OPENQRNG

QT1

Operating manual v1.0

contact@openqrng.io

Working principle

The QT1 is a quantum tunneling-based random number generator. Quantum Tunneling is a quantum-mechanical phenomenon where a particle tunnels through a barrier that it classically could not surmount. It is the operative mechanism in tunnel diodes.

The design of the QT1 is inspired by two papers[1] [2] describing how randomness can be extracted from quantum tunneling effect.

To achieve this, we exploit a phenomenon of tunnel diode happening at the peak current (where the tunneling effect is maximum) of the device. As shown in the papers, sweeping current across the device results in a voltage hysteresis cycle. The current value where the voltage across the device switch to the high value varies randomly at each sweep cycle.

The purpose of the QT1 design is to sweep current across a tunnel diode, detect at which current value the voltage across the device switch to the high value, and extract randomness from the distribution of these current values. This working principle together with conception information will be detailed in the next sections.

Technical specifications

- 1Mbps rate
- USB serial interface
- USB powered
- Compatible with any operating system supporting serial interface
- NIST-SP800-22 test suite compliant

[1] Bernardo-Gavito, R. et al. Extracting random numbers from quantum tunnelling through a single diode. Sci. Rep. 7(1), 1–6 (2017). https://www.nature.com/articles/s41598-017-18161-9

[2] K. Aungskunsiri, R. Amarit, and K. Wongpanya, “Random number generation from a quantum tunneling diode,” Appl. Phys. Lett. 119, 074002 (2021). https://doi.org/10.1063/5.0055955

Operating description

Block diagram

The block diagram of the QT1 is as follow:

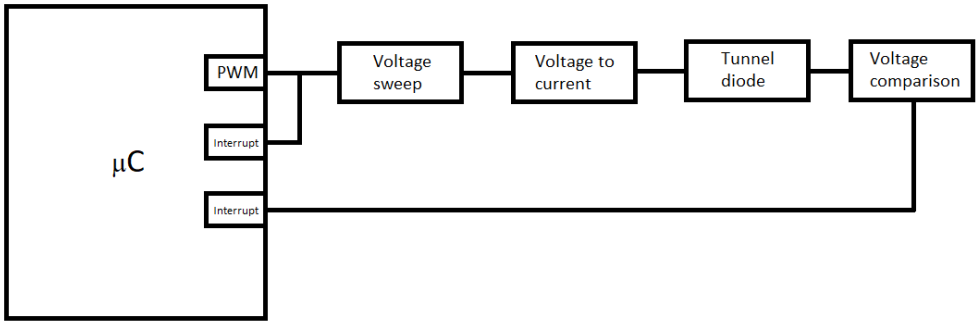

Fig. 1: QT1 block diagram

Voltage sweeping

The voltage sweep curve is obtained from an RC charging/discharging circuit. By inputting a 1MHz PWM signal to the RC circuit, the capacitor will charge and discharge according to the PWM level. To work at a 1MHz frequency, each charging/discharging cycle must be 1µs, so a charging or discharging cycle must be 500ns. 500ns is the 5*RC time constant, the RC time constant of about 100ns. Finally, we set a second order RC circuit with R=1kΩ, and C=125pF to have a cleaner curve.

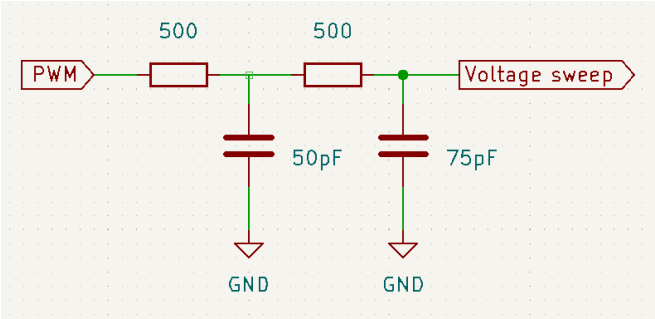

Fig. 2: voltage sweeping circuit

This block function does not depend on the chosen tunnel diode.

Voltage to current and tunnel diode

To convert the voltage sweep to current sweep, we simply use a voltage to current circuit. The tunnel diode act as a load in the circuit.

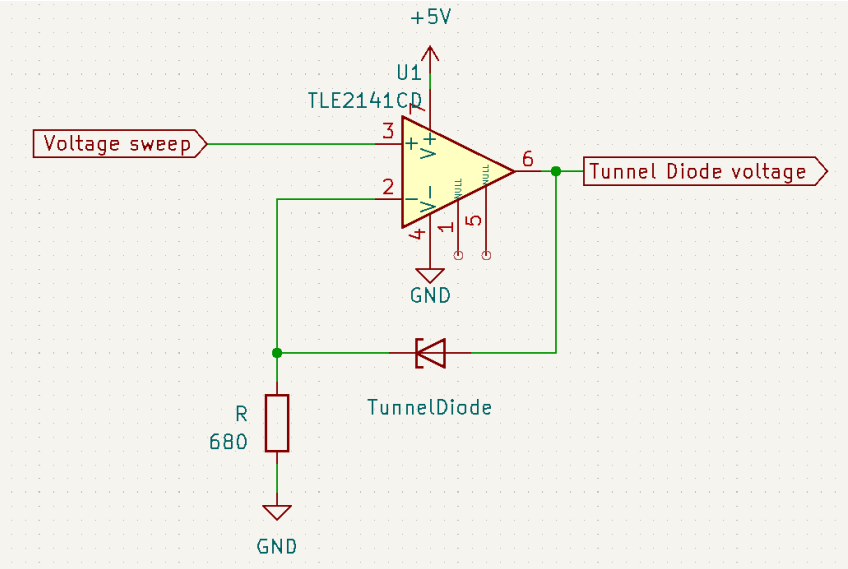

Fig. 3: voltage to current and tunnel diode

The resistor R value depends on the chosen tunnel diode. The resistor value should be chosen so that the peak current (Ip) should be reach at about 3.25V using R=U/I relation, corresponding to approx. 4*RC time constant, in order to work in the quasi-linear part of the RC charging curve.

The datasheet of the chosen op amp. TLE2141 shows an output swing of V_{cc-} + 0.1V to V_{cc+} - 1V. As the op amp. is powered by 0-5V, the minimum voltage on its input cannot go lower than 0.1V. It means that the voltage sweep should not go lower than 0.1V, to do so, the duty cycle of the PWM signal is set to 70% to not discharge the capacitor completely.

Voltage comparison

To detect the tunnel diode switch to high voltage value, a fast comparator is used. The comparator will output a logical ‘1’ when the input voltage is higher than a reference voltage. More especially, we previously set the switching voltage to 3.25V, so the comparator should detect when the voltage exceeds 3.25V plus a threshold defined by the mean between the low and high voltage of the tunnel diode. These values depend on the chosen tunnel diode. For the QT1, 3.7V is a good level to detect.

To generate a stable 3.7V reference, it is not possible to use the 5V input of the USB input as it fluctuates depending on the USB host, which could add undesired offset to the voltage reference. To avoid this, an op amp. is used to amplify the stable 3.3V generated by the microcontroller. It is achieved by the following circuit:

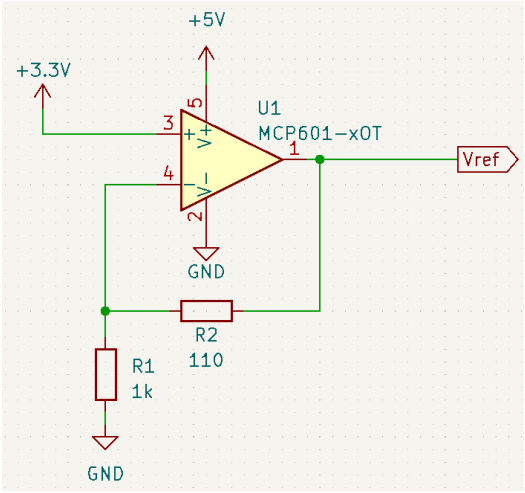

Fig. 4: Vref generation

Now we have all ingredients to detect the tunnel diode switch to its high voltage value using the comparator circuit:

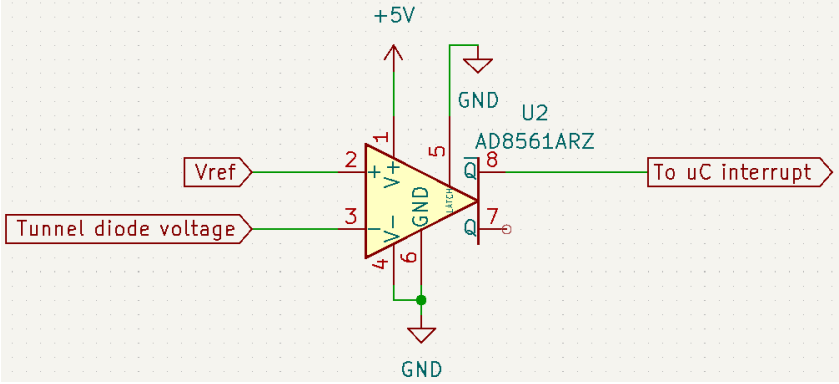

Fig. 5: comparator circuit

Randomness extraction

As we mention previously, the information used to extract randomness is the current value where the tunnel diode switch to its high voltage, in fact, the current flowing through the tunnel diode is proportional to the voltage sweep, and because we are working in the quasi-linear region of the voltage sweep, the elapsing time is also proportional to the voltage variation. So finally, to get a relevant information on the current where the tunnel diode switch to its high voltage, we measure the time elapsed between the start of the PWM signal, and the detection of the comparator going from logical '0' to '1'. To achieve this, we use interrupt mechanism of the microcontroller. Interrupts allow to perform specific task when a trigger condition occurs. In our case, we set a first interrupt which store the clock cycle count of the microcontroller when the PWM cycle starts, and a second interrupt raising when the output of the comparator going from logical '0' to '1', which counts the elapsed clock cycle between these two events. The chosen microcontroller, a Teensy 4.0, has a sufficient resolution as it operates at 600MHz.

The distribution of cycle count between PWM start and tunnel diode high voltage switch detection should give something like:

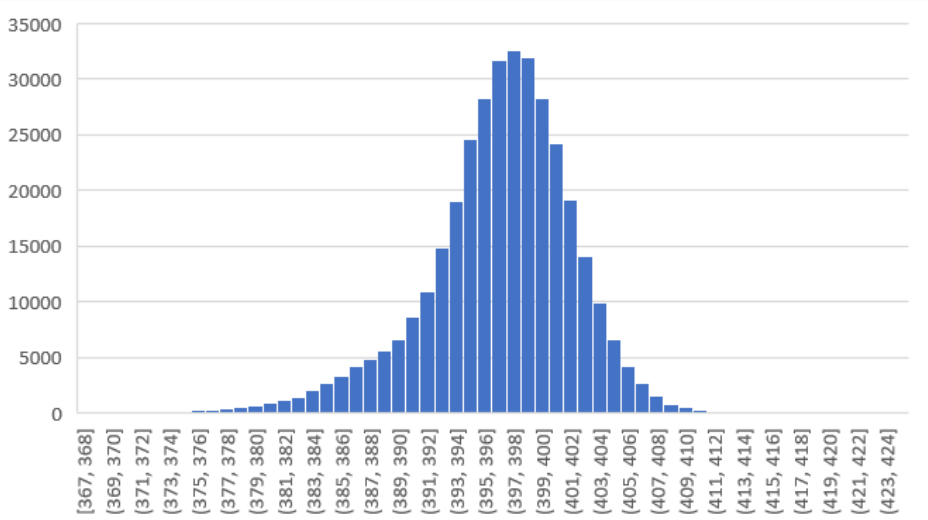

Fig. 6: cycle count distribution for 350000 measurements.

Finally, to extract a random bit from the cycle count, we simply take the parity of the value.

LICENSE INFORMATION

Hardware

All hardware is licensed under the CERN-OHL-S v2.

Copyright OpenQRNG 2023.

This source describes Open Hardware and is licensed under the CERN-OHL-S v2.

You may redistribute and modify this documentation and make products using it under the terms of the CERN-OHL-S v2. This documentation is distributed WITHOUT ANY EXPRESS OR IMPLIED WARRANTY, INCLUDING OF MERCHANTABILITY, SATISFACTORY QUALITY AND FITNESS FOR A PARTICULAR PURPOSE. Please see the CERN-OHL-S v2 for applicable conditions

Software

Copyright (C) 2023 OpenQRNG

All software is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this source. If not, see https://www.gnu.org/licenses/.