Admin Manual
Project Semester 2023

Carlo Bauer
Paul Betsch
Marina Göppel
Stefan Kleinhenz Leiva
Carsten Michel
Lukas Siegle

June 14, 2023

Contents

1 Intended Use

The intended use of this Pendulum is to serve as a True Random Bit Generator. A True Random Bit Generator employs a physical noise source to generate genuine random bits. The API provides the capability to control the desired number of bits and the quantity in which they should be generated. These random bits can subsequently be utilized for diverse applications, including cryptography, simulation, and scientific research.

This prototype of a True Random Bit Generator (TRNG) was conceived and developed within the framework of our project semester during the summer of 2023 at the University of Applied Sciences Mannheim.

1.1 Specifications

These specifications were provided to us by our client, Thales:

ID	Date	Requirement
1	March 22, 2023	The noise source should be as creative as possible.
2	March 22, 2023	Each team has to choose a different noise source.
3	March 22, 2023	The physical noise source has to be digitized into digital numbers.
4	March 22, 2023	The prototype has to meet all the requirements of the BSI PTG.2 standard
5	March 22, 2023	The REST-Interface should be able to handle incoming requests which contains a positive amount of randoms bits.
6	March 22, 2023	The prototype should be fully functional while being offline.
7	March 30, 2023	Specific requirements with YAML files and JSON descriptions.
8	April 6, 2023	The pendulum has to have a limited swing time to reduce similarities in his behaviour.
9	April 6, 2023	The REST-Interface has to provide a request to start and shutdown the prototype.
10	April 21, 2023	The REST-Interface has to implement a standby mode. The init request should wake up the prototype and the shutdown request should put the prototype into sleeping mode.
11	April 21, 2023	The implementation of the HTTP response status code 432 to end the communication if the prototype is not initialized. The response status code 555 to signal a timeout, if the startup procedures takes longer than 60 seconds.
12	April 28, 2023	In the standby mode any consumer, that is not related with the REST-Interface, has to consume no or significantly less electricity.
13	April 28, 2023	The module with the REST-Interface has to provide access to a LAN interface. A REST-Interface provided over WIFI is not permitted.
14	April 28, 2023	50 x 320 Bytes have to be generated until the Qualification Review on the 10th of May 2023 and tested with the NIST-Tests.
15	May 23, 2023	7.2 million bits have to be generated and send to the project supervisor of Thales until 12:00 on the 12th of June, 2023

2 System Architecture

We chose to employ UML (Unified Modeling Language) to illustrate the interactions and structure of our system's components. To achieve this, we adopted the widely recognized 4+1 view software architecture model. This model allowed us to capture different aspects of our system's architecture comprehensively.

Through UML, we utilized various diagram types, including a distribution diagram, sequence diagram, and component diagram, to depict the interactions between our system's components. These diagrams provided a clear visualization of how different elements within our system collaborate and communicate with each other.

2.1 Component diagram

The purpose of a component diagram is to show the relationship between different components in a system.

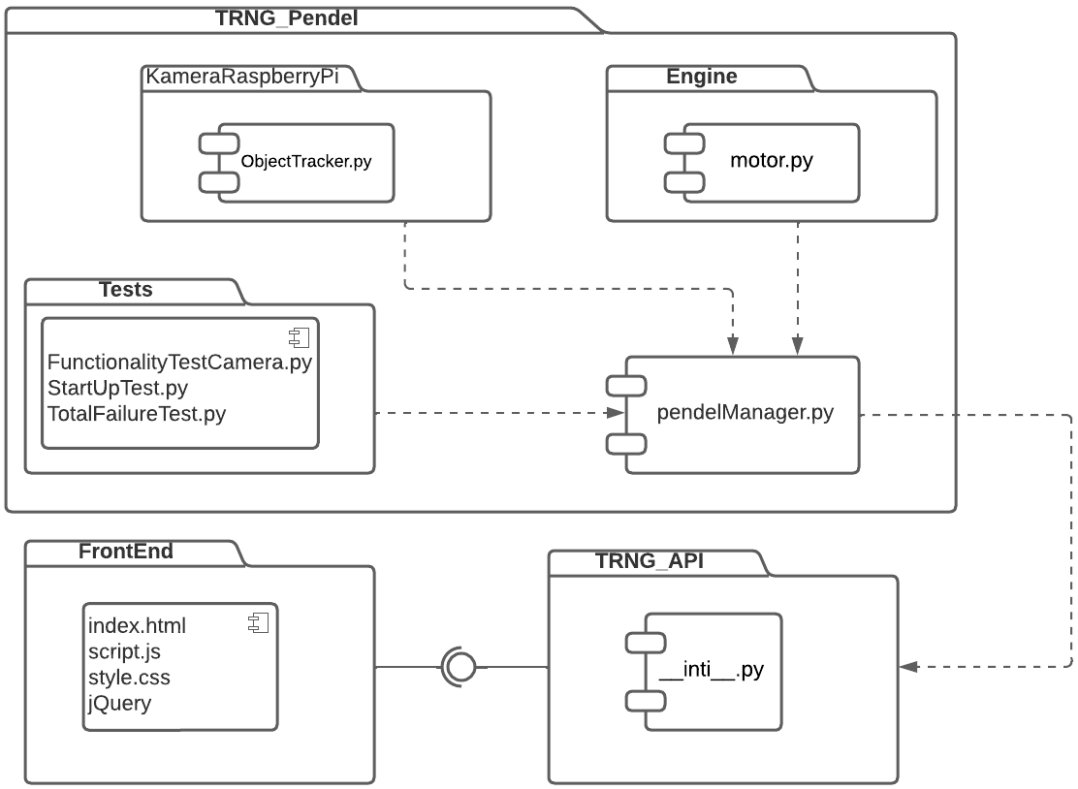

Figure 1: Component diagram

The structure of the component diagram can be outlined as follows:

- **ObjectTracker.py:** This file handles motion capturing using the OpenCV framework. It contains functions and classes related to tracking black surfaces in the video stream.
- **motor.py:** This file is responsible for controlling the relay using GPIO pins. It handles the on/off control of the motor and lifting magnet.
- **Tests:** This directory contains the tests for the project. These tests are used to verify the functionality of various components and modules.
- **pendelManager.py**: This file manages multiple processes and exposes public methods for the REST API. It coordinates the interaction between different modules and handles the logic of the pendulum system.
- **TRNG API:** The REST API provides all the necessary functionalities and logic responsible for the aforementioned requests and the initialization of the TRNG and its associated components.
- **Frontend:** In order to have an easy access and control for the REST API, we have implemented a frontend webpage.

In a component diagram in UML "lollipop" symbols are used to represent the interfaces of a component. The "lollipop" symbol consist of a circle attached at the end of a line. The line represents the component itself, while the circle represents the interface.
An interface defines the public methods and properties offered by the component.

The dependency symbol shows that one part of the system depends on another. This symbol is represented by dashed lines linking one component to another.

2.2 Deployment diagram

The deployment diagram describes the physical deployment of information generated by the software program on hardware components.

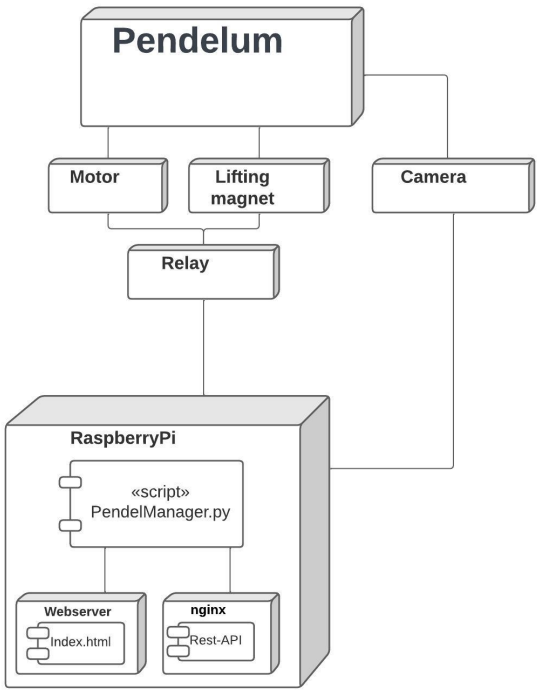

Figure 2: Deployment diagram

The deployment diagram consists of the following components:

- Raspberry Pi: The central unit that controls the Pendulum Manager and the REST API.
- Motor and lifting magnet Control: The motor and lifting magnet are controlled using a relay connected to the Raspberry Pi. The relay allows the Raspberry Pi to switch the power supply to these components on and off.
- Relay: The relay is connected to the motor and lifting magnet, providing them with voltage and acting as a switch.
- INA 219: This module measures the current at the lifting magnet. This allows us to check if the magnet is working or not.

Overall, the Raspberry Pi acts as the core of the prototype, integrating the Pendulum Manager, REST API, motor control via a relay, and camera functionality using the OpenCV framework.

2.3 Sequence diagram

A sequence diagram is a type of interaction diagram because it describes how and in what order a group of objects works together. This diagram consists of a group of objects that are represented by lifelines, and the messages they exchange over time during the interaction.

2.3.1 Init request

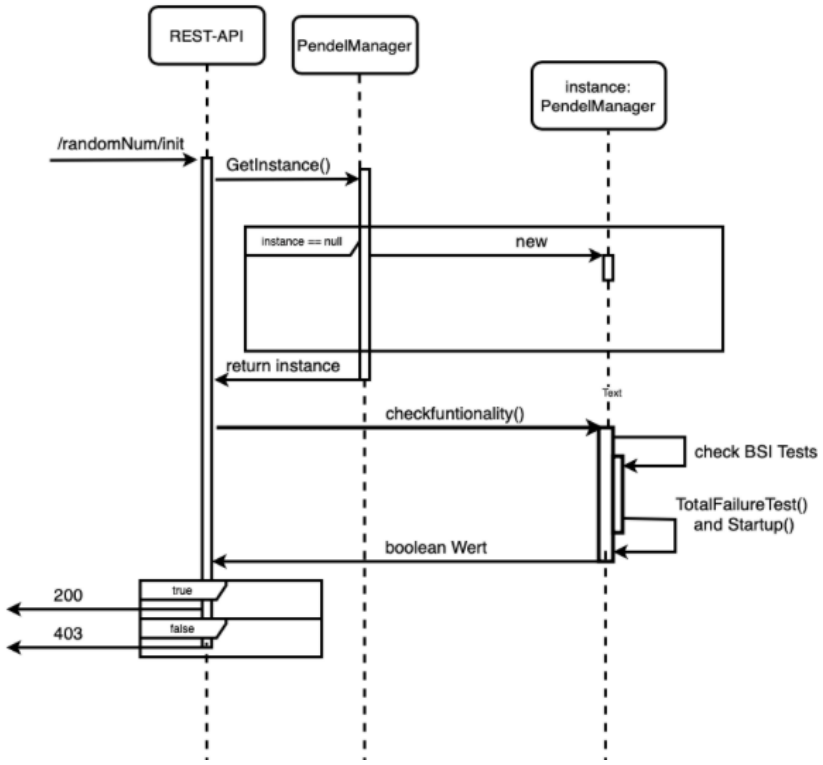

Figure 3: Sequence diagram - init request

The init request consists of the following steps:

1. We retrieve an instance of our Pendulum Manager.
2. The checkFunctionality() method is called to check if all components, such as the pendulum's motorization and the camera, are working.
3. Afterwards, the two BSI tests, namely the Total Failure Test and the Startup Test, are performed on the first 1024 bits.
4. These tests return a boolean value. If the return value is true, we receive a status of 200, and if it is false, we receive a status of 403.

2.3.2 GetRandom request

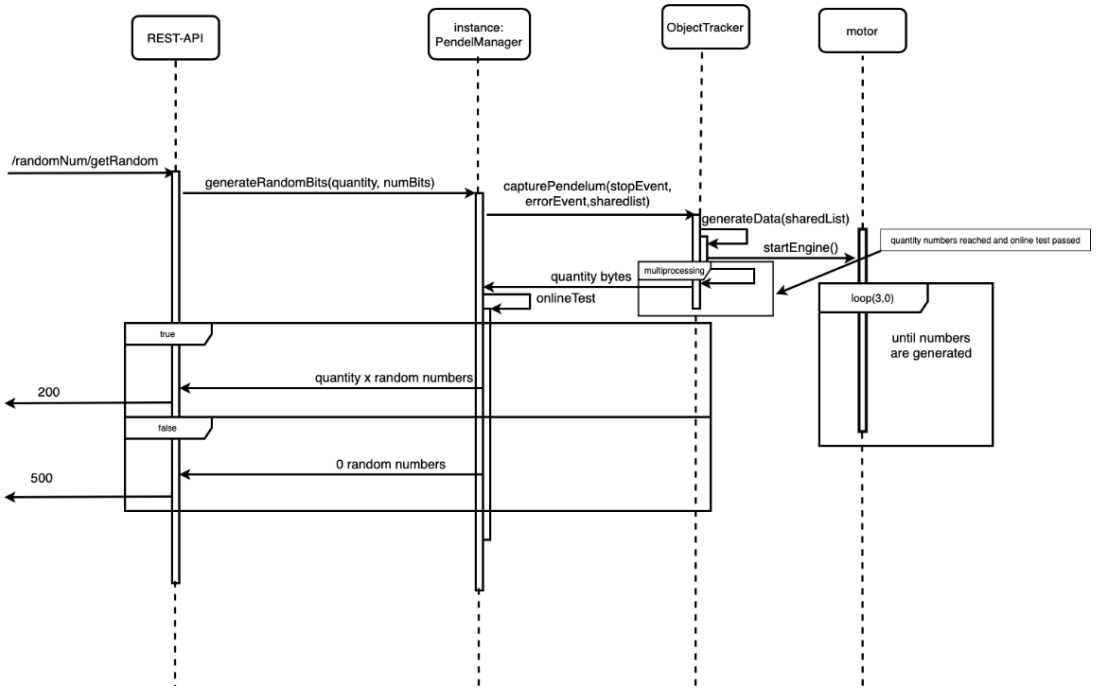

Figure 4: Sequence diagram - getRandom request

The getRandom request consists of the following steps:

1. The generateRandomBits() method is called to generate random bits. This is accomplished by implicitly starting the motor in the code. The motor runs every 7 seconds, re-energizing the pendulum for a duration of 2 seconds.

2. The online test is conducted on a minimum of 1024 bits. The test is executed every second set of 1024 bits that has been generated. Only if the test passes successfully, the next 1024 bits are written into a return queue. If the online test fails, the process is repeated. In the event of three consecutive failures, a 500 response status code will be returned.

3. At the end, the motor stops once enough bits have been generated.

4. If no errors occur, we receive a 200 response status code along with the generated bits.

2.3.3 Shutdown request

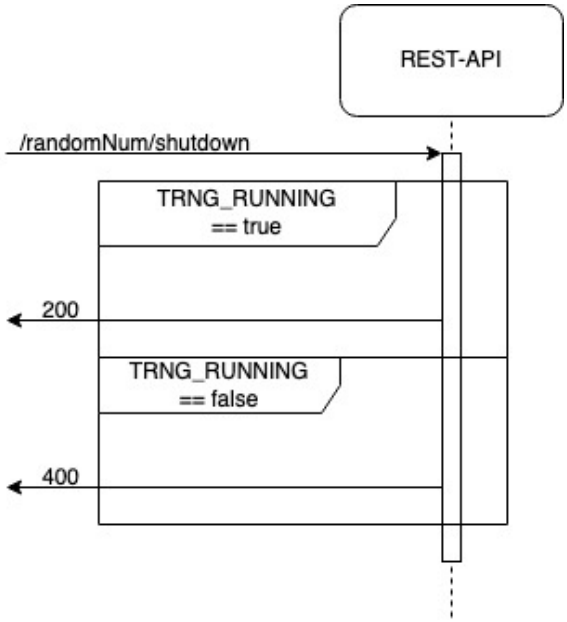

Figure 5: Sequence diagram - shutdown request

The getRandom request consists of the shutdown steps:

1. In this request, the value of the variable TRNG_RUNNING is set to true if it the pendulum is currently running.
2. When the pendulum is turned off, it is set to false, effectively shutting it down. The client receives a status of 200.
3. However, if the pendulum is not running and an attempt is made to turn it off, an error with a status of 400 is returned.

2.4 Circuit diagram

The following section displays the circuit diagram of all the electronic components used, illustrating the connections to the Pi, relays, and INA219 module.

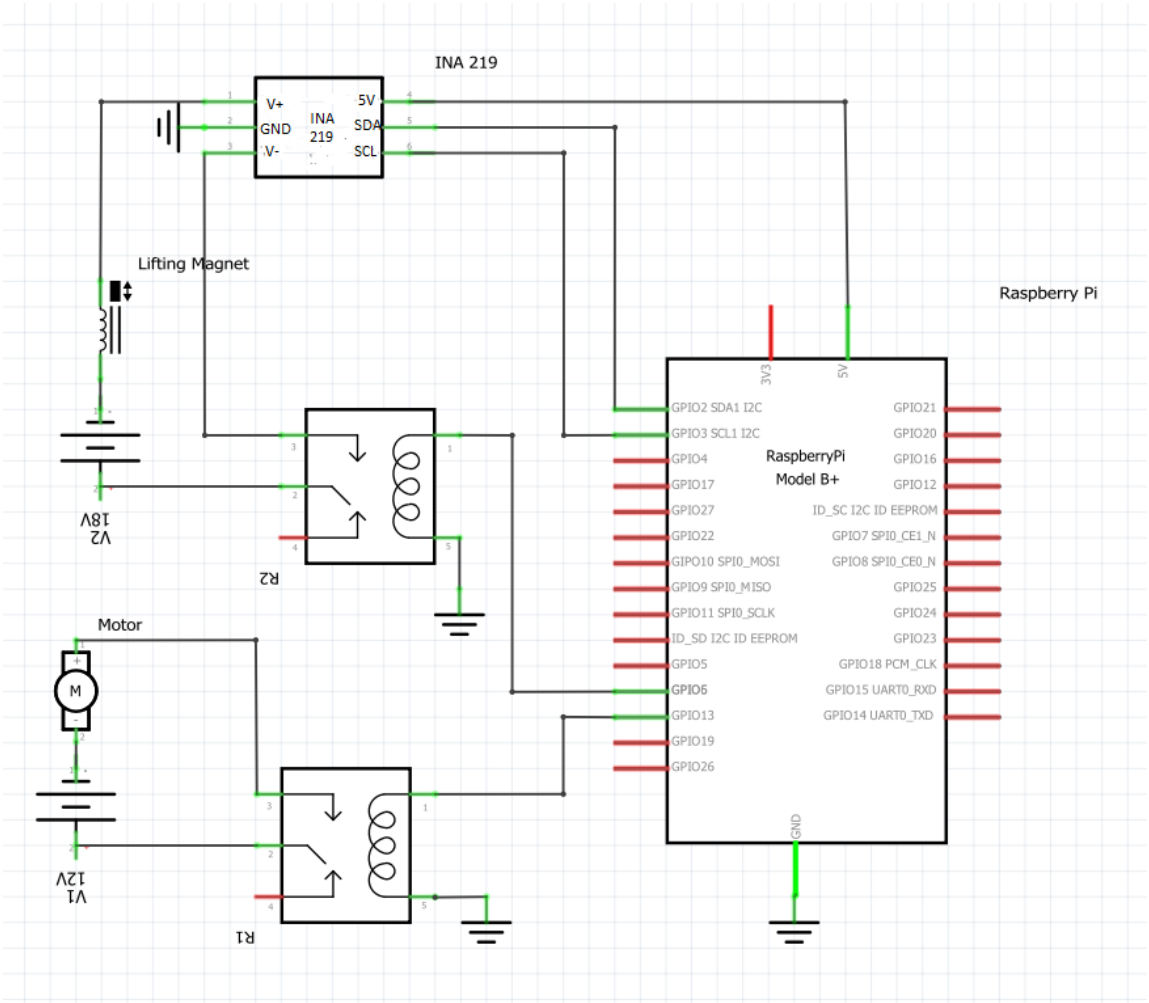

Figure 6: Circuit diagram

3 Installation Instructions

3.1 Product Content

With this TRNG you receive:

Amount	Article
1	Control part
1	Camera part
3	Power plugs
2	Metal bars
3	Pendulum arms
4	Wing nuts
1	8GB micro SD card

The **control part** is already premounted and consists out of:

Amount	Article
1	12V electric motor
1	Raspberry Pi 3B
1	Lifting magnet (Mod: GCs-25.19/V1879 Isliker Magnete)
1	Elegoo 4 Channel DC 5V Relay Module
1	Electronic parts cover
1	INA 219 module

The **camera part** is already premounted and consists out of:

Amount	Article
1	3D printed camera casing
1	Dealikee Raspberry Pi camera module 5MP

3.2 Assembly

To ensure the functionality of the TRNG, it is essential to assemble it following the provided instructions. Additionally, it is necessary to place the prototype on a leveled surface and use a white background.

1. Place the two wooden plates on the leveled surface with the camera facing towards the pendulum.
2. Attach the pendulum arms with about one centimeter distance to the magnet. Make sure to tighten the nuts thoroughly.

3. Take the metal rod marked with an 'L' and place the marked side on the left side of the camera with the arrow facing towards the pendulum.
4. Repeat this process for the rod marked with an 'R' but this time on the right side of the camera.

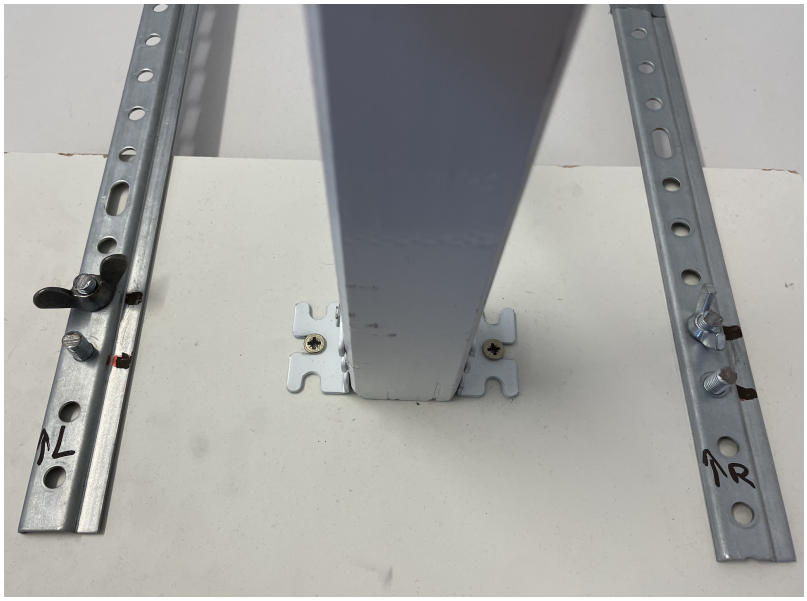

Figure 7: Camera path

5. Use the wingnuts to secure the bars onto the wooden plates. The holes, which should be used, are marked black.
6. Take the cable from the camera, put it through the top of the Raspberry Pi Casing and plug it into the Raspberry Pi as shown on the image. Make sure the male and the female pins are actually connected.

Figure 8: Camera connector

7. Connect your PC and the Raspberry Pi via a LAN cable.
8. Take the cable for the magnet and the engine and plug them into a power outlet.
9. As soon as the power cable is plugged into the Raspberry Pi the prototype will boot himself up and soon will be ready to use.

Read the subsection 'Usage' for instructions on how to use the REST API.

3.3 Warnings

i. With this construction, electricity is present in most parts. It is strongly recommended to disconnect all power supplies before working with or repairing the prototype.

ii. All cables are enclosed within a cable conduit. Please refrain from removing or touching them.

iii. While the prototype is active, the pendulum swings in a circular motion. Please maintain a minimum distance of 1 meter from the pendulum to avoid any potential impact.

iv. While the prototype is initialized, it is important not to interfere in any way with the sampling process.

v. Please refrain from touching any cables or circuit boards integrated within the prototype.

vi. Please use the prototype only in a well-lit environment to ensure that no shadows are captured by the camera. Shadows can significantly affect the quality of the generated random bits.

vii. Please note that may happen that after the generation process, the lifting magnet does not retract and is still fully powered. If this event is not handled, the lifting magnet may cause overheating and can damage the prototype. If you notice that the lifting magnet is not retracting please unplug the power plug for the lifting magnet. And look for further actions in the Troubleshooting section.

4 Usage

4.1 General REST API Endpoints

This table covers various endpoints and their paths for accessing a random number generator in general. The random number generator produces HEX-encoded bit arrays as its output, with leading zeros added if necessary. To add leading zeros was explicitly wished by the client, we strongly advise against adding leading zeros to an hex-encoded random number in a normal context.

Endpoints	Path	Responses	Definition
GetRandomNums	/randomNum/getRandom	200	successful operation; HEX-encoded bit arrays (with leading zero if required)
		432	system not ready; try init
		500	data generation failed; check noise source
InitRandomNums	/randomNum/init	200	successful operation; random bits generator is ready and random bits can be requested
		409	system already running
		500	functionality not given; check hardware
		555	unable to initialize the random bits generator within a timeout of 60 seconds
ShutdownRandomNums	/randomNum/shutdown	200	successful operation; random bits generator has been set to 'standby mode'
		409	system already shutdown

4.2 How to use the REST API

A REST API allows different software applications to communicate and exchange data over the internet using standardized HTTP protocols. Our REST API provides all the necessary functionalities and logic responsible for the aforementioned requests and the initialization of the TRNG and its associated components.

To use the REST API you can choose a HTTP client of your choice e.g. Postman or Curl. In the following enumaration we explain how to interact with the API using Curl.

1. First of all it's necessary to initialize the TRNG with the following command:

```
curl https://172.16.78.60:5520/trng/randomNum/init
```

2. If the request returns a status code 200 you can start generating random bits. To get random bits use following command:

```
curl https://172.16.78.60:5520/trng/randomNum/getRandom?quantity=16numBits=64
```

3. After you recieved your random bits it is possible to put the TRNG into standby mode by sending the following request:

```
curl https://172.16.78.60:5520/trng/randomNum/shutdown
```

4.3 Frontend

In order to have an easy access and control for the REST API, we have got as requirement to implement a frontend webpage. Accordingly, we have developed and implemented a user-friendly interface in order to access and interact with the REST-API´s functionalities.

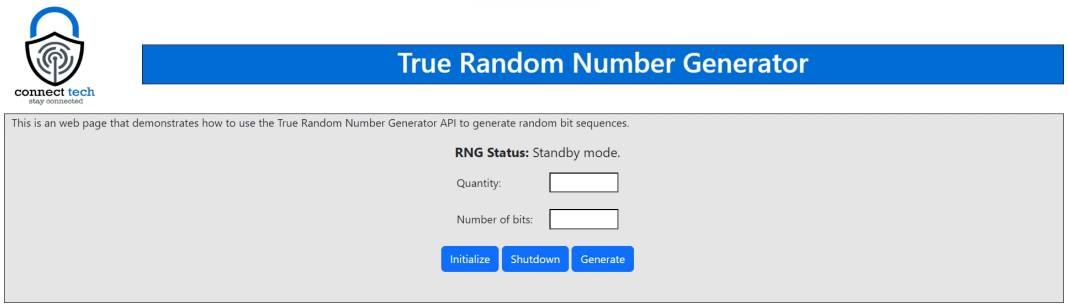

Figure 9: Webpage

- **Quatity:** This text box is intended for specifying the desired number of bits to be generated. It only accepts positive numbers as input.
- **Number of bits:** This text box is for specifying the length of the bits to be generated. It only accepts positive numbers as input.
- **Initialize:** This button is responsible for the aforementioned init request. Once the button is clicked, it will initialize the pendulum and its associated components.
- **Shutdown:** This button is responsible for the aforementioned shutdown request. Once the button is pressed, it will shut down the pendulum and its associated components.
- **Generate:** This button is responsible for initiating the getRandom request mentioned earlier. When the button is clicked, bits will be generated based on the input from two text boxes. The user can specify the desired number of bits and their desired length. Once the bits are generated, they will be displayed in a table format.

The RNG Status text provides the current state of the pendulum, indicating whether it is initialized, shut down, or generating bits.

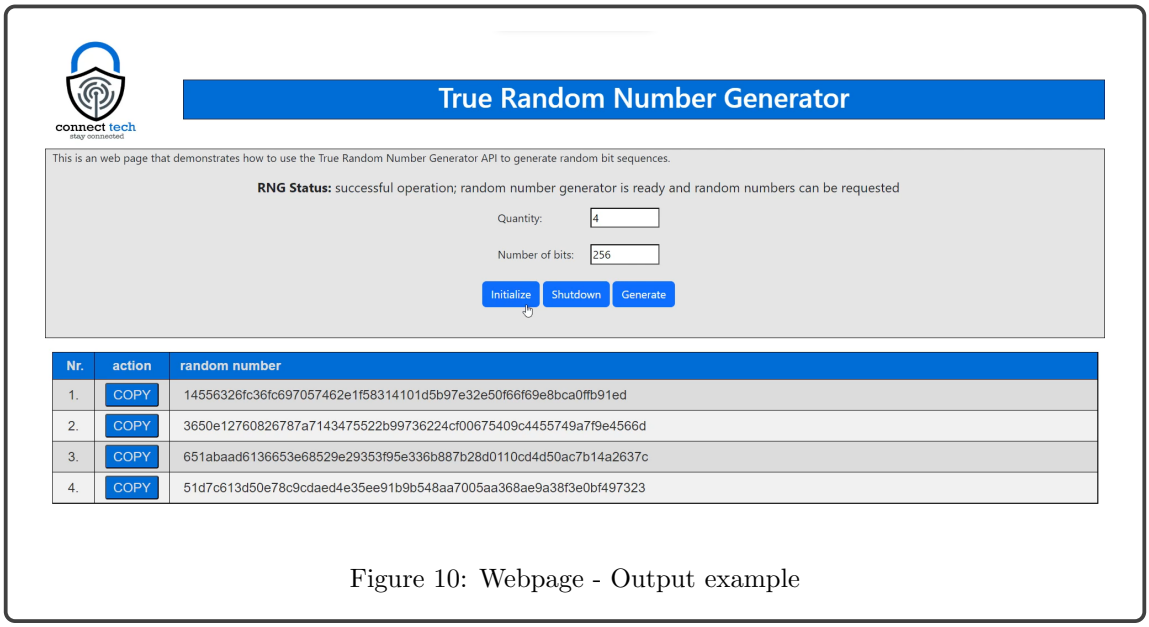

Figure 10: Webpage - Output example

4.4 Maintenance

This table describes all the recommended maintenance intervals to allow a long service time of the prototype.

Intervals	Component	Description
24h	Sliding contacts	The sliding contacts should be exchanged and all cables should be checked.
24h	Lifting magnet	Check the cables of the lifting magnet, they should be connected to the sliding tracks. Make sure the lifting magnet is still glued solid to the drive pulley.
24h	Cables	All cables built in the prototype have to be checked and renewed if necessary.
72h	Drive pulley	Renew the sliding tracks with conductive aluminium tape.
72h	Electric motor	Check all cables and renew them if necessary according to the provided circuit diagramm.
72h	Relay	Check if the relay opens and closes the circuit properly. If not exchange them relay module and connect according to the provided circuit diagramm.
720h	Strap	Check the functionality of the strap and renew them if it has any flaws.
720h	Drive pulley	Oil the hole in the middle of the drive pulley to avoid high friction and squeaking
720h	Ball bearing	Check if they are still in place and working properly. If not exchange them or push them back and glue them in place.

5 Troubleshooting

Component	Problem	Solution
Magnet	Sliding contacts lost connection	Push contacts back on the tracks
	Sliding contact is defect	Exchange sliding contact
	Cable from Magnet to sliding tracks is defect	Glue back cable onto tracks
	Power supply is defect	Exchange power supply
	Lifting magnet is defect	Exchange magnet
Relay	Relay is defect	Unscrew cables and screw them back into place
		Check connection between relay and Raspberry Pi
Motor	Cables are loose	Reconnect cables
	Strap is broken	Exchange strap
	Strap is not in place	Put strap back in place
	None of the above worked	Check connection between relay and Raspberry Pi
		Check power source
		Exchange motor
Pendulum	Pendulum arm is broken	Exchange with 3D printed arm
Camera	No connection	Check cable on Raspberry Pi and on camera
		Exchange camera

5.1 Analyzing

In case non of the above listed Problems is matching your problem, you can try debugging the software and analyze the console output of either the frontend or backend. In the following section you can read how to access the control of the TRNG and analyse the frontend or backend.

Access the Control:
To manage and control our TRNG we used a Raspberry Pi Model 3 B with Raspbian OS Lite 64-Bit Operating System. You can access this Pi with an Monitor connected via HDMI and a USB Mouse and Keyboard. To work with the System you have to login with the given credentials:
Username: connecttech
Password: ConnectTech#1289

Analyze the frontend: The service

6 Test Documentation

6.1 Functionality Tests

These tests are being run to evaluate whether the components for the automation of the pendulum are fully functional. If not the pendulum is not able to generate random bits.

6.1.1 Pendulum movement

Description:
Verifies whether the pendulum moves when the Magnet Test is successful helps determine the functionality of the motor.

Result:
If the pendulum fails to move during the test, it can be deduced that the issue lies with the motor, as other components have been ruled out through the process of elimination.

6.1.2 Lifting magnet

Description:
Measures the applied current on the lifting magnet after the sliding contact. If insufficient current is detected, it returns false, indicating a lack of proper current flow.

Result:
In the event of a test failure, it implies that the lifting magnet is not functioning correctly.

6.1.3 Camera

Description:
Performs a camera functionality test to verify the presence and proper functioning of a connected camera module on the Raspberry Pi.

Result:
The test ensures that the camera is properly detected and initialized, allowing it to capture the pendulum.

6.2 Camera Tests

6.2.1 Camera Cover Test

Description:
In this test, the camera is intentionally covered during live operation, resulting in the camera only displaying a completely black image. The system is designed to detect this error condition and stop generating random numbers.

Result:
The system successfully detects the interference caused by the camera cover, and as a result, no

further numbers are generated. The user is promptly notified of the error with a corresponding error description.

6.2.2 Camera Disturbance Test

Description:
In this test, an object that significantly disrupts the camera's ability to track the pendulum, such as a hand, a laptop, or a mobile phone, is deliberately held or moved within the capture zone during live operation. The system is designed to detect this error condition and stop generating random numbers.

Result:
The system successfully detects the interference caused by the object, and as a result, no further numbers are generated. The user is promptly notified of the error with a corresponding error description.

6.3 Lifting magnet Tests

6.3.1 Remove lifting magnet power supply

Description:
In this test, the cable supplying power to the lifting magnet is intentionally disconnected during live operation. The system is designed to detect this error condition and stop generating random numbers.

Result:
The system successfully detects the interference caused by the power deficit, resulting in the cessation of generating further numbers. The user is promptly notified of the error with a corresponding error description.

6.3.2 Release contact of sliding contacts

Description:
The system successfully detects the interference caused by the released sliding contact, resulting in the cessation of generating further numbers. The user is notified of the error with a corresponding error description, indicating the issue with the sliding contact and its impact on the generation of random numbers.

Result:
The system successfully detects the interference caused by the sliding contact. No further numbers are produced. The user is notified of the error with a corresponding error description.

6.4 Motor Tests

6.4.1 Remove motor power supply

Description:
In this test, the cable supplying power to the engine is intentionally disconnected during live oper-

ation. The system is designed to detect this error condition and promptly cease generating random numbers.

Result:
The system successfully detects the interference caused by the sliding contact. No further numbers are produced. The user is notified of the error with a corresponding error description.

6.5 Statistical Tests

6.5.1 Start-Up Test

Consist out of the following tests:

- Monobit
  - Assesses the proportion of zeros and ones in the entire sequence. It determines whether the observed fraction of ones is close to 1/2, indicating randomness.
- Chi Squared
  - Test to evaluate if it follows a discrete uniform distribution.
- Total failure Test
  - Tests for the presence of specific patterns in a binary sequence using the Approximate Entropy Test. It calculates the test statistic and p-value to determine if the sequence deviates significantly from randomness

6.5.2 Online Test

Consist out of the following tests:

- Monobit
  - As mentioned before.
- Block frequency
  - Test to evaluate if the proportion of ones in each block follows a specific distribution.
- Run Test
  - Test to evaluate the presence of patterns of consecutive ones or zeros.
- Longest one block
  - Test to check the longest run of ones within a binary data string.

6.5.3 BSI Tests

Description:
The BSI Test Execution for the Pendulum True Random Number Generator (TRNG) involves conducting specific tests outlined in the BSI Paper PTG 2.0. These tests include:

Randomness Test: Validates the quality and randomness of generated random numbers.
Entropy Source Test: Assesses the entropy sources used by the system.
Statistical Test Suite: Applies statistical tests to evaluate randomness properties.

Each test is performed according to the instructions in the BSI Paper PTG 2.0.

7 Component Information

In the case of one of the prototype components breaking, you can find a list of the components to re-buy it.
If any of the 3D printed components break, they can be printed using pearl white PLA printing material. All files are complementary in our GitHub repository.

Component	Specification
RaspberryPi	RASPBERRY PI 3B+ Raspberry Pi 3 B+, 4x 1,4 GHz, 1 GB RAM, WLAN, BT
Relay	4 channel DC 5V Relay with Optokoppler for Arduino UNO R3 1280 DSP ARM PIC AVR STM32
Lifting magnet	ITS-LZ-3869-Z-24VDC lifting magnet 30 N 59 N 24 V/DC 16.8W
Engine	10:1 Metal gearengine 37Dx50L mm 12V
RaspberryPi fan	Low-Profile CPU Cooler with Raspberry Pi Heatsink for Raspberry Pi 3B+/3B
RaspberryPi camera	Camera compatible with Raspberry Pi Camera 5MP OV5647 Webcam Video 1080p for Raspberry pi 3 B+/3B/2B/4B.
INA219 module	INA219 I2C Bi-directional DC Current Power Supply Sensor Power Monitor Sensor Module 3-5.5V
RaspberryPi camera cable	display cable 15 pins FFC (2m)

8 Contact Details

Members of ConnectTech:

- Carlo Bauer
- Paul Betsch
- Marina Göppel
- Stefan Kleinhenz Leiva
- Carsten Michel
- Lukas Siegle

To contact us please refer to our Product Owner Carsten Michel under 2120533@stud.hs-mannheim.de