Zurich University
of Applied Sciences

School of
Engineering

InES Institute of
Embedded Systems

Performance Evaluation of the IDQ6MC1 Quantum Random Number Generator

Abstract - IoT endpoints and systems are popular targets for cyberattacks which is why they need to be protected with strong security algorithms. The keys and tokens that are used for authentication, encryption, access control and many other aspects of modern security all rely upon strong random numbers. While modern operating systems and computers have long had secure random number generators (RNG) worked out, IoT devices still lack a strong source of randomness, also known as entropy, due to their constrained resources. This report evaluates a quantum random number generator targeted at smart devices, IoT and vehicle-to-everything (V2X) applications regarding execution time and energy consumption.

Many security-relevant operations, such as key generation, authentication, and zero-knowledge proof rely on random numbers. They are used to generate keys or tokens which are added to a message in order to make the message content unpredictable. The stronger the entropy in a random number, the harder it will be for an attacker to guess it and forge a request. A strong entropy is therefore essential. Not all random numbers are equally suited to be used for cryptographic operations. We distinguish between the ones generated by a pseudorandom number generator (PRNG) and a true random number generator (TRNG). Pseudo random number generators can only provide statistic randomness. They are determined using mathematical operations which are given an initial value, typically called seed or key. If the same initial value is used, the result of the PRNG will be the same. Pseudorandom numbers should therefore never be used for security protocols but might be used to simulate sensor data. Since all computers are deterministic machines, they can only generate pseudorandom numbers, but true random numbers are needed for secure communication. True random number generators can produce random, unpredictable numbers even if all components of the process (algorithms, initial values) are known. This characteristic is in line with Kerckhoff's principle of cryptography which says that a cryptography system should be secure, even if everything about the system is public knowledge, except for the key. Usually, this is true only for physical processes such as e.g. electronic noise, a beam splitter, or the movement of the mouse. They are therefore often referred to as hardware random number generators. Especially quantum processes have a strong entropy.

The IDQ6MC1 [1] is a true quantum random number generator chip. It targets smart devices, IoT and V2X applications. The chip collects true randomness from the shot noise of a light-emitting diode (LED) captured by a CMOS image sensor. This process is intrinsically and provably random. The numbers resulting from this process can then be fed to a deterministic random bit generator (DRBG) to produce random bits in compliance with the NIST SP800-90A/B/C [2], [3], [4] recommendations for secure DRBGs (A), secure entropy sources (B) and random bit generator (RBG) construction (C). In addition, the manufacturer claims that it passes IID, non-IID, DieHarder, and NIST SP800-22 [5] test suites for RNGs for cryptographic applications. Moreover, the chip is AEC-Q100 [6] certified, which states the chip's reliability when being embedded in any security system of a connected car securing in-vehicle and V2X communications. Communicating either over I2C or SPI, the chip can be integrated into any system. The driver, written in C for standard Linux systems, is only available upon request but can easily be adapted to run with any embedded RTOS or alternative OS.

Fig. 1 shows the two different evaluation boards available to test the chip. One is the QRNG EVK based on the Raspberry Pi platform and the second one features both the IDQ6MC1 and IDQ250C2 chip with the Raspberry Pi pin header as well. For the evaluation, we used the QRNG EVK.

Fig. 1: QRNG EVK (left) and IDQ6MC1 & IDQ250C2 evaluation board (right)

In a previous study [7], we evaluated the performance of cryptographic operations of different secure elements, including the generation of random numbers. Following up on this work, we evaluate the performance of the IDQ6MC1. Although the IDQ6MC1 is not a secure element, we want to make the evaluation comparable to the secure elements as well as to a software implementation. In order to achieve that, we chose the same basic hardware, application and measurement setup. The application was thus implemented on an nRF52840 development kit from Nordic using the Zephyr RTOS. For the software implementation, we use the crypto library MbedTLS. A more detailed description of the implementation and test setup used can be found in [7]. The application initialises the IDQ6MC1 or the MbedTLS context, respectively, and then reads a given number of random bytes. We measure three different sizes, 32, 256 and 1024 bytes. To measure the execution time, we use a GPIO signal which is set high when the operation starts and set low when it finishes. As mentioned earlier, software random number generators such as MbedTLS use a seed to generate a random number, where only the seed is truly random. By default, MbedTLS reseeds its DRBG after every 10’000 calls. To measure the TRNG from the nRF52840, we are forcing MbedTLS to reseed its DRBG on each call. Hence, the reseeding of the DRBG is included in the time and energy measurements.

Fig. 2 shows the measurement setup. The nRF52840 DK is supplied with 3.3 V by the Keysight N6705B power analyser on channel 1. Channel 2 measures the GPIO that indicates the execution time, and channel 3 supplies the IDQ6MC1 with 2.8 V. The overall energy consumption is calculated by adding the energy supplied by channels 1 and 3.

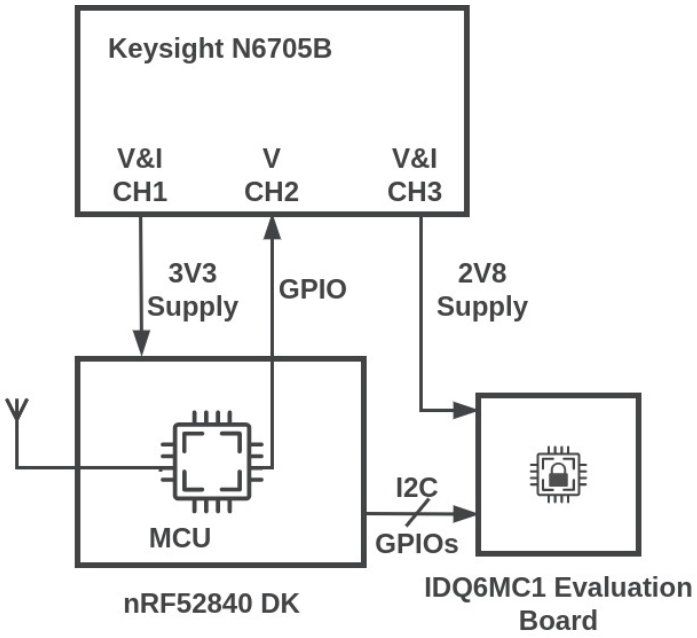

Fig. 2: Measurement setup of the node for acquiring the execution time and energy consumption during the random number generation

Tab. 1 shows the execution time and energy consumption of the complete node for 32, 256 and 1024 bytes. The values in the table show the median of a sample rate n=100. The column “MbedTLS” shows the absolute values of the reference implementation and the “IDQ” column shows the absolute values of the MCU and the IDQ6MC1. Both implementations scale linearly, but execution time and energy consumption increase much faster for the IDQ6MC1. This has mainly two reasons. First, the IDQ6MC1 is externally connected via I2C, and with a maximum packet size of 16 bytes, for a 1024 bytes-size random number, 64 read cycles have to be executed, which contributes in large part to the execution time and energy consumption. Second, although MbedTLS is forced to reseed its DRBG, the seed is always 32 bytes long whether the requested random number is 32, 256 or 1024 bytes long. This seed length is the default configuration for MbedTLS. The maximum length of the seed is 64 bytes. Tab. 2 shows the absolute values from the IDQ6MC1 exclusively (channel 3).

	MbedTLS		**IDQ**	
	Time [s]	**Energy [J]**	**Time [s]**	**Energy [J]**
32 bytes	5.73×10^{-3}	6.32×10^{-5}	6.55×10^{-3}	3.82×10^{-4}
256 bytes	6.39×10^{-3}	7.08×10^{-5}	5.28×10^{-2}	3.11×10^{-3}
1024 bytes	8.68×10^{-3}	9.73×10^{-5}	2.13×10^{-1}	1.26×10^{-2}

Tab. 1: Execution time and energy consumption of the node

	Energy [J]
32 bytes	3.19×10^{-4}
256 bytes	2.58×10^{-3}
1024 bytes	1.04×10^{-2}

Tab. 2: Absolute values of the energy consumption for the IDQ6MC1 without the MCU

Fig. 3 and Tab. 3 show the IDQ6MC1 in comparison to the previously evaluated secure elements. In this test case, 32 bytes of random data were generated. Instead of absolute values, the values in Tab. 3 show by how much the execution time and the energy consumption has increased or decreased compared to the software implementation with MbedTLS. 100% corresponds to the absolute values of MbedTLS in Tab. 1.

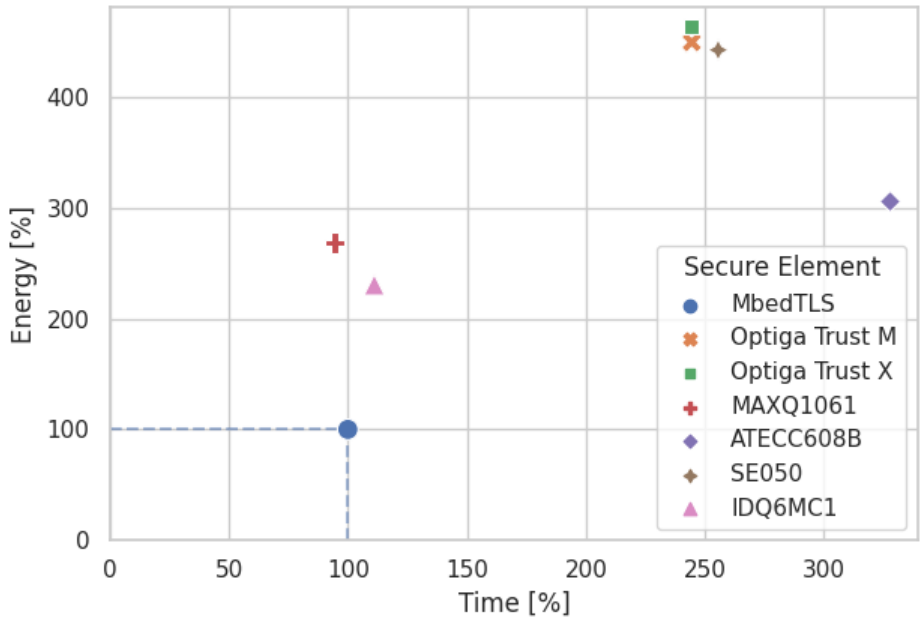

Fig. 3: Time and energy measured for getting 32 bytes of random data relative to the reference implementation. (median of values, rounded to the nearest percent, n = 100)

	Time [%]	Energy [%]
MbedTLS	100	100
Optiga Trust M	244	449
Optiga Trust X	244	463
MAXQ1061	94	268
ATECC608B	328	306
SE050	256	442
IDQ6MC1	111	230

Tab. 3: Table of values for Fig. 3

The IDQ6MC1 is the second fastest and has the lowest energy consumption out of all the tested devices, with only 111% of the time and 230% of the energy consumption compared to the software implementation, only the MAXQ1061 is faster. When larger numbers are requested, the IDQ6MC1 does not perform as well because of the maximum packet size of 16 bytes for the I2C communication. Although in practice, an entropy of 32 or 64 bytes is often enough so that this shortcoming can be neglected. In the scope of this work, we only

evaluated the performance via I2C to make the measurements comparable to our previous work. However, the IDQ6MC1 also offers an SPI interface with a data rate of 24 MHz which would speed up the communication considerably but might change the energy consumption as well. On top of that, the chip is compliant with NIST SP800-90A, B, and C, and AEC-Q100 certified and passes several other test suites. In contrast, the SE050 and ATECC608B are the only secure elements that are compliant with NIST800-90A/B. Therefore, the use of this quantum random number generator is more than justified in any application where a strong entropy source is needed.

References

[1] ID Quantique, “Quantis QRNG Chip,” 11 08 2022. [Online]. Available: https://marketing.idquantique.com/acton/attachment/11868/f-025e/1/-/-/-/-/Quantis%20QRNG%20Chip_Brochure.pdf.

[2] E. Barker und J. Kelsey, «NIST Special Publication 800-90A, Recommendation for Random Number Generation Using Deterministic Random Bit Generators,» National Institute of Standards and Technology, NIST, 2015.

[3] M. S. Turan, E. Barker, J. Kelsey, K. A. McKay, M. L. Baish and M. Boyle, “NIST Special Publication 800-90B, Recommendation for the Entropy Sources Used for Random Bit Generation,” National Institute of Standards and Technology, NIST, 2018.

[4] E. Barker und J. Kelsey, «(Second Draft) NIST Special Publication 800-90C, Recommendation for Random Bit Generator (RBG) Constructions,» National Institute of Standards and Technology, NIST, 2016.

[5] A. Rukhin, J. Soto, J. Nechvatal, S. Miles, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Bankds, A. Heckert, J. Dray und S. Vo, «Special Publication 800-22, A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications,» National Institute of Standards and Technology, NIST, 2010.

[6] Automotive Electronics Council, “AEC - Q100 - Rev-G, FAILURE MECHANISM BASED STRESS TEST QUALIFICATION FOR INTEGRATED CIRCUITS,” 05 2007. [Online]. Available: http://www.aecouncil.com/Documents/AEC_Q100_Rev_G_Base_Document.pdf. [Accessed 08 2022].

[7] M. Noseda, L. Zimmerli, T. Schläpfer und A. Rüst, «Performance Analysis of Secure Elements for IoT,» *IoT,* 3 2022.

Zurich University
of Applied Sciences

School of Engineering

Institute of Embedded Systems (InES)

Lea Zimmerli, Andreas Rüst
Technikumstrasse 9
CH-8400 Winterthur

lea.zimmerli@zhaw.ch
andreas.ruest@zhaw.ch
www.zhaw.ch/ines