Redefining Security

QUANTUM-SAFE SECURITY WHITE PAPER

Quantis QRNG chips physical model and test results

Origin of quantum randomness and their security

Version 1.1

March 2020

ID QUANTIQUE SA
Chemin de la Marbrerie 3

1227 Carouge/Geneva
Switzerland

T +41 22 301 83 71
F +41 22 301 89 79

info@dquantique.com
www.idquantique.com

Table of contents

1. Introduction

The most crucial feature of a random number generator in cryptographic use is its ability to generate random numbers that are fundamentally unpredictable. To assess the "level of unpredictability" (also called the "amount of entropy"), a clear physical model with no hidden assumptions about the inner workings of the device is necessary. IDQ's QRNG chips generate "entropy" directly from a quantum process that is provably random and unpredictable, and their inner working is simple enough to be succinctly laid out. This quality is at the root of their security.

2. Physical principle for entropy generation in IDQ's QRNG chips

In order to get high quality of entropy, IDQ's patented quantum random number generator (QRNG) chips exploit the fact the number of photons emitted by a common light source fluctuates randomly. These fluctuations, also called "quantum shot noise", are purely of a quantum origin, and are therefore fundamentally random as per the laws of physics: an array of single-photon sensitive pixels is illuminated for a short time during which each pixel receives a random number of incident photons

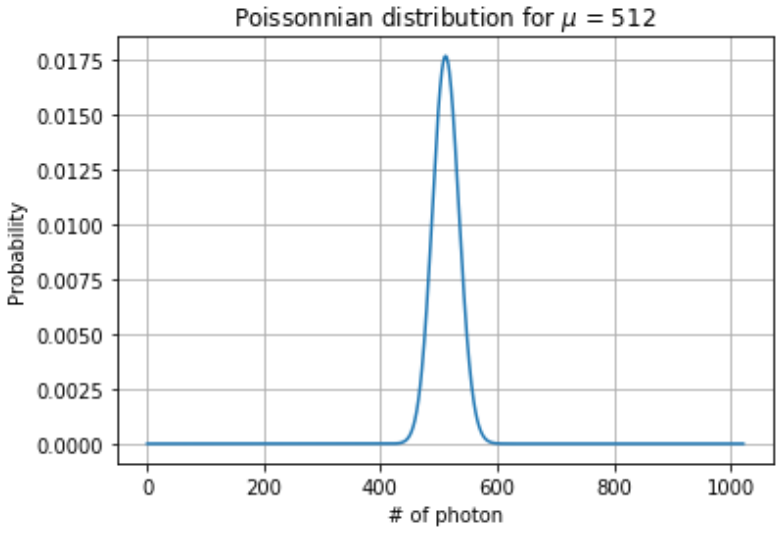


Figure 1: Poisson distribution with a mean value of 512 photons.

that follows the statistics of a *Poisson* distribution. *Figure 1* shows the Poisson distribution with a mean value of 512 as an example. Note that a particular feature of the Poisson distribution is that its mean is equal to its variance.

The structure of all IDQ QRNG chips is shown on *Figure 2*: a light emitting diode (LED) and a CMOS image sensor (CIS) pixel array are respectively integrated inside a QRNG chip as a light source and a multipixel single-photon detector. All pixel outputs are digitized by a single analog-digital converter (ADC). Based on these ADC output values, the number of detected photons per pixel, as well as their fluctuations, can be measured. Essentially, the quantum shot noise is directly converted into numbers at the output of the ADC. The passage from quantum randomness to an actual random number is straightforward and by no mean affected by other unaccounted (and possibly contriving) physical processes that could increase predictability and thwart security.

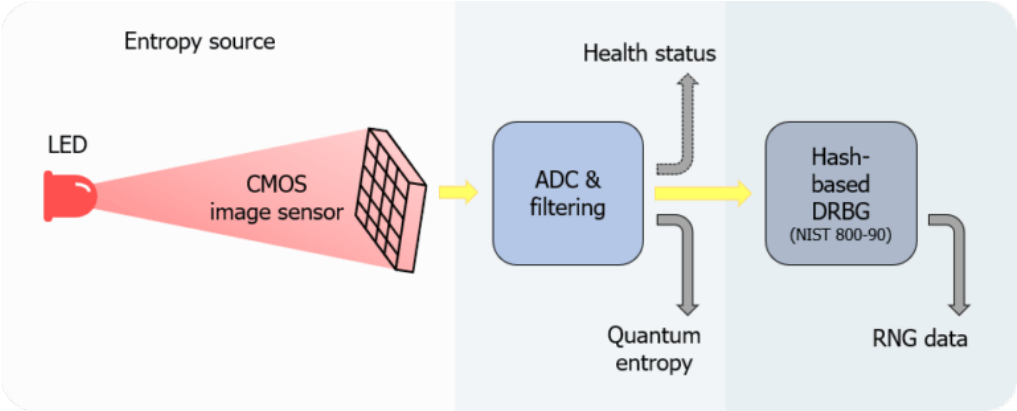


Figure 2 - Structure of IDQ's QRNG chips.

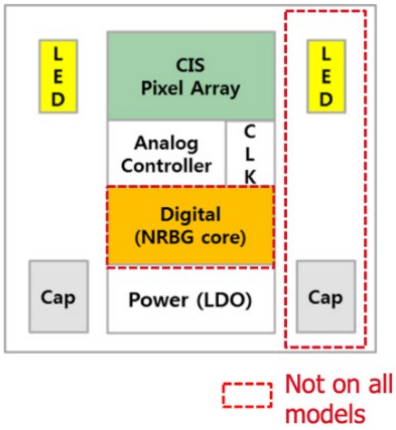


Figure 3 - Internal structure of the QRNG chips. The NRBG core implements the DRBG mechanism.

Note that the principle of operation and internal structure as described in *Figure 3* is the same for all IDQ QRNG chips, but each IDQ's QRNG chip has an implementation that slightly differ from one another. For example, each IDQ QRNG chip model has a specific CIS resolution (number pixels), number of LEDs used, and resolution of the ADC. Moreover, the hash based DRBG is embedded only in IDQ6MC1 and IDQ20MC1 chips.

In practice, an auto-calibration function controls the optical power supplied by the LED as well as the exposure time of the CIS. This sets the average of the ADC output in a good range to ensure highest entropy. It is in particular important that all pixels are not underexposed or saturated, to avoid pixel outputs become predictable. Environmental and operating conditions fluctuations (e.g. temperature, voltage or current), can affect the optical power. Fortunately, the auto-calibration of the power keeps the entropy maximal even in their presence. A health check function inside IDQ QRNG chips is always monitoring the brightness level. If it is too high or too low beyond the max and min thresholds, then the health check function makes a notice of this event to internal functions and to external user, the auto-calibration will start to re-calibrate for setting the brightness level again into the normal range. Note that if there is any failure on operation of analogue components like LED and CIS, beside the brightness, the health check function will also notice the failure to users.

In addition, and importantly, the digitized output of each pixel is sampled to keep only two bits, for example, the 3rd and the 4th bits out of the 10-bit ADC, in case of IDQ6MC1 and IDQ20MC1. The set of sample alphabets consists of 00 (or "0"), 01 (or "1"), 10 (or "2"), 11 (or "3"). This process, as well as the randomness extraction (deterministic random bit generation, DRBG) performed after the sampling, is represented on *Figure 4*. The whole process is compliant to the NIST SP 800-90A/B/C recommendations.

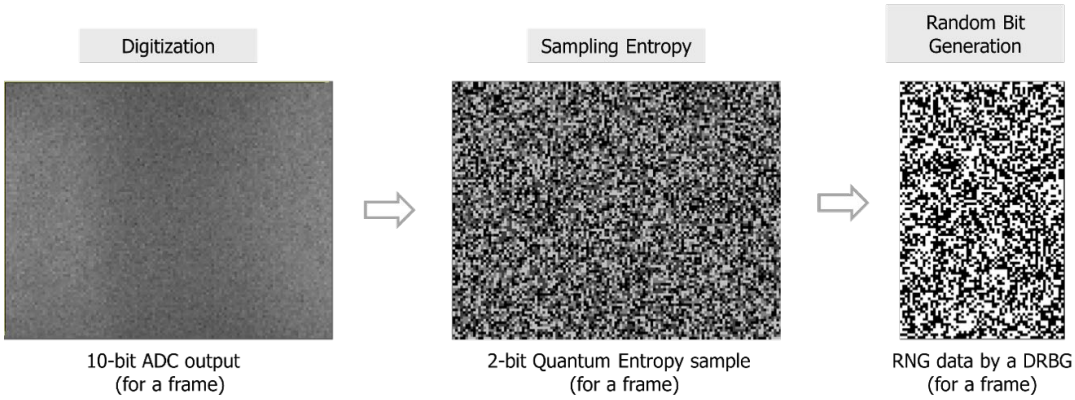


Figure 4 - Visual representation of a digitized frame captured on the CIS, to the entropy bits generated by sampling the ADC output, then to the random bit (RNG data) produced after the randomness extractor (DRBG) that is onboard of the chips. The randomness extractor reduces the number of bits generated, but it also provides numbers that have an added layer of cryptographic security that can be necessary for certification. The whole process is designed based on the NIST SP800-90C's Enhanced NRBG – Oversampling Construction.

The analysis of the entropy generated from this process will depend on the mean number of photons detected. Thanks to the robust design and the sampling, the entropy of the two-bit strings produced does not depend on the optical power very much. *Figure 5* shows that in the ideal case, if the mean value of each pixel's ADC output remains around between ~100 to ~900, the 2-bit entropy sample from each pixel can have the maximal min-entropy value, $H_{min} = -log_2 P_{max} = 2$. Note that the min-entropy is a good entropy measure for cryptographical use because it gives lower (i.e. more conservative) estimation than other entropy measures such as the Shannon entropy, the Collision entropy and so on.

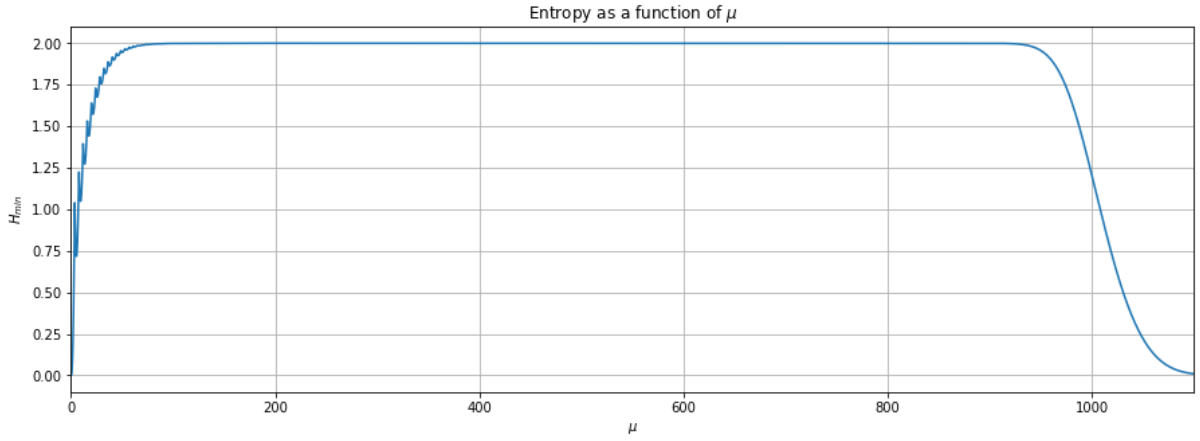
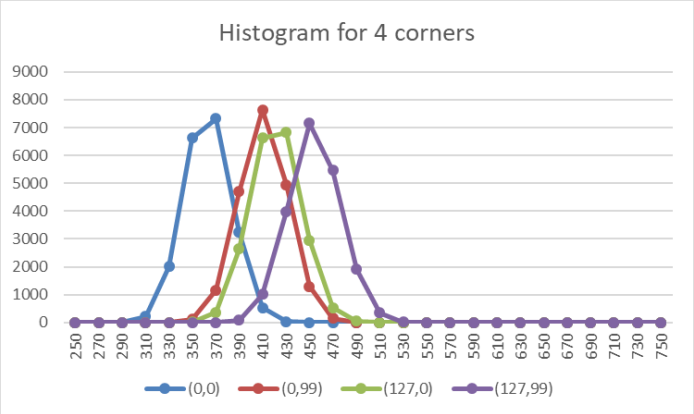


Figure 5 - Minimum entropy (H_min) of each 2-bit sequence generated by a pixel, as a function of the digitized mean number of photons captured by the pixel. Even though the mean value of each pixel is different due to various reasons, each pixel's 2-bit sample sequence equally has the maximum min-entropy.

3. Entropy estimation for QRNG chips

Figure *6* shows the probabilities of each 2-bit sequence obtained from a QRNG chip (IDQ6MC1) containing a CIS pixel array of 128x100 resolution. Considering the geometrical positions of pixels and of the light source in the QRNG chip, 4 corner pixels at (x,y) coordinates (0,0), (0,99), (127,0) and (127,99), are chosen to gather raw digitized datasets. These are good candidates to see extreme statistics. In this instance, (0,0) and (127,99) pixels respectively give the lowest and highest means. In all of the four pixels, the mean and the variance values are high enough and are essentially equal (within the uncertainty of their estimation). In other words, the Poisson distribution property that "mean = variance" is satisfied, as shown on *Figure 6*. Also, there is no under or overexposure. The distribution of all other pixels is in between the extreme values shown in *Figure 6.* The conditions necessary to generate a maximal entropy are therefore satisfied. Note that the following statistics are calculated based on 20,000 pixel output values. Considering that IDQ6MC1 runs in the frame rate of 241 fps, it takes about 83 seconds to gather such number of values. However, the statistics still satisfies the Poissonian property and this implies that the optical power's variation and instability could be negligible, since they make the variance bigger.



	(0,0)	(0,99)	(127,0)	(127,99)
Mean	372.6283	420.1958	430.6277	463.6176
Variance	368.4144	406.6577	413.6829	461.0258

Figure 6 - Distribution of the digitized pixels outputs under illumination.

After the sampling is applied, the 2-bit quantum entropy samples from each of the pixels in consideration have good uniformity and entropy results shown on *Figure 7* and *Figure 8*. The occurrence probabilities of 4 2-bit sample values, 00(0), 01(1), 10(2), and 11(3), are unbiased and around 25%. The minimum entropy scores are around 1.97 which nears the theoretical limitation per 2-bit sample. Note that the min-entropy of 1.97 is calculated based on 20,000 samples only. When we compute the entropy using a larger dataset, for example, 10 million samples or more, the entropy score gets much closer to 2.

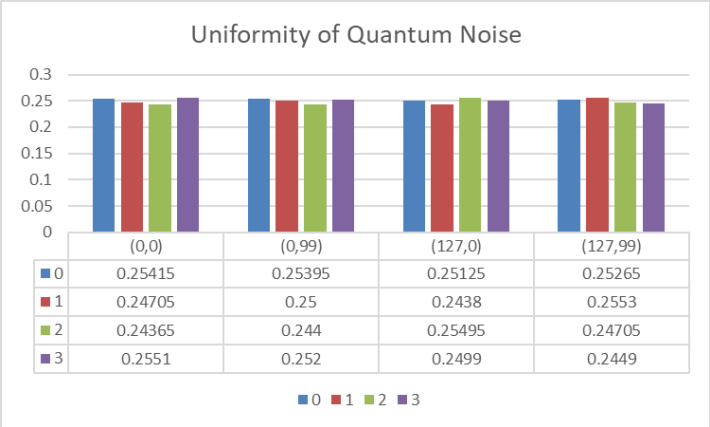


Figure 7 - Probability to obtain each 2-bit sequence for each of the 4 pixels in consideration.

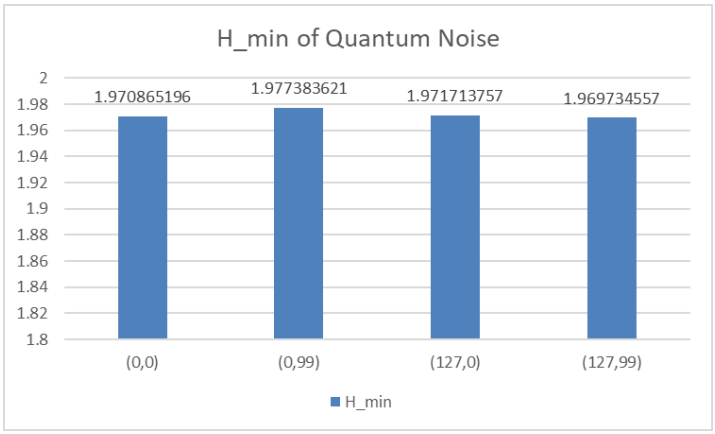
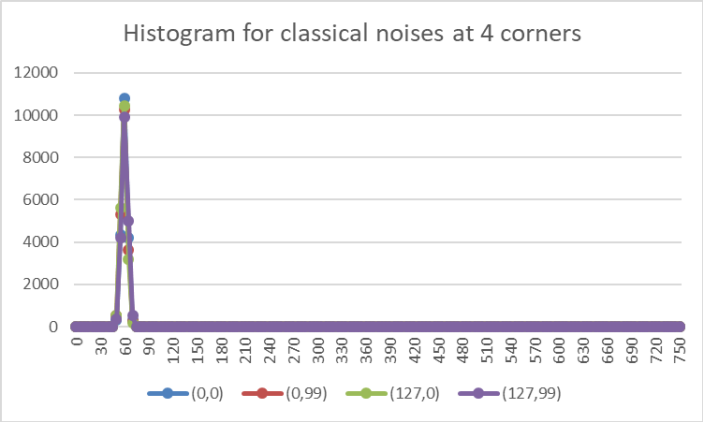


Figure 8 - Direct estimation of the min-entropy (H_min) of each of the 4 pixels in consideration.

Figure 9 shows how much classical noise can be added in the pixel outputs, which includes all classical noises from the electronics. In order to measure them, we turned off the LED, disabled the black level compensation (BLC) function of CIS and minimized the offset level of the ADC. It is usually hard to see the classical noises, because the values of classical noises mostly are 0s and rarely 1, 2 or 3 in normal operation. The variance of them remains around 12 to 14, that is, the standard deviation

is under 4. It is unlikely that classical noises make random transitions at the 3rd and 4th bits. To minimize the effect of classical noise and make quantum noise dominant, only the 3rd and 4th bits are used as they are not impacted by the fluctuation of classical noise.



	(0,0)	(0,99)	(127,0)	(127,99)
Mean	61.979	61.4432	61.2043	62.2857
Variance	11.99476	12.89997	12.11866	14.07518

Figure 9 - Distribution of the digitized pixel outputs without illumination

Note that if necessary, it is also possible to significantly reduce the above classical variance by enlarging the input voltage range of the ADC, while making the ADC less sensitive to small voltage difference. Of course, this action will reduce the quantum variance by light source as well. However, as shown in *Figure 6*, the quantum variance is sufficiently big so that we can maintain the entropy of the 3rd and 4th bits as they are by adjusting the input voltage range at an adequate level. Of course, the auto-calibration will also make the quantum variance, that is, the quantum entropy bigger again.

4. Test Results

4.1. NIST SP 800-90B IID test result for quantum noise datasets

This NIST IID entropy test runs the permutation test, the Chi-square (independence and goodness-of-fit) test and the longest repeated sequence test, in order to check if the entropy source really generates independent and identically distributed sequences. By design, all the tests respectively

have some normal failure rates of around 2.25%, 0.2% and 0.1%. As shown in the following table, IDQ QRNG chipsets pass the IID tests with normal failures.

Entropy source	Q/T	Conditioning	Permutation Fails	Chi-square Fails	LRS Fails
IDQ250C2	Q	No	4/175 (2.28%)	0/175 (0.00%)	0
IDQ6MC1	Q	No	53/2400 (2.22%)	18/2400 (0.75%)	0
IDQ20MC1	Q	No	15/576 (2.60%)	0/576 (0.00%)	0

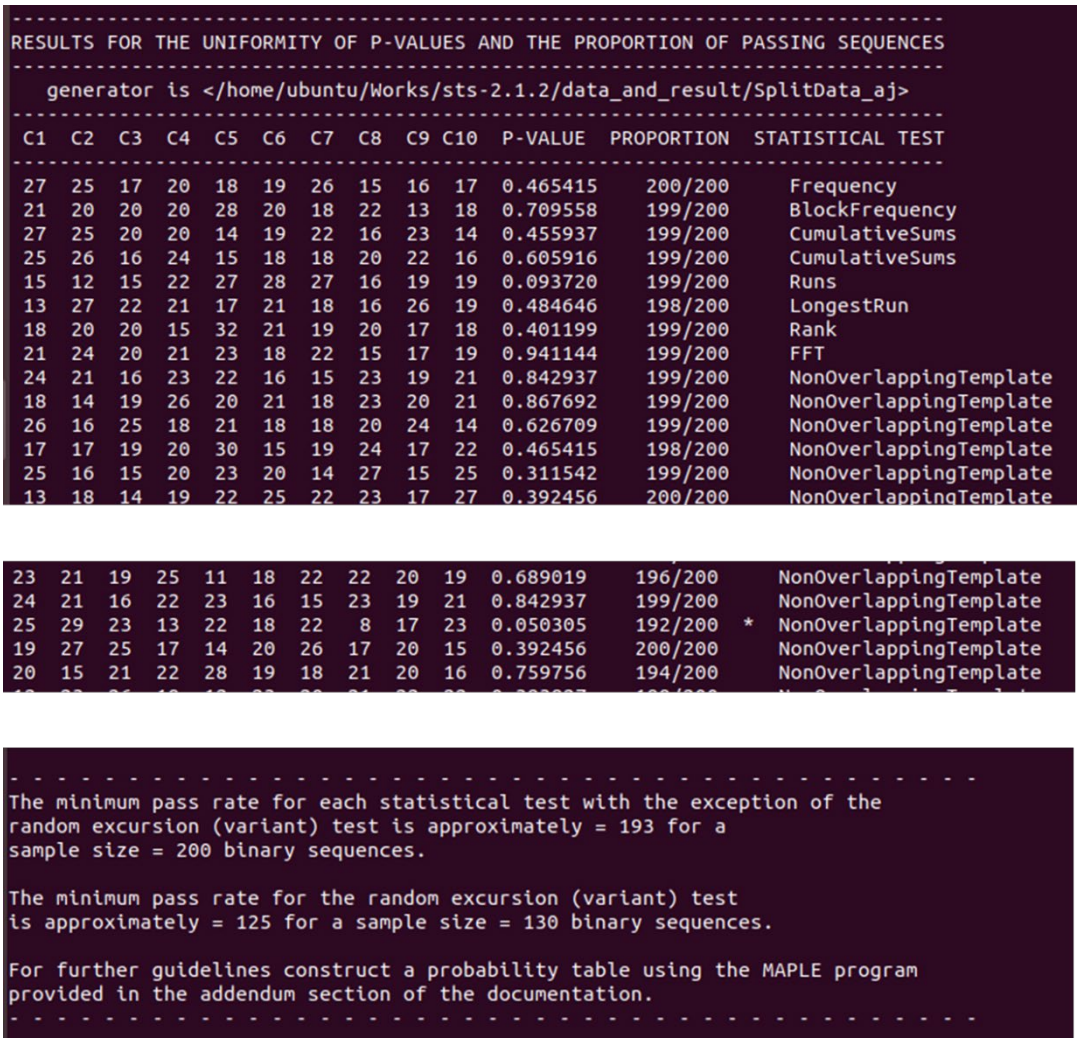
4.2. NIST SP 800-90B non-IID test result for quantum noise datasets

The NIST non-IID test is for estimating the min-entropy in a conservative way, without applying any statistical assumption on the entropy source outputs. In the test suite, 10 estimators provide the estimated min-entropy per sample with 99% confidentiality and finally the minimum of the 10 scores is picked up. Note that to get a reliable test result, the size of datasets should be at least 10MB.

Entropy source	Q/T/P	Conditioning	# of tests	Non-IID Entropy (per byte)
IDQ250C2	Q	No	175	7.4648
IDQ6MC1	Q	No	52	7.4135
IDQ20MC1	Q	No	1000	7.4721

4.3. NIST SP 800-22 sts-2.1.2 randomness test result for an RNG dataset from an IDQ6MC1 chip

IDQ6MC1 chips are using the NIST approved DRBG mechanisms and thus the RNG output datasets always have good randomness and pass all randomness test. In default setting, the test suite consists of 188 testers (including sub-testers) which have small failure probability according to the significance level. However, since there are many testers, we can see often one or two failures on the final test report with probability of around 30%. This failure rate looks high but is expected from a properly functioning RNG. The following is a part of the test result.



```
------------------------------------------------------------------------------
RESULTS FOR THE UNIFORMITY OF P-VALUES AND THE PROPORTION OF PASSING SEQUENCES
------------------------------------------------------------------------------
   generator is </home/ubuntu/Works/sts-2.1.2/data_and_result/SplitData_aj>
------------------------------------------------------------------------------
 C1  C2  C3  C4  C5  C6  C7  C8  C9 C10  P-VALUE  PROPORTION  STATISTICAL TEST
------------------------------------------------------------------------------
 27  25  17  20  18  19  26  15  16  17  0.465415    200/200     Frequency
 21  20  20  20  28  20  18  22  13  18  0.709558    199/200     BlockFrequency
 27  25  20  20  14  19  22  16  23  14  0.455937    199/200     CumulativeSums
 25  26  16  24  15  18  18  20  22  16  0.605916    199/200     CumulativeSums
 15  12  15  22  27  28  27  16  19  19  0.093720    199/200     Runs
 13  27  22  21  17  21  18  16  26  19  0.484646    198/200     LongestRun
 18  20  20  15  32  21  19  20  17  18  0.401199    199/200     Rank
 21  24  20  21  23  18  22  15  17  19  0.941144    199/200     FFT
 24  21  16  23  22  16  15  23  19  21  0.842937    199/200     NonOverlappingTemplate
 18  14  19  26  20  21  18  23  20  21  0.867692    199/200     NonOverlappingTemplate
 26  16  25  18  21  18  18  20  24  14  0.626709    199/200     NonOverlappingTemplate
 17  17  19  20  30  15  19  24  17  22  0.465415    198/200     NonOverlappingTemplate
 25  16  15  20  23  20  14  27  15  25  0.311542    199/200     NonOverlappingTemplate
 13  18  14  19  22  25  22  23  17  27  0.392456    200/200     NonOverlappingTemplate
```



```
 23  21  19  25  11  18  22  22  20  19  0.689019    196/200     NonOverlappingTemplate
 24  21  16  22  23  16  15  23  19  21  0.842937    199/200     NonOverlappingTemplate
 25  29  23  13  22  18  22   8  17  23  0.050305    192/200  *  NonOverlappingTemplate
 19  27  25  17  14  20  26  17  20  15  0.392456    200/200     NonOverlappingTemplate
 20  15  21  22  28  19  18  21  20  16  0.759756    194/200     NonOverlappingTemplate
```



```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
The minimum pass rate for each statistical test with the exception of the
random excursion (variant) test is approximately = 193 for a
sample size = 200 binary sequences.

The minimum pass rate for the random excursion (variant) test
is approximately = 125 for a sample size = 130 binary sequences.

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

4.4. Dieharder v.3.31.1 randomness test result for an RNG dataset from an IDQ6MC1 chip

The Dieharder test requires to run with a random bit file of at least 12GB to avoid the rewinding issue. It provides three types of results, pass, fail and weak according to the p-value. When a certain tester gets ‘weak’ result, it is possible to run it again until it gets either ‘pass’ or ‘fail’ result. The following is a part of result.

```
#=============================================================================#
#            dieharder version 3.31.1 Copyright 2003 Robert G. Brown          #
#=============================================================================#
   rng_name    |           filename             |rands/second|
 file_input_raw|            S2Q400-RNG-10GB.bin|  6.04e+07  |
#=============================================================================#
        test_name   |ntup| tsamples |psamples|  p-value |Assessment
#=============================================================================#
   diehard_birthdays|   0|       100|     100|0.84201435|  PASSED
      diehard_operm5|   0|   1000000|     100|0.25297365|  PASSED
  diehard_rank_32x32|   0|     40000|     100|0.85299503|  PASSED
    diehard_rank_6x8|   0|    100000|     100|0.79093749|  PASSED
   diehard_bitstream|   0|   2097152|     100|0.15476560|  PASSED
        diehard_opso|   0|   2097152|     100|0.76642259|  PASSED
        diehard_oqso|   0|   2097152|     100|0.70893143|  PASSED
         diehard_dna|   0|   2097152|     100|0.99940527|   WEAK
         diehard_dna|   0|   2097152|     200|0.75671137|  PASSED
diehard_count_1s_str|   0|    256000|     100|0.69194048|  PASSED
diehard_count_1s_byt|   0|    256000|     100|0.33650383|  PASSED
 diehard_parking_lot|   0|     12000|     100|0.64976124|  PASSED
    diehard_2dsphere|   2|      8000|     100|0.85608105|  PASSED
    diehard_3dsphere|   3|      4000|     100|0.20895372|  PASSED
     diehard_squeeze|   0|    100000|     100|0.10212708|  PASSED
        diehard_sums|   0|       100|     100|0.06647441|  PASSED
        diehard_runs|   0|    100000|     100|0.17470352|  PASSED
        diehard_runs|   0|    100000|     100|0.31026173|  PASSED
       diehard_craps|   0|    200000|     100|0.60040260|  PASSED
       diehard_craps|   0|    200000|     100|0.96785949|  PASSED
 marsaglia_tsang_gcd|   0|  10000000|     100|0.34696504|  PASSED
 marsaglia_tsang_gcd|   0|  10000000|     100|0.97517953|  PASSED

      rgb_lagged_sum|  27|   1000000|     100|0.73897942|  PASSED
      rgb_lagged_sum|  28|   1000000|     100|0.90274042|  PASSED
      rgb_lagged_sum|  29|   1000000|     100|0.51067478|  PASSED
      rgb_lagged_sum|  30|   1000000|     100|0.28031247|  PASSED
      rgb_lagged_sum|  31|   1000000|     100|0.29051266|  PASSED
      rgb_lagged_sum|  32|   1000000|     100|0.74029673|  PASSED
     rgb_kstest_test|   0|     10000|    1000|0.56797391|  PASSED
     dab_bytedistrib|   0|  51200000|       1|0.25123374|  PASSED
             dab_dct| 256|     50000|       1|0.17611116|  PASSED
Preparing to run test 207.  ntuple = 0
        dab_filltree|  32|  15000000|       1|0.95819161|  PASSED
        dab_filltree|  32|  15000000|       1|0.82049568|  PASSED
Preparing to run test 208.  ntuple = 0
       dab_filltree2|   0|   5000000|       1|0.84534061|  PASSED
       dab_filltree2|   1|   5000000|       1|0.13153858|  PASSED
Preparing to run test 209.  ntuple = 0
        dab_monobit2|  12|  65000000|       1|0.59847777|  PASSED
```

```
      rgb_lagged_sum|  27|   1000000|     100|0.73897942|  PASSED
      rgb_lagged_sum|  28|   1000000|     100|0.90274042|  PASSED
      rgb_lagged_sum|  29|   1000000|     100|0.51067478|  PASSED
      rgb_lagged_sum|  30|   1000000|     100|0.28031247|  PASSED
      rgb_lagged_sum|  31|   1000000|     100|0.29051266|  PASSED
      rgb_lagged_sum|  32|   1000000|     100|0.74029673|  PASSED
     rgb_kstest_test|   0|     10000|    1000|0.56797391|  PASSED
     dab_bytedistrib|   0|  51200000|       1|0.25123374|  PASSED
         dab_dct| 256|     50000|       1|0.17611116|  PASSED
Preparing to run test 207.  ntuple = 0
        dab_filltree|  32|  15000000|       1|0.95819161|  PASSED
        dab_filltree|  32|  15000000|       1|0.82049568|  PASSED
Preparing to run test 208.  ntuple = 0
       dab_filltree2|   0|   5000000|       1|0.84534061|  PASSED
       dab_filltree2|   1|   5000000|       1|0.13153858|  PASSED
Preparing to run test 209.  ntuple = 0
        dab_monobit2|  12|  65000000|       1|0.59847777|  PASSED
```

5. Conclusion

In this document, we show how much good entropy IDQ QRNG chips can provide, based on physical and mathematical modelling. According to the simple characteristics of our quantum entropy source, the entropy quality is easy-to-understand, easy-to-control, and easy-to-maintain. IDQ QRNG chips successfully passes the IID test suite of NIST SP 800-90B entropy test suite. The min-entropy score obtained in both the IID and non-IID test suites have been shown to reach the maximal values attainable with these tests. Interestingly, these maximal values are obtained on the quantum entropy data, i.e. without using any randomness extractor (DRBG). Moreover, the random bit strings obtained after the DRBG successfully passes all the randomness tests, for example, the NIST SP800-22 sts-2.1.2 and the Dieharder v.3.31.1. The randomness extractor is the Hash-based DRBG mechanism, which is well designed to get high quality randomness, approved by the NIST SP 800-90A and operated by the oversampling-NRBG (Non-deterministic Random Bit Generator) construction of the NIST SP 800-90C.