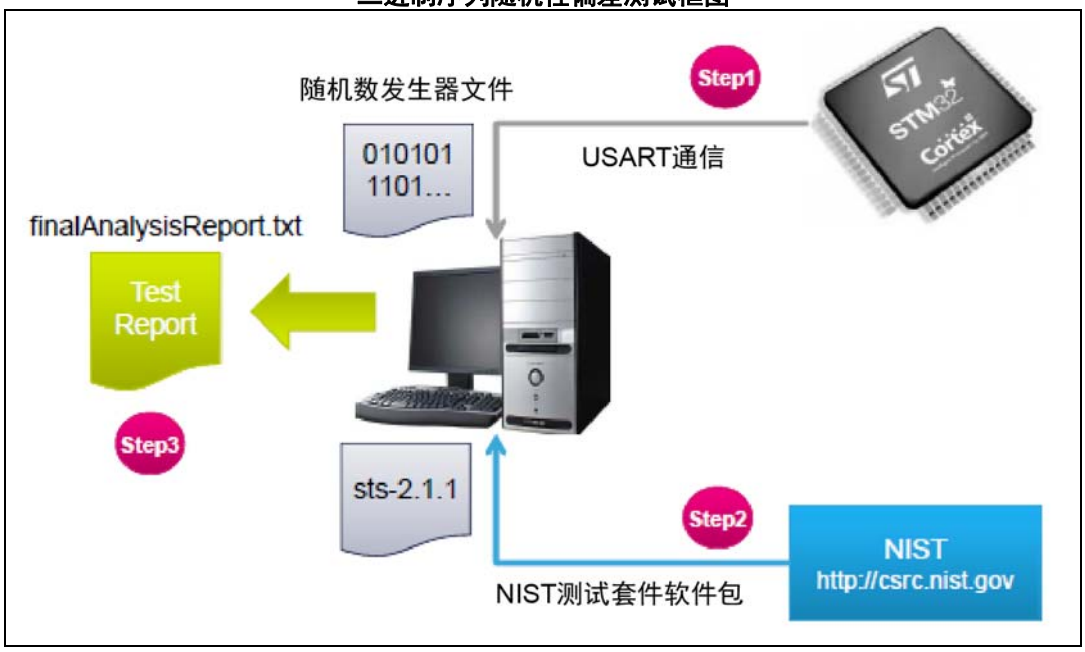
待分析的发生器文件应存储在 *data* 文件夹 (sts-2.1.1\data) 下。

更多关于 NIST 统计测试套件工作原理的详细信息，请参见特殊出版物 “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications”800-22 版本 1a 中的入门知识章节。

3.2 NIST SP800-22b 测试套件步骤

图 2 介绍了使用 NIST 统计测试套件包 sts-2.1.1 验证 STM32 微控制器生成的输出数随机性所需执行的不同步骤。

图 2. 基于 NIST 测试套件的
二进制序列随机性偏差测试框图



3.2.1 第一步：随机数发生器

将 STMicroelectonics 板件连接至工作站。根据板件类型建立连接：

- 零调制解调器母头 / 母头 RS232 线缆
- A 转 mini-B 型 USB 线缆

按照上一节中的说明通过 UART 固件运行 STM32 微控制器 RNG 生成随机数。可使用 HyperTerminal 等终端仿真应用程序 （从 Windows 98 开始 Windows 操作系统随附的程序）将数据存储在工作站上。

3.2.2 第二步：NIST 统计测试

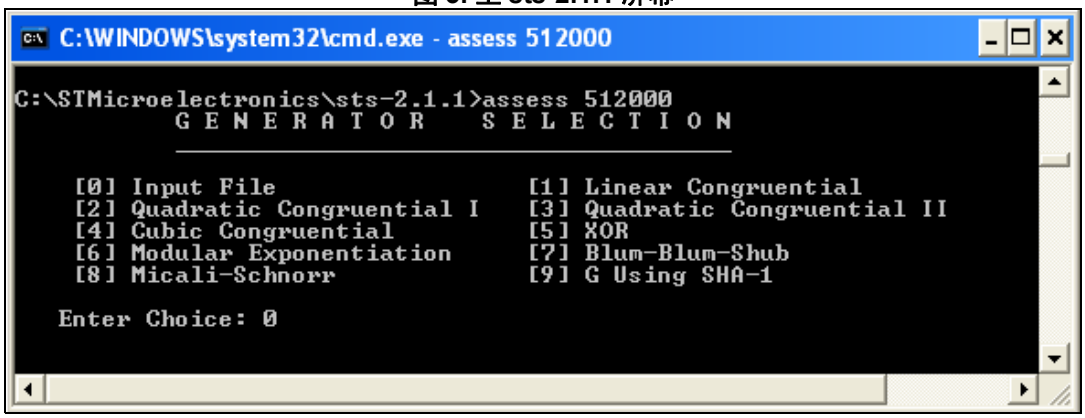
按照 NIST 统计测试套件中的说明使用 visual C++ 编译器编译 sts-2.1.1 程序包，以生成可执行程序。

运行 NIST 统计测试套件程序之后，会显示一系列菜单提示，供用户选择要分析的数据以及要应用的统计测试。

在该应用笔记中， NIST 统计测试套件会在名称 *assess.exe* 下进行编译，并会保存到 NIST_Test_Suite_OutputExample 文件夹下。按照上一节的说明，随机数定义为 *每个块 512000 位*。

会出现*图 3* 所示的第一个屏幕。

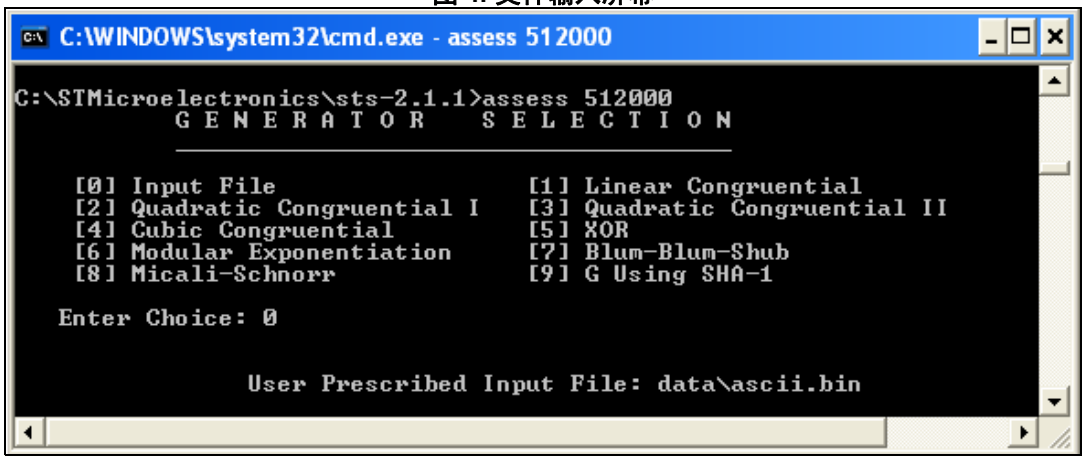
图 3. 主 sts-2.1.1 屏幕



如果输入数值 0，程序会要求输入待测试随机数的文件名和路径。

会出现*图 4* 所示的第二个屏幕。

图 4. 文件输入屏幕

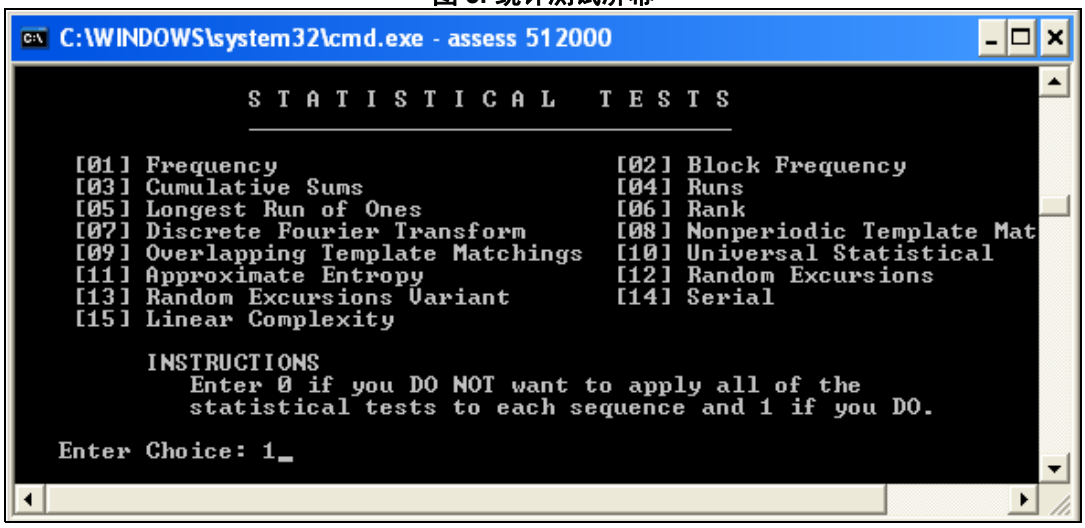


该应用笔记提供的每个系列的示例由两个文件组成，文件是使用 NIST 建议的不同文件格式通过 STM32 微控制器随机数发生器生成的：

1. *ascii.bin*：由 ASCII 0 和 1 组成的序列。
2. *binary.bin*：数据文件中的每个字节包含 8 个数据位。

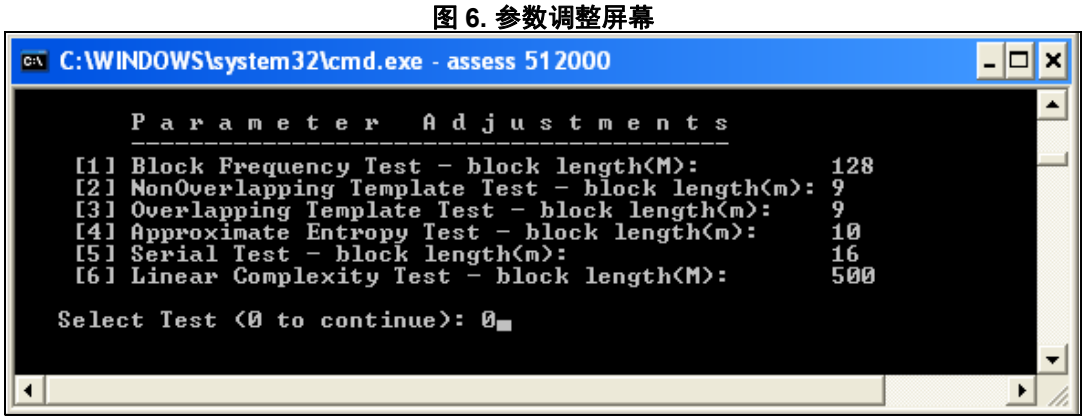
随后， NIST 统计测试套件会显示可通过*图 5* 所示的屏幕运行的 15 个测试。

图 5. 统计测试屏幕



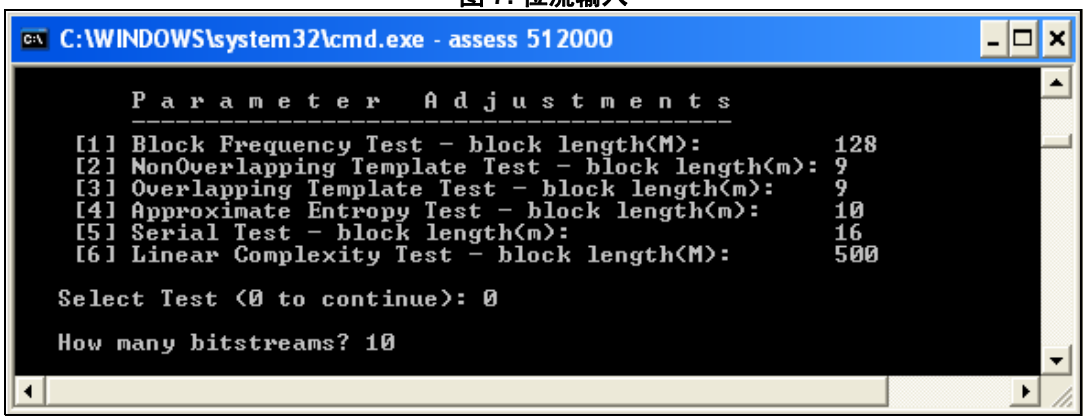
在这种情况下，选择了 “1” 应用到所有统计测试。显示的 *图 6* 可用于更改参数调整。

图 6. 参数调整屏幕



本例中，会保持默认设置，并会选择 “0” 值进入下一步。

图 7. 位流输入

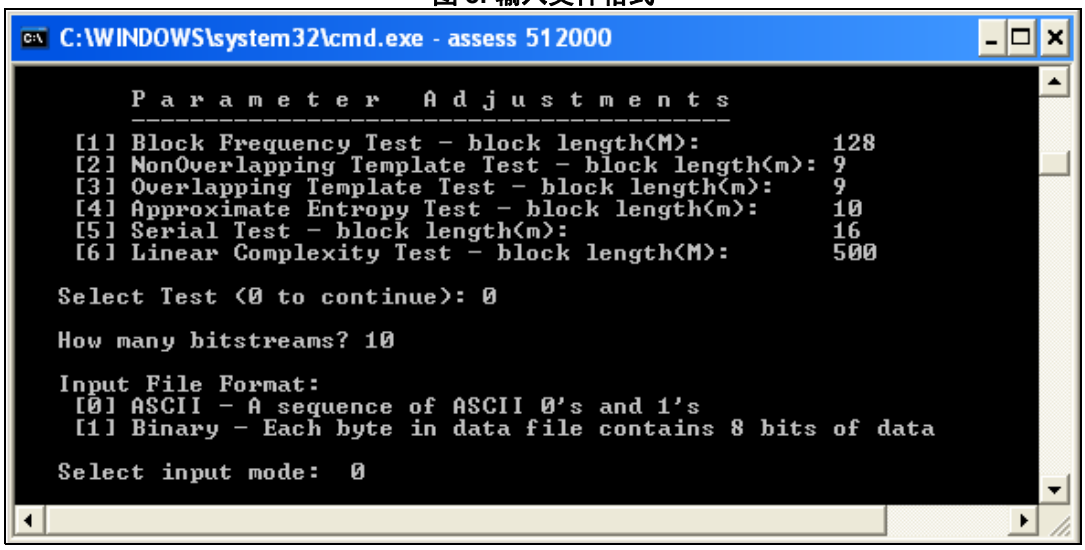


NIST 统计测试套件要求输入位流数；这里输入的是 “10”。

您已选择了 10 个 512000 位构成的块，相当于 5120000 位。

随后必须通过以下屏幕指定文件包含的是以 ASCII 格式存储的位还是以二进制格式存储的十六进制字符串。

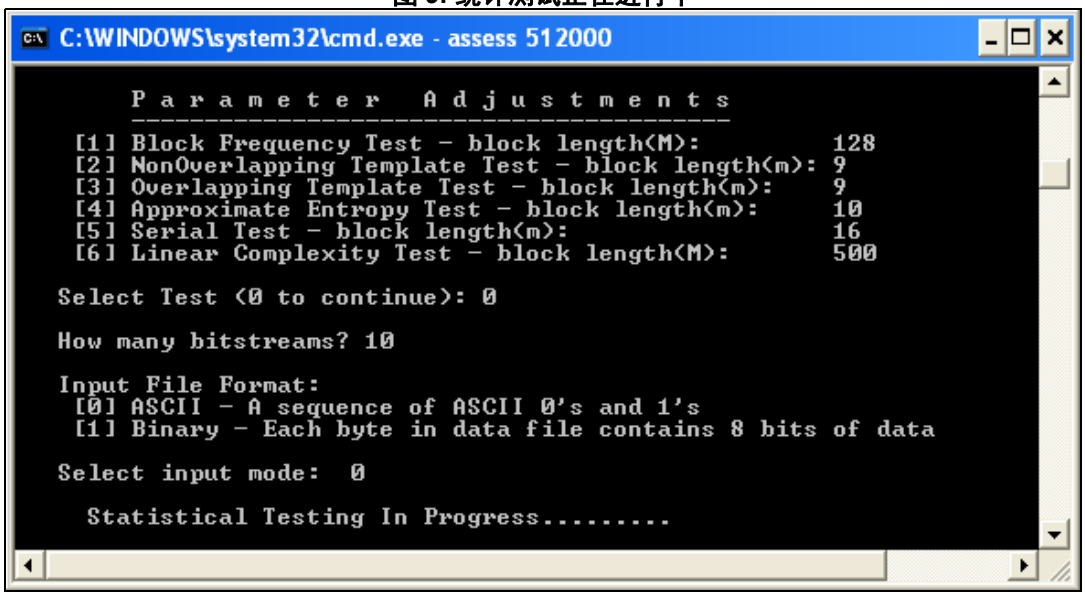
图 8. 输入文件格式



选择了 “0” 值是因为文件采用的是 ASCII 格式。

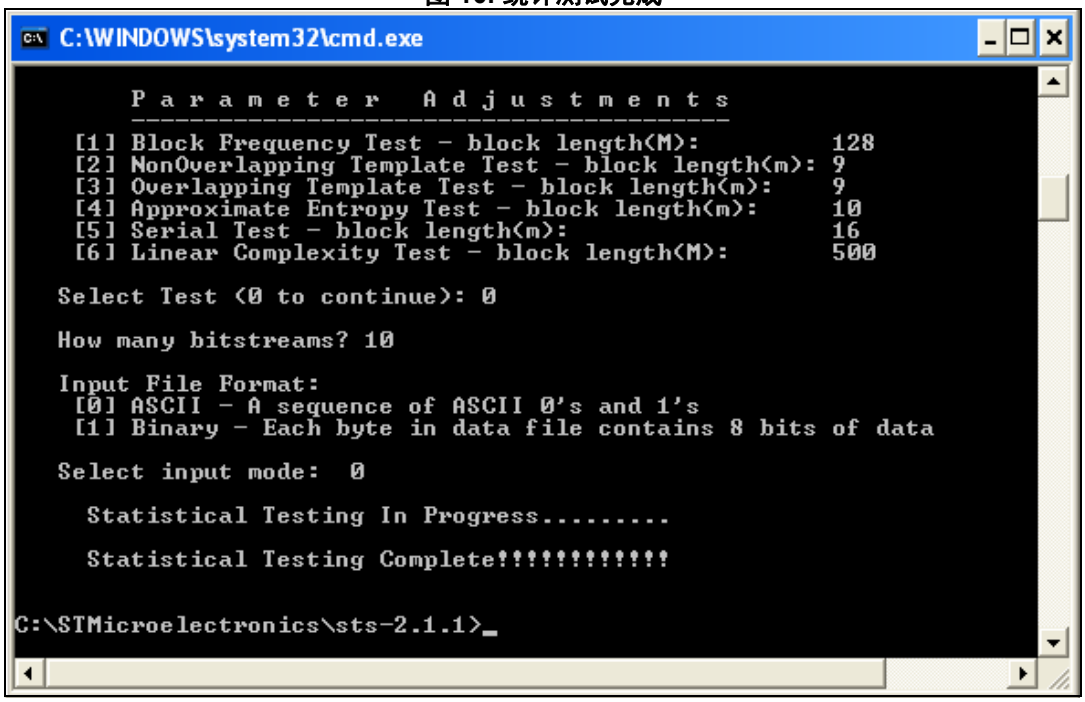
输入所有必填输入后， NIST 统计测试套件会开始对输入文件进行分析。

图 9. 统计测试正在进行中



测试过程结束后，如*图 10* 所示，统计测试结果会出现在 sts-2.1.1\experiments\AlgorithmTesting 中。

图 10. 统计测试完成



3.2.3 第三步：测试报告

NIST 统计测试提供的分析例程有助于对分析结果进行解读。统计测试完成后，会生成一个名为 finalAnalysisReport 的文件，文件保存位置为 sts2.1.1\experiments\AlgorithmTesting。报告包含 15 项测试的实验结果摘要，如*附录 A* 所述。

NIST 统计测试还针对每项测试提供详细报告，报告保存位置为 sts-2.1.1\experiments\AlgorithmTesting\< 测试套件名称 >。

注： *每个系列的两个示例的完整 NIST 统计测试套件输出报告位于 “NIST_Test_Suite_OutputExample”。*

1. *Ascii_File_Format* 示例。本示例包含 2 个文件夹：
   - ***Input_File*** 中包含采用 ascii 格式保存的随机数发生器。
   - ***Final_Analysis_Report*** 包含基于该输入文件的完整 NIST 统计测试套件输出报告、实验结果摘要以及每个测试对应的报告。
2. *Binary_File_Format 示例。*本示例包含 2 个文件夹：
   - ***Input_File*** 中包含采用二进制格式保存的随机数发生器。
   - ***Final_Analysis_Report*** 包含基于该输入文件的完整 NIST 统计测试套件输出报告、实验结果摘要以及每个测试对应的报告。

4 结论

该应用笔记介绍了使用 NIST 统计测试套件 SP800-22rev1a （2010 年 4 月版）检验由 STM32 控制器随机数发生器外设生成的数字随机性的主要规则和步骤。

NIST 统计测试套件的 15 种测试均通过检验，符合在每个支持 RNG 外设的 STM32 系列器件上运行的每项统计测试的最低通过率。

附录 A 附加信息

结果以表格形式表示，表格中包含 p 行、 q 列。

- 行数 p 对应于应用的统计测试数。
- 列数 q=13 的分布如下：
  - 第 1-10 列对应于 P-values10 的频率，
  - 第 11 列是通过应用 chi-square test11 得出的 P-value，
  - 第 12 列是通过的二进制序列所占的比例，
  - 第 13 列是对应的统计测试。

表 3 给出了一个示例。更多详细信息，请参见 sts-2.1.1\experiments\AlgorithmTesting) 下的 finalAnalysisReport 文件。

表 3. 示例

P-VALUES 一致性与通过序列比例对应的结果

发生器为 <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

- -

对于大小为 10 个二进制序列的样本，每次统计测试 （随机偏移 （变化）测试）的最低通过率大概是 8。

对于大小为5 个二进制序列的样本，随机偏移 （变化）测试的最低通过率大概是4。
要获得更具体的指南，请使用文档附录部分中提供的MAPLE 程序构建可能性表。

版本历史

表 4. 文档版本历史

日期	版本	变更
2013 年 5 月 13 日	1	初始版本。
2016 年 6 月 22 日	2	更新了*第 节：前言*。 更新了*第 1 节：STM32 微控制器随机数发生器*。 增加了*图 2：嵌入 RNG 硬件外设的 STM32 线*。 更新了*图 1：框图*。 更新了*第 3.1 节：固件说明*。 更新了*第 3.2.1 节：第一步：随机数发生器*。 更新了*第 3.2.2 节：第二步：NIST 统计测试*。 更新了*第 4 节：结论*。

表 5. 中文文档版本历史

日期	版本	变更
2017 年 1 月 2 日	1	中文初始版本。

重要通知 - 请仔细阅读

意法半导体公司及其子公司 （“ST”）保留随时对 ST 产品和 / 或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于 ST 产品的最新信息。 ST 产品的销售依照订单确认时的相关 ST 销售条款。

买方自行负责对 ST 产品的选择和使用， ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的 ST 产品如有不同于此处提供的信息的规定，将导致 ST 针对该产品授予的任何保证失效。

ST 和 ST 徽标是 ST 的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。