HECTOR

D4.2

Demonstrator platforms accompanying report

Project number:	ICT-644052
Project acronym:	**HECTOR**
Project title:	Hardware Enabled Crypto and Randomness
Project Start Date:	1st March, 2015
Duration:	36 months
Programme:	H2020-ICT-2014-1
Deliverable Type:	DEM
Reference Number:	ICT-644052 / D4.2/ 1.0
Workpackage:	WP4
Due Date:	December 2017 - M34
Actual Submission Date:	21st December 2017
Responsible Organisation:	TCS
Editor:	Emeline Hufschmitt
Dissemination Level:	PU
Revision:	1.0
Abstract:	Deliverable D4.2 contains the complete setup of the three HECTOR demonstrators designed in the framework of work-package WP4. The specifications of each demonstrator are those described in D4.1. This document accompanying the demonstrators gives a full description of each demonstrator from motivation to user manual guide. Compliances to requirements are completed and key performance indicators highlighted.
Keywords:	Demonstrator, true random number generator, physically unclonable function, authenticated encryption

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 644052.

Editor

Emeline Hufschmitt (TCS)

Contributors (ordered according to beneficiary numbers)

Sandra Lattacher, Martin Deutschmann (TEC)
Valdimir Rožić, Bohan Yang, David Singelée (KUL)
Viktor Fischer, Oto Petura, Ugo Mureddu (UJM)
Valentin Mascré (TCS)
Bernard Kasser (STR)
Ruggero Susella (STI)
Marcel Kleja (MIC)
Maria Eichlseder (TUG)
Gerard van Battum, Marnix Wakker (BRT)

Disclaimer

Executive Summary

Deliverable D4.2 consists of the complete setup of the demonstrator platforms. The aim of the demonstrators is to validate cryptographic primitives developed during the HECTOR project:

- *True Random Number Generator*,
- *Physically Unclonable Function*,
- *Authenticated Encryption with Associated Data,*

and to illustrate their relevance for the real-world security.

The three selected demonstrators, whose specifications are fully described in D4.1 [7], are the following:

Demonstrator 1 Stand-alone, high performance secure random number generation device;

Demonstrator 2 Secure portable USB data storage;

Demonstrator 3 Secure Messaging Device.

The demonstrators are implemented on two hardware platforms. While Demonstrator 2 and 3 are portable, small form-factor devices powered by the USB bus, Demonstrator 1 needing higher performance is powered by an external supply. The three demonstrators feature a USB interface to be connected to the host computer and a metallic case ensuring electric shielding and tamper evidence.

The demonstrators have been completely developed and validated and this document's purpose is to provide a summary of the motivations as well as high-level descriptions of the specifications, installation requirements and user guides for each demonstrator.

Contents

List of Figures

Chapter 1

Introduction

Deliverable D4.2 consists of the complete setup of the demonstrator platforms as output of T4.2 and T4.3. The software and hardware specifications of the HECTOR demonstrator platforms were fully detailed in D4.1. This deliverable includes the following items:

- the integration of the hardware modules in the platforms and system level design including control unit and communication interface,
- the development of a software interface accessing the hardware module from the PC via USB,
- the development of a user friendly GUI to observe the auditing capabilities.

Demonstrator 1 is a high throughput secure TRNG. It exploits two physical random number generators: a PLL-TRNG and a DC-TRNG. A typical application could be a personalization center where high volumes of high quality random numbers are required to initialize secure devices.

Demonstrator 2 is a secure portable USB data storage. It is a personal, single-user device used to protect data at rest. It implements a TERO-PUF, the AEAD algorithm ASCON 128-a, a PLL-based true random generator and a two-factor authentication protocol.

Demonstrator 3 is a secure messaging device which offers the possibility to exchange secure messages between two devices, over an insecure network and through untrusted PCs. It makes use of the same hardware platform as Demonstrator 2 but implements communications between host and PC differently in order to encrypt and authenticate stream of messages from one device to another.

This document is accompanying the platforms delivery. It is divided in three main chapters, each dedicated to one demonstrator. For each of them the motivation is first given, in order to give perspective to the demonstrator. Then each demonstrator specifications and building blocks are briefly described. Next part concerns installation and user manual description. Finally conformance to requirements are reviewed and completed.
The last chapter concerns recommendations R4 and R5 from the report of the second review meeting. Detailed answers are given using inputs from each demonstrator purpose and specification.

Chapter 2

Demonstrator 1: Standalone, High Performance Secure Random Number Generation Device

2.1 Motivation

The aim of Demonstrator 1 is to show that the proposed TRNG designs and especially the new stringent TRNG design approach can be successfully applied in high-end real-world data security applications, in which a secure and robust TRNG is an essential construction block. The proposed demonstrator could be used as a stand-alone security peripheral – the source of high quality random numbers – in high-performance data servers and communication systems. Demonstrator 1 could serve in the future as a high-speed random number generator, exploiting two physical random number generators: a PLL-TRNG and a DC TRNG. This kind of generators could find their typical application in personalization centers where high volumes of high quality random numbers are required to initialize a large quantity of devices.
The security evaluation of Demonstrator 1 shows its compliance with the AIS20/31 requirements [6], verification of the quality of the TRNG output, observation of entropy margins during variation of the TRNG operational conditions and verification if the behavior of the TRNG is in line with the stochastic model.
Besides being compliant with the AIS 20/31 document, HECTOR Demonstrator 1 is also compliant with the newly developed US standard NIST SP 800-90B [13] and stringent recommendations of the French DGA (Direction Générale de l'Armement), currently being edited. Simpler AIS 20/31 compliance demonstration lowers security evaluation and certification costs, which should result in having more products going through the certification process and increase confidence in the solution.

2.2 Description of the Demonstrator 1 Platform

Demonstrator 1 is a physical true random generator including cryptographic post-processing algorithm, which is aimed at generation of random bitstreams achieving security level PTG.3, as defined in AIS 20/31. The generation of random numbers is provided as a security service for the user.

2.2.1 Hardware Design

The functional block diagram of the Demonstrator 1 platform is depicted in Fig. 2.1. The main intellectual property (IP) blocks are implemented in three FPGAs:

- the Altera Cyclone V FPGA contains the phase-locked loop based TRNG (PLL-TRNG),
- the Xilinx Spartan 6 FPGA contains the delay chain based TRNG (DC-TRNG),
- the Microsemi SmartFusion2 FPGA contains a control unit based on the ARM Cortex M3 core, cryptographic post-processing block, and data interfaces including RAM memory interface.

The TRNG cores are connected to the main control FPGA using fast low voltage differential signaling (LVDS) interface used for data acquisition and slow synchronous serial interface used as control interface.
Raw random bit streams coming from the two TRNG cores are acquired (cryptographically post-processed or not) in the 64 MB RAM memory using the direct memory access (DMA). The memory is divided in two halves: one is used for data acquisition and the second one is used as a solid disk, which is accessible from the host computer (PC) as a data storage medium (mass storage device). Consequently, the maximum size of the acquired data block is 32 MB. Thanks to the DMA transfer and the use of fast data interface, no data is lost during data acquisition.

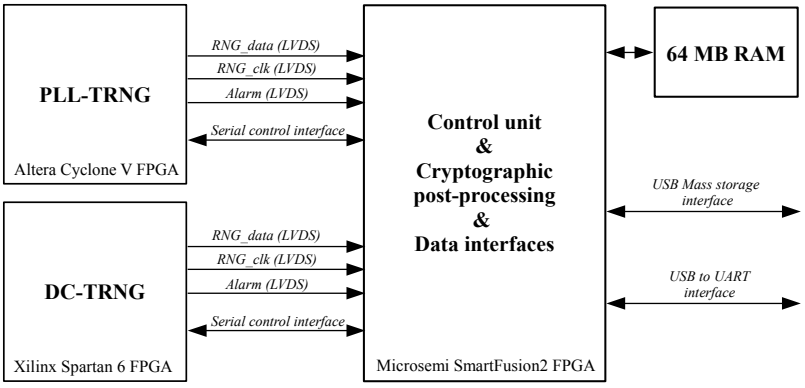

Figure 2.1: Functional block diagram of the HECTOR Demonstrator 1 platform

The block diagram of the Demonstrator 1 hardware is depicted in Fig. 2.2. Besides the three FPGA devices and the external memory presented in the previous paragraph, it contains a USB hub, and two USB interfaces (USB to UART convertor from the FTDI company and USB physical layer), an SD card connector, and other data and power connectors.
The SD card can contain device drivers, which can be downloaded to the host PC. Demonstrator 1 communicates with the host PC via two USB ports: a UART serial interface (virtual COM port) and a mass storage device interface. The UART port is used to send 64-bit commands or small data blocks and to receive 64-bit demonstrator state word or small data blocks. The mass storage device port is used to transfer high volume data files (up to 32 MB) from Demonstrator 1 to the host PC.
The Demonstrator 1 hardware is depicted in Fig. 2.3. Three FPGA areas visible in the picture are separated by a ground plate. They are closed in three separated chambers inside the metallic shielding.

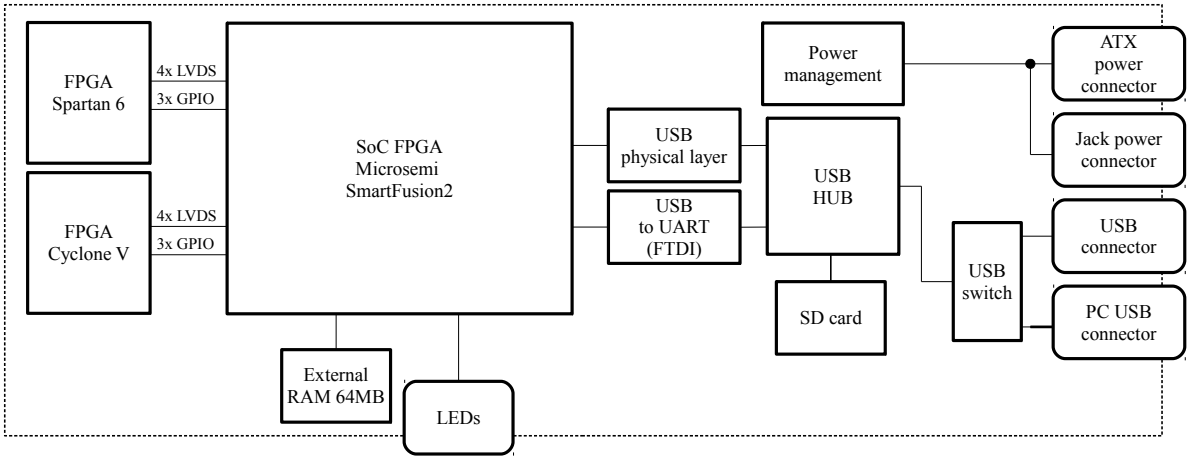

Figure 2.2: Block diagram of the HECTOR Demonstrator 1 hardware

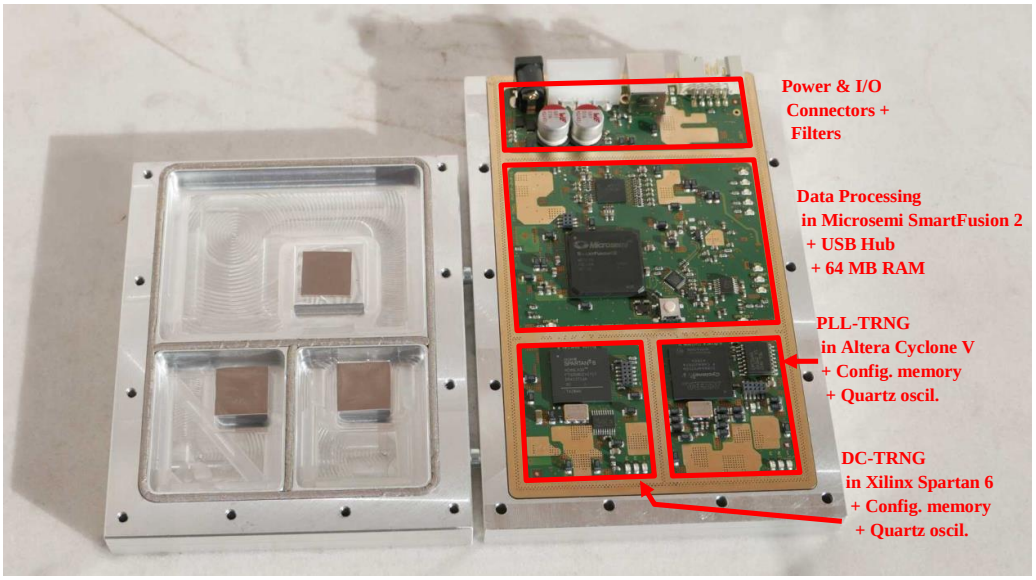

Figure 2.3: HECTOR Demonstrator 1 hardware

2.2.2 Description of the PLL-TRNG Core

The final version of the PLL-TRNG proposed in the framework of the HECTOR project uses the differential jitter between two PLLs as a source of randomness. It is depicted in Fig. 2.4. The generator is accessible in two modes: Evaluator mode and User mode.

Evaluator Mode

In Evaluator mode of the PLL-TRNG, two kinds of RNG core outputs can be obtained to verify its correct operation, including operation of the two PLLs, of the sampling flip-flop, and of the XOR gate:

1. **Output of the XOR gate** can be used to characterize the jitter and to verify correct operation of the two PLLs,

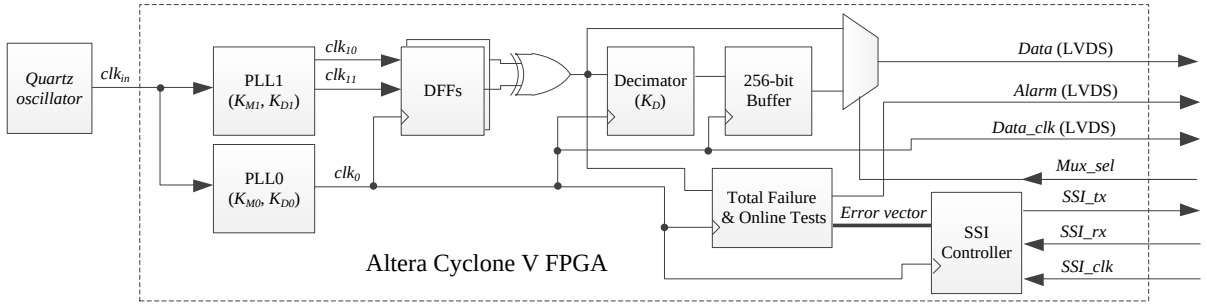

Figure 2.4: Block diagram of the PLL-TRNG core implemented in Altera Cyclone V FPGA as a part of HECTOR Demonstrator 1

2. **Raw binary signal**. Access to this signal is needed to apply Procedure B of the TRNG evaluation, in which tests T6 to T8 of AIS 31 need to be applied on the generated raw binary signal.

Availability of Evaluator mode is a very interesting option since it makes security evaluation of the generator much easier, especially by reducing the cost of certification.
The type of core output is selected using the *Mux_sel* signal. By default, the raw random data output is selected.

User Mode

In User mode of Demonstrator 1, the raw random bitstream featuring entropy rate per bit of at least 0.997 as required by AIS 20/31 is sent to the cryptographic post-processing block implemented in the SmartFusion2 device. In User mode, the user has thus access only to the cryptographically post-processed output signal.

Parametric Statistical Tests

Output of the XOR gate is used internally by both the Total failure and Online tests. The Total failure test (test T0 [6]) counts number of random samples appearing in each of 255 periods T_Q (one period T_Q is composed of K_D periods of clk_0). If no random sample was found, the Total failure alarm (*Err_t0*) is triggered and the system interrupt is requested.
Two Online tests observing the XOR gate output are implemented:

- the Online test T1 [6] computes parameter P_1 corresponding to the number of random samples during subsequent periods T_Q (a period T_Q is the period of pattern appearing at the output of the XOR gate, which depends on frequency ratio of the clock signals generated in PLLs),

- the test T2 [6] computes parameter P_2 corresponding to the variance of the clock jitter measured during 4080 periods T_Q.

Computed values P_1 and P_2 are compared with thresholds obtained from the stochastic model and required for the Shannon entropy rate per bit of 0.997 ($4 < P_1 < K_D/4$, $228 < P_2 < 1280$).

The execution time of the Online test (4080 periods T_Q, i.e. about 4 ms) represents a compromise between precision of entropy estimation and reactivity of the test. While guaranteeing a sufficient precision, it is much shorter than that of the fastest general-purpose statistical tests FIPS-140-1 [10], which need at least 20 000 random bits (i.e. almost 5 times more bits).

2.2.3 Description of the DC-TRNG Core

The Delay-Chain based TRNG uses jitter accumulated in a free-running ring oscillator as a source of randomness. It is depicted in Figure 2.5. The role of the tapped delay chain is to enable samplings from the output of the ring oscillator using very high timing resolution (approximately 17 ps on a Xilinx Spartan-6 FPGA). The data sampled by the delay chain is a digital representation of the waveform of the ring-oscillator output signal. Due to the white-noise sources that are present in all electronic circuits, the frequency of the free-running ring oscillator is not stable. Timing positions of the Ring Oscillator (RO) output signal edges, relative to the reference clock become more uncertain over time. The variance of this timing uncertainty is proportional to the jitter accumulation time (the period of the sampling signal clk_A).

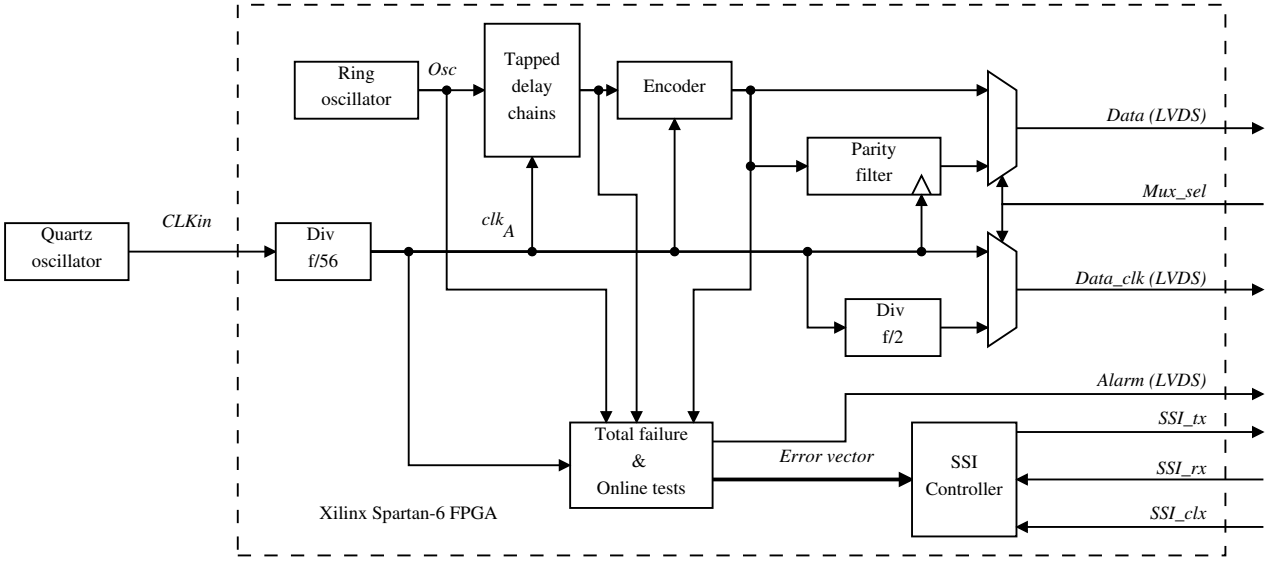

Figure 2.5: Block diagram of the DC-TRNG design

The output Osc of the entropy source (free-running ring oscillator) is connected to a delay line consisting of $m = 40$ serially connected delay buffers. The output of each buffer connects to a flip flop. These m flip-flops comprise an m-bit register that is clocked by the clk_A signal. This signal is obtained by down-sampling the reference clock signal provided by an on-board quartz oscillator. The down-sampling factor is a design parameter that is chosen to allow for a sufficient jitter accumulation time. The output of the register is connected to a priority encoder which encodes the position difference of two consecutive samples into a single output bit.
The generator can work in two modes: Evaluator mode and User mode.

Evaluator Mode

In Evaluator mode of the DC-TRNG, the raw random bits (before the algorithmic post-processing) can be collected to derive parameters for online tests. This data output can be

used as input for Procedure B of the TRNG evaluation, too. Tests T6 to T8 of AIS 31 can thus be applied on the raw random bits [6].
In order to provide an additional security margin while fulfilling the bitrate requirements of the output, a two-stage parity filter is used to improve the robustness of the TRNG core against possible active attacks. This two-stage parity filter XORs two consecutive raw random bits to generate one random bit. The output of the parity filter can be read in the DC-TRNG Evaluator mode, too.

User Mode

In User mode of Demonstrator 1, the random bit output of the algorithmic post-processing is connected to the input of the cryptographic post-processing implemented in the SmartFusion2 device. The random bit sequence meets the AIS 20/31 requirements, according to which this input needs to feature a 0.997 entropy per bit as the lower bound.

Embedded Statistical Tests

Two total failure tests are provided in this implementation. The first total failure test uses an edge detector to check whether the free-running ring oscillator is toggling or not. The edge detector is reset every two clock cycles. It triggers an alarm if the input signal has not changed. The second total failure test is designed to generate an alarm if there is no edge sampled in the tapped delay chain.
DC-TRNG contains two statistical tests that detect long-term weaknesses in the generated raw data bitstream. These two tests have different false-alarm rates. On-line test 1 (Sensitive test) has a higher false-alarm rate but is more efficient in detecting attacks. On line test 2 (Robust test) is less efficient in detecting attacks but it is more robust against false alarms. Both tests work on a sequence of 512 consecutive raw bits. The Online test 1 is counting the overlapping template 111 in the sequence. At the end of 512-bit sequence, the counter value is compared with the pre-computed upper and lower boundaries ($35 < N_{111} < 93$). If the counter exceeds these boundaries, an alarm signal is generated. The false alarm rate of this test is 1%. Online test 2 is tracking the XORed result C_1 of two consecutive bits in the 512 consecutive raw bits. At the end of the sequence, result value is computed and compared with the pre-computed boundaries ($185 < C_1 < 326$). The false alarm rate of this test is 10^{-6}.

2.2.4 Description of the Control Device Hardware

The SmartFusion2 System on Chip (SoC) FPGA combines the FPGA fabric with a simple hardwired microcontroller sub-system (MSS). The role of the MSS and its firmware is explained in the next section. Besides the control logic accessible by the local microcontroller, which is described in the next section, the FPGA fabric in the Demonstrator 1 control device contains two serial-to-parallel converters, two sets of continuous tests and a cryptographic post-processing block (see Fig. 2.6). Two continuous tests ensure compliance with requirements of the French DGA and with those of NIST SP 800-90B [13]: the Repetition count test and Adaptive proportion test. Both tests evaluate continuously the raw random bit stream just before the cryptographic post-processing block (as required by NIST SP 800-90B). Consequently, they test correct operation of all RNG blocks until the cryptographic post-processing (and namely the decimator and serial-to-parallel converter in PLL-TRNG and the parity filter in DC-TRNG, which are not tested by dedicated tests of the two TRNGs, as required by the French DGA).

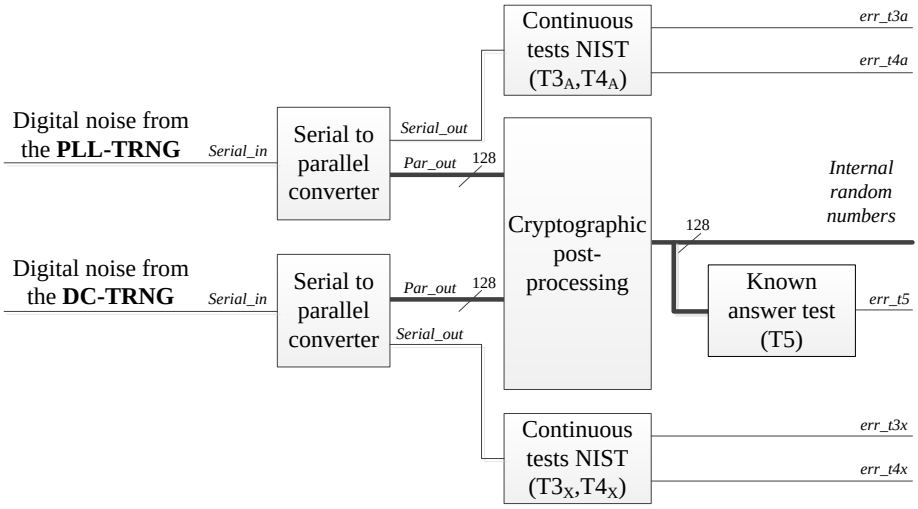

Figure 2.6: Block diagram of hardware in the Demonstrator 1 control device

The execution time of the Continuous tests is similar to that of Online tests (4096 periods T_Q, i.e. about 4 ms).
Results of functional and statistical tests are saved in a 64-bit state register. The contents of the state register is sent to the control interface of the RNG, which can be reached by the system processor of Demonstrator 1.
If any of the error bits is asserted, the RNG triggers a high speed alarm and a system interrupt is requested. At the same time, generation of the raw signal is stopped, input shift register of the cryptographic post-processing unit is not filled in any more, and consequently, the cryptographic post-processor will not get ready until the RNG is not reset by the system processor.
After the reset was asserted (not represented in Fig. 2.4), the startup procedure is launched and once succeeded, the random data can be available again.

2.2.5 MSS Firmware

The main role of the SmartFusion2 SoC FPGA in Demonstrator 1 is to communicate with the host PC, to create and to manage the file system, and to control data acquisition.
The acquired data are saved into one half of the RAM memory (data buffer), which accessible from the MSS. The MSS controls communication with the host PC via serial communication protocol (a COM bus) and creates, manages and exploits a file system, which is placed in the second half of the RAM memory and which is accessible from the PC as a USB mass storage device (see Fig. 2.7).
The heart of the MSS is a simple microcontroller hardwired in the SmartFusion2 device. It is based on the ARM Cortex M3 core. The Demonstrator 1 firmware is divided into few parts: a UART packetizer-depacketizer, a communication control unit and a USB mass-storage class driver.
The FPGA fabric in Demonstrator 1 contains RAM interface, a set of 32-bit data registers available for MSS, a command execution controller, and other associated logic.
Data acquisition is initialized by the host PC. The PC sends a command packet to the MSS a COM port. The command is decoded and sent into the communication control unit. The

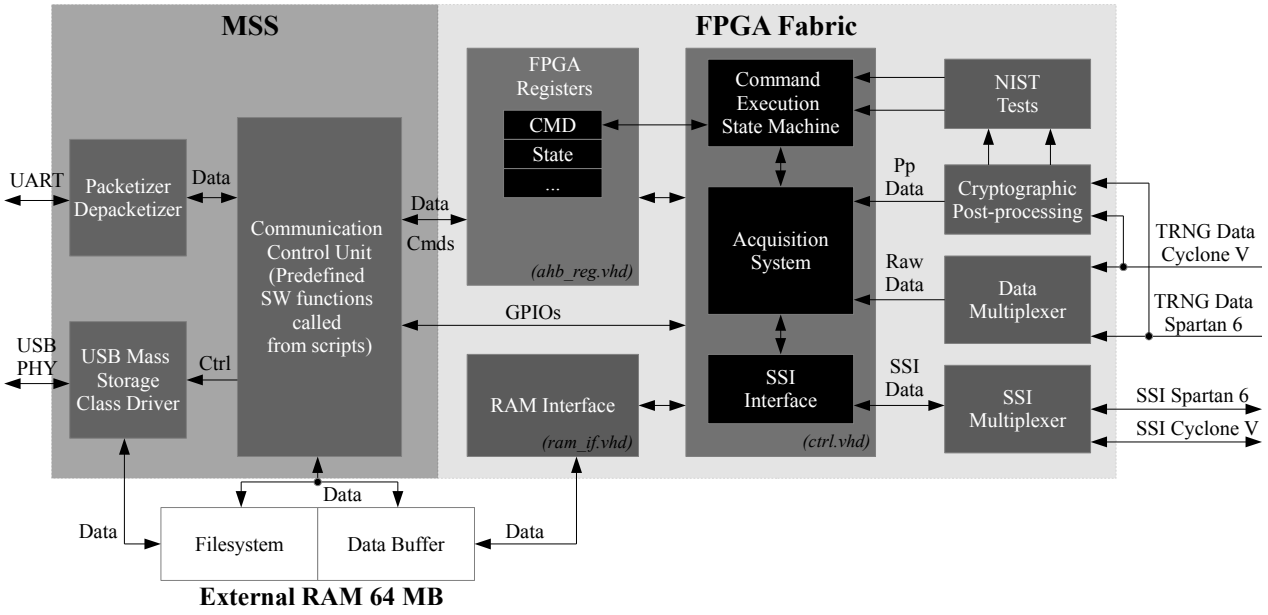

Figure 2.7: Block diagram of hardware means in Demonstrator 1 SoC

control unit initializes the FPGA fabric, which starts to acquire selected data stream to the 64-MB RAM memory available inside the demonstrator. At the end of data acquisition, the MSS confirms execution of the command using a command confirmation packet, which is sent back to the PC. During data acquisition, the PC requests continuously MSS about evolution of the data acquisition process. After data acquisition, the host PC sends another command, which creates a file system in the demonstrator memory and copies the acquired data into a new binary file created in the file system. Demonstrator 1 is thus seen from the PC as a mass storage device and the newly created file can be read by the host PC using a mass storage device driver (i.e. Demonstrator 1 behaves as a USB key connected to the PC).

2.3 Installation and User Manual

In this section, we describe shortly hardware and software tools that constitute the Demonstrator 1 toolkit. Next, we describe steps needed to install Demonstrator 1 hardware and software. Finally, we present user interface and the way the demonstrator can be operated.

2.3.1 Prerequisites

The Demonstrator 1 toolkit is composed of the following hardware and software tools:

- Demonstrator 1 hardware including USB cable
- Demonstrator 1 power adapter
- Host PC including Windows 7, 8, or 10 or Linux operating system
- TCL interpreter. Windows TCL interpreter is embedded in the Demonstrator 1 software bundle. On Linux, following software must be installed:
  - TCL version >= 8.5

  - Tk version >= 8.5
  - TkLib with Plotchart module
- Set of Demonstrator 1 control TCL scripts
- FTDI virtual com-port (VCP) driver

2.3.2 Installation

Hardware Setup

Demonstrator 1 can be used as an external device or it can be installed inside a host desktop PC into the 3.5 inch rack. If it is used as an external device, the Demonstrator 1 power adapter must be used and plugged into the corresponding connector of the demonstrator. Then, the delivered USB cable (type A-B) must be used to connect demonstrator with the host PC.
If Demonstrator 1 is used as an internal device, it must be first fixed in the 3.5-inch bay. Standard 10-wire flat USB cable and ATX power cable must be used to install the demonstrator hardware in the PC.

Software Setup

Once the demonstrator hardware is installed and the demonstrator is powered and plugged into the external or internal USB bus, the availability of the FTDI VCP driver must be checked. In the Linux OS, the driver is installed by default, but Windows needs the driver to be installed manually. If the driver is installed, the operating system will assign a COM port to it.
At this stage, the user has to determine manually, what is the COM port number that has been assigned to demonstrator. In Windows OS, this can be done using the Device manager. In Linux, the COM port is represented by a device node `/dev/ttyUSBX`, where X corresponds to the COM port number. This COM port number must be specified by the user in the graphical demonstrator interface as explained in the following section.

2.3.3 User Manual

Demonstrator 1 is accessible using the Tool Command Language (TCL) scripts, which describe the Graphical User Interface (GUI) and controls Demonstrator 1 functions. Before the demonstrator is used for random data acquisition or evaluation, its default parameters must be set up.

Setting Up Default Parameters of Demonstrator 1

After the Demonstrator 1 is connected to the PC and its COM port is determined, one can launch the Demonstrator 1 GUI interface in order to communicate with the hardware. This can be done in Windows by running the `run.bat` file, or in Linux by running the `tclsh gui.tcl` command.
When launched, the Demonstrator 1 interface will greet the user with a welcome splash screen shown in Fig. 2.8

Figure 2.8: Demonstrator 1 GUI welcome splash screen

After the splash screen is closed, the main interface is presented (see Fig. 2.9). In this interface the user must specify initial parameters of Demonstrator 1:

- **HECTOR board COM port** – full name of the COM port as determined in the previous section (COMX in Windows or /dev/ttyUSBX in Linux).
- **File name (without extension)** – requested output file name. The software will automatically add a `.bin` suffix to this name. The acquired TRNG output data will be stored in the resulting file.
- **File size [bytes]** – requested file size in bytes. This number must be a multiple of 4.

Reset Demonstrator 1 and its Blocks

Once all the fields are set to correct values, the user can send commands to the Demonstrator 1 hardware. It is a good practice to reset Demonstrator 1 before it is used for the first time after the power up. This can be done through the Reset menu, which contains several items:

- **Reset All** – resets all the connected hardware (SoC and both TRNGs).
- **Reset SoC** – resets only the Microsemi SmartFusion2 SoC.
- **Reset PLL TRNG** – resets only the PLL TRNG.
- **Reset DC-TRNG** – resets only the DC-TRNG.

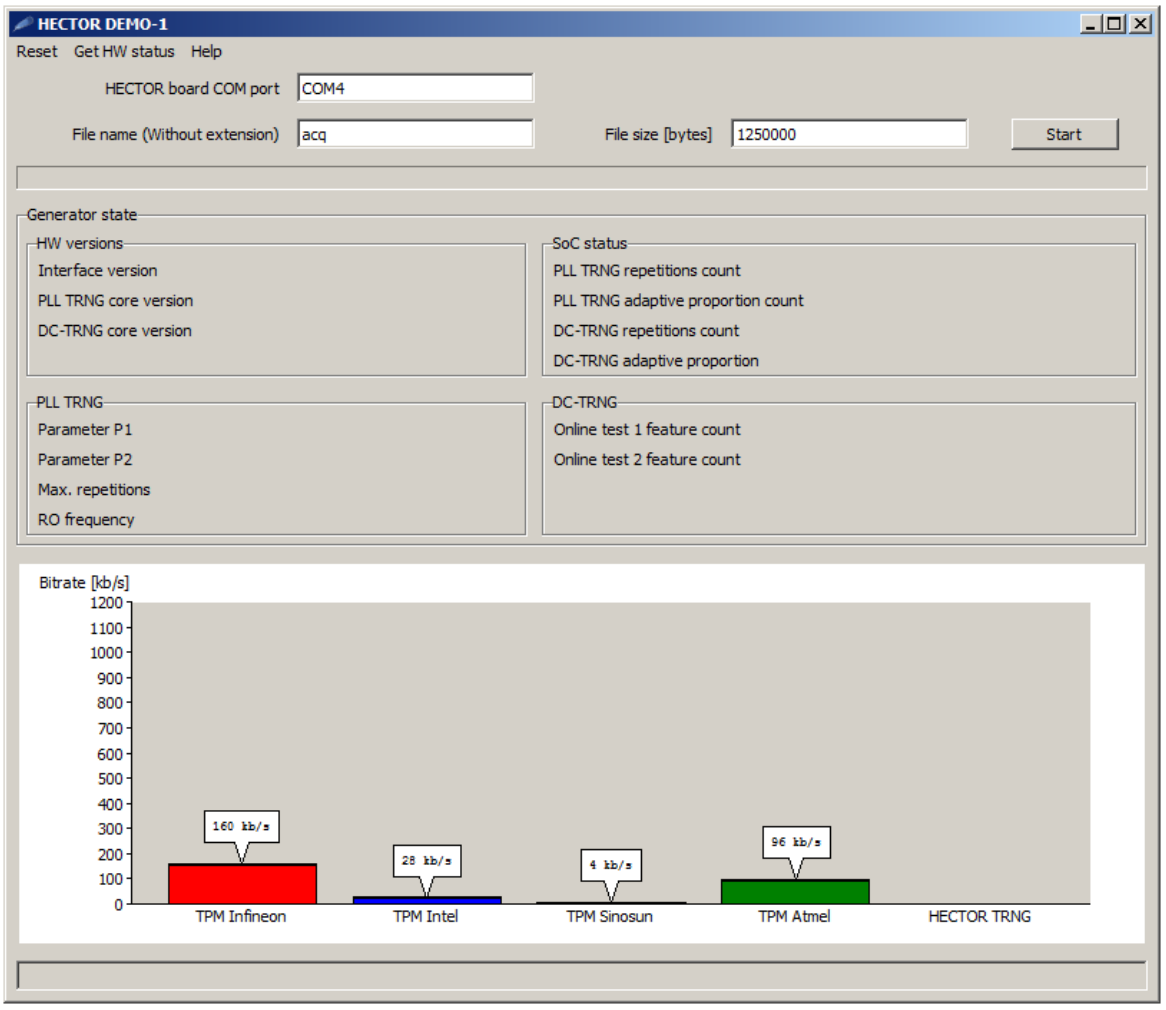

Figure 2.9: Demonstrator 1 GUI main screen

TRNG Status Request and Data Acquisition

When the device is reset, the user can retrieve either only the status of the demonstrator, or its status as well as TRNG output data. Each time, a corresponding log file containing status of the demonstrator is created and saved in data directory. In order to retrieve status only, one can click on the *Get HW status* in the main menu.

In order to retrieve random data and TRNG status, the user must click on the *Start* button on the main screen. Data acquisition will then start and after it's finished, the Demonstrator 1 status together with output bit rate is reported on the main screen as shown in Fig. 2.10.

Parameters measured by online tests are shown along with their limiting values on the main screen under *Generator state*. If any of measured parameters is out of range, an error message is shown and data acquisition is terminated (no data is acquired).
If no error occurred, the binary file is created at a location specified by the user.

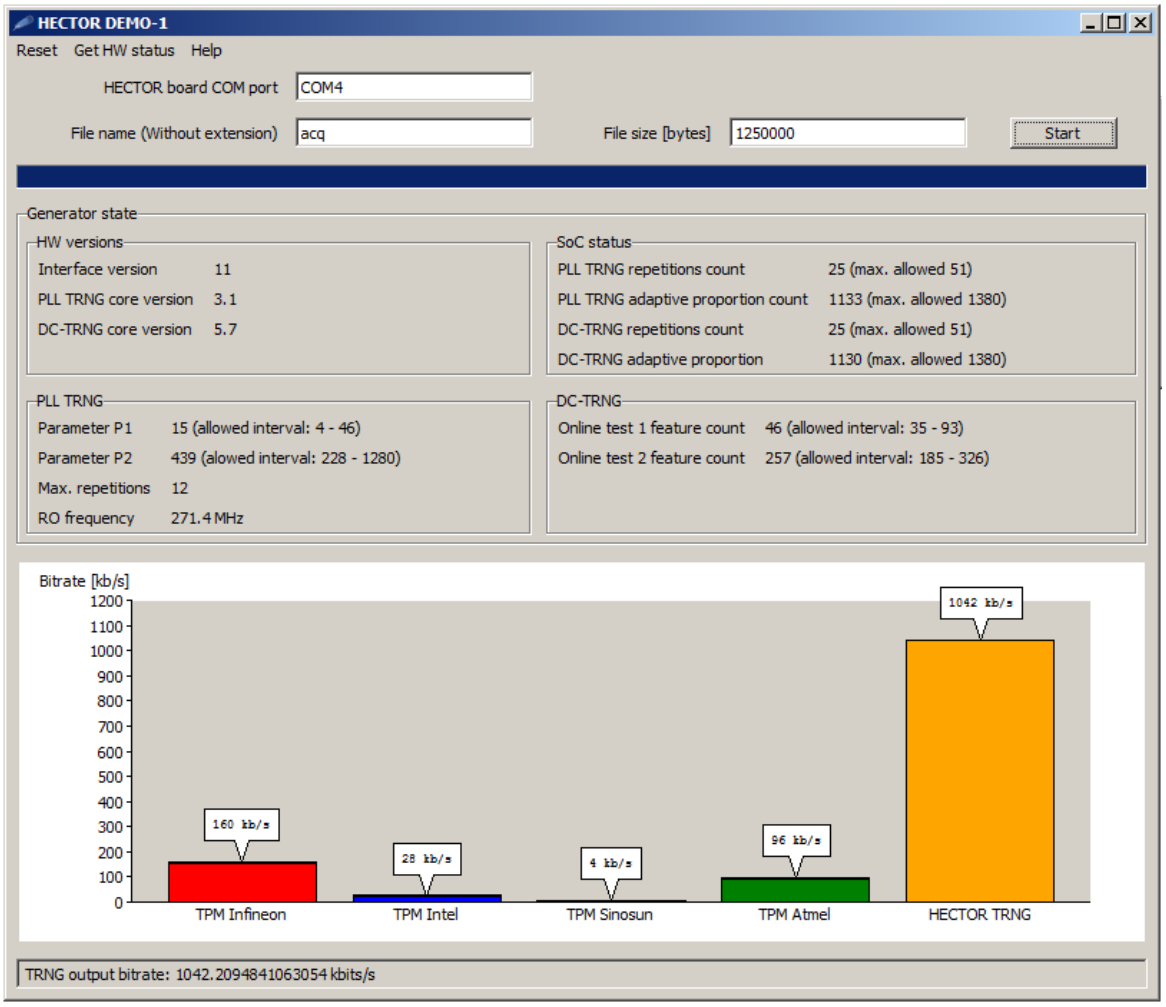

Figure 2.10: Demonstrator 1 GUI screen showing data acquisition details

2.4 Conformance to Requirements and Key Performance Indicators

The following table shows the status of Demonstrator 1 regarding requirements defined in D1.2.

Conformance to Requirements for Demonstrator 1

Requirement	Current status	Remark
Functional requirements		
Ready after 3 seconds	< 4 ms	**Satisfied**
Output data rate over 1Mb/s	1.04 Mb/s	**Satisfied**
Secure DRNG	DRG.3 Implemented	**Satisfied**
Hardware requirements		
Availability of an embedded processor in the control FPGA	ARM Cortex M3 embedded in SmartFusion2 FPGA	**Satisfied**
USB connectivity between the control FPGA and the host PC	USB Mass Storage Device and UART interface used	**Satisfied**
The use of up to 15k logic cells in the control FPGA	5k logic cells	**Satisfied**
The use of up to 10k logic cells in FPGAs with RNG cores	1k (in Cyclone V) and 1k (in Spartan 6) logic cells	**Satisfied**
Total USB consumption smaller than 5W	4.5 W	**Satisfied**
Data interface faster than 2 Mb/s	16 Mb/s	**Satisfied**
Security requirements		
TRNG is AIS20/31 PTG.3 compliant	Validated by BrightSight	**Satisfied**
TRNG require less than 300mW	80 mW	**Satisfied**
TRNG startup tests require less than 3 seconds to complete	< 4 ms	**Satisfied**

2.4.1 Key Performance Indicators

Table 2.4.1 shows main key performance indicators for Demonstrator 1.

Key Performance Indicators for Demonstrator 1	
Security	High security AIS20/31 [6] PTG.3 compliant NIST SP800-90B [13] compliant (online tests implemented) French DGA requirements (not published yet) compliant
Speed	Bitrate over 1 Mb/s – considerably higher than solutions benchmarked in [11]

Chapter 3

Demonstrator 2: Secure Portable USB Data Storage

3.1 Motivation

USB flash drive or *USB memory* are popular means of data storage, back-up and transfer. Such devices may contain sensitive personal information and/or company assets in digital form, e.g. bank statements, legal documents, commercial treaties, sales and billing documents, source codes, technical documentation or other intellectual property. The risk for the information to be compromised is high when commuting, travelling, or just leaving the drive unattended in a car or a hotel room. To **enhance privacy** one has to use a secure solution. Such a product then contains not only a flash memory and a USB interface but also a hardware encryptor. Vendors offer secure USB portable devices where data are protected by approved cryptographic algorithms, e.g. AES. However weaknesses have been discovered in existing solutions [4]. There is also a lack of trusted secure storage solutions engineered and manufactured in EU.
The goal of Demonstrator 2 is to show that the proposed cryptographic primitives, algorithms and protocols in HECTOR can be successfully applied in this real-world data security application, and that user and device authentications are sufficiently strong and secure.
Demonstrator 2 was also inspired by MICRONIC's **hardware platform** already available at the beginning of the project.

3.2 Description of the Platform

Demonstrator 2 is a secure portable USB storage, see Fig. 3.1. It is a personal, single-user

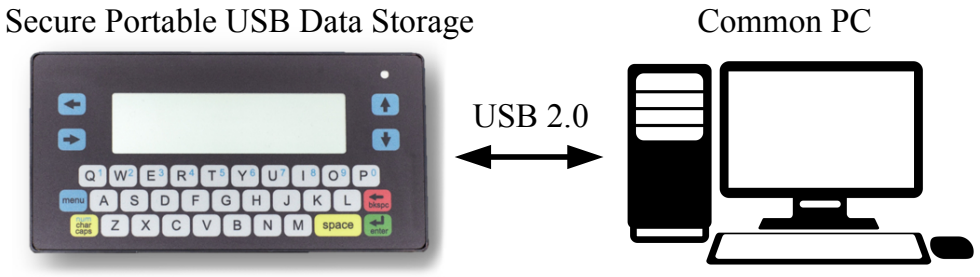

Figure 3.1: HECTOR Demonstrator 2 overview.

device. It protects the data stored on it while being at rest. It requires the user to authenticate before allowing access to data. It is powered from the USB bus without the need of an external power source.
The secure USB storage device contains encrypted data that can be unlocked by supplying a user passphrase code through the built-in demonstrator keyboard. In principle USB storage devices are physically controlled by the user of the device. Security is provided by the protection of the device itself and by the fact that the owner is the only person that knows the passphrase. The device has been designed to resist attacks in the "Lost and found" scenario. The data shall be secured at rest also when an attacker gains access to the not-activated device without knowing the passphrase. When found (or stolen), an attacker can only try a user defined number of passphrases before the device will permanently lock its data.

3.2.1 Hardware Design

The hardware platform of demonstrator 2 is given in Fig. 3.2.

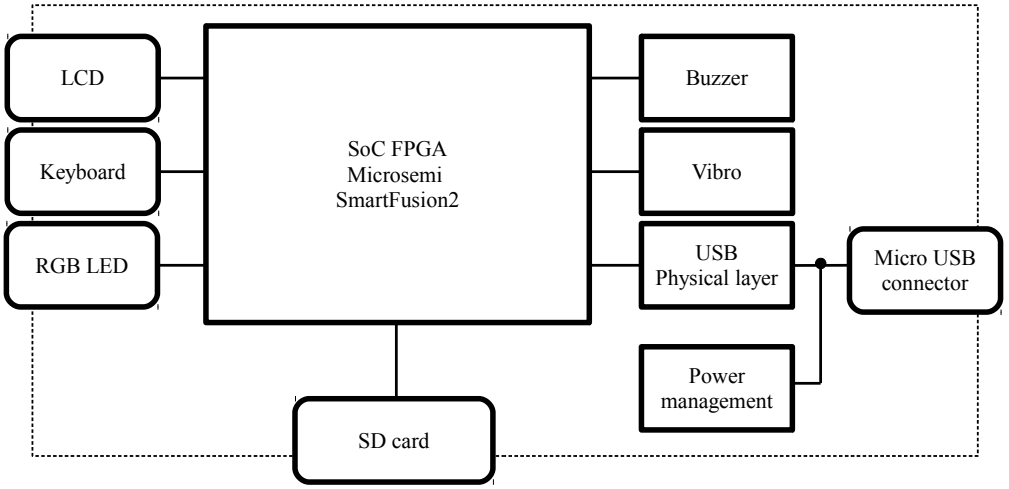

Figure 3.2: Demonstrator hardware platform.

The HECTOR demonstrator integrates a SmartFusion2 SoC FPGA which consists of two parts:

- the Microcontroller Subsystem which integrates a Cortex M3 processor and other subsystems like SPI, I2C, USB controllers,
- the FPGA fabric which implements hardware blocks which communicates via the AHB bus by the firmware running on the Cortex M3 (see Fig. 3.3)

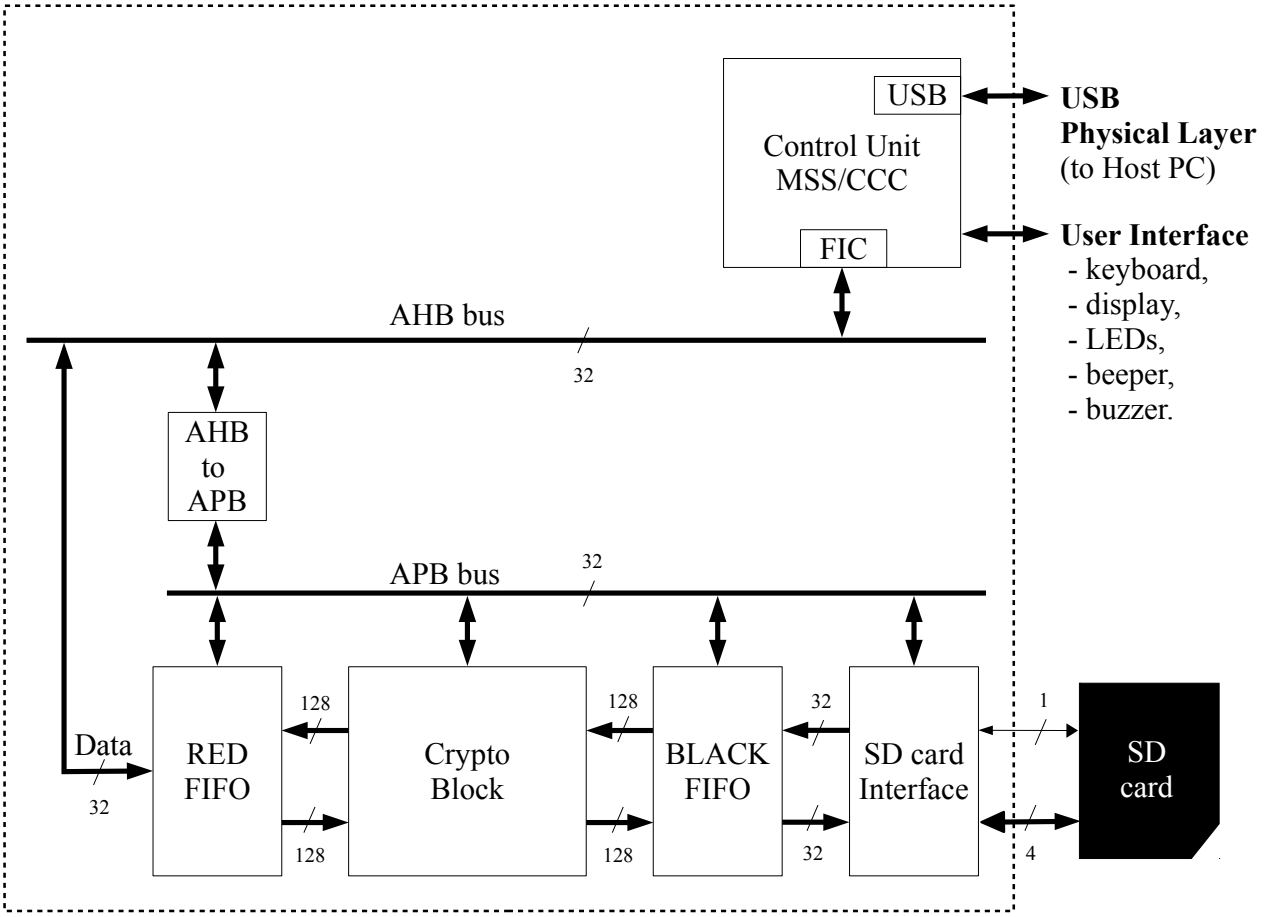

Figure 3.3: FPGA top architecture.

A Secure Digital (SDHC/SDXC) card is connected to the FPGA fabric to store the encrypted data.

3.2.2 Building Blocks

Ascon

Demonstrator 2 makes use of an Authenticated Encryption cipher Ascon, a third-round candidate currently competing in the ongoing CAESAR competition. The Ascon specification describes a family of authenticated encryption designs [1]. In the framework of Work Package 4 the consortium agreed on the use of an instantiation dubbed "Ascon-128a", which is short for $\text{Ascon}_{12,8}$-128-128. This instantiation can process two inputs, metadata and plaintext, in 128-bit blocks, and produces a ciphertext and tag by encrypting the plaintext, and authenticating both metadata and plaintext. The plaintext can be empty to obtain a pure message authentication code ("integrity-only mode").
In demonstrator 2, Ascon is used at every step:

- in authentication mode only and in encryption mode in the two-factor authentication protocol (see below),
- in encryption and authentication mode during the data encryption process.

The algorithm used in the demonstrator is fully implemented in the FPGA. The core is a

slightly modified generic implementation from [2]. The core is wrapped in a block which adapts it to the demonstrator needs – fixed lengths of associated data, sector plaintext and ciphertext.

TERO-PUF with Post-processing

The implemented PUF is composed of 128 TERO cells, two counters and a subtractor. The protocol to generate a 2-bits response to a challenge is the following:

1. Select the pair of TERO cells to be compared
2. Activate the TERO cells (ctrl signal = 1) for $2\mu s$
3. Deactivate the TERO cells
4. Generate a strobe (which means subtractor output is ready to be stored)
5. Reset counters

This is done for all implemented TEROs and the resulting bits are concatenated to form a 128-bits response. The response is stored in a shift register.
PUF – TERO cells and counters are implemented in FPGA. The output value from the PUF hardware block is post-processed by firmware in the ARM processor. The post-processing follows all the recommendations of [8]:

- **Majority voting** – the reference response is defined as the most likely response over the first 50 readouts during enrollment ;
- **Dark bit selection** – the 13 least stable bits during enrollment are discarded as dark bits. We therefore end up with a 115 bit PUF response ;
- **Kang's scheme** – to reconstruct the PUF, Kang's scheme is used as described in [5]. The helper data needed for PUF reconstruction is stored in eNVM ;
- **Golay code** – the error correction code is Golay code (23, 12, 7) running in five loops.

PLL-based TRNG with Embedded Tests

Demonstrator 2 uses a TRNG with online tests and cryptographic post-processing that has been provided in the framework of T2.1 as a functional block in VHDL. The demonstrator platform features a physical TRNG, which uses differential jitter between clock generated in two PLLs as a source of randomness. The TRNG implemented in Demonstrator 2 uses the Total failure test (test *T0*) and On-line tests (tests *T1* and *T2*) described in D4.1 [7].
Each call to this embedded TRNG produces 128 bits of random data.

Passphrase

The pass-phrase consists of eight words. It is a human-readably encoded high-entropy random number generated by the TRNG during enrollment. The Diceware_8k list, a list of 8192 words up to 6 characters long, is applied to the number. The user has to learn the phrase and enter it whenever activating the device.
A retry counter must be chosen during the enrollment phase to prevent brute force attacks on the pass-phrase.

Two-factor Authentication Protocol

A two factor protocol has been developed in HECTOR project, see [7] chapter 3.2.1. The 128-bit master key (key encryption key) is derived from a high entropy pass-phrase (96 bits) and from device's PUF (32 bits). The master key is used to encrypt the data encryption key (DEK) before storing it in the demonstrator. The DEK is also generated by the TRNG during enrollment. When the user authenticates , firstly, the authenticity of the helper data is verified and then the DEK is decrypted and verified.

Non-volatile Memory (eNVM)

Besides the firmware, the embedded flash memory contains device settings (language, sound), retry counter, helper data and DEK. The helper data and the DEK are stored along with the corresponding nonces and authentication tags. Neither component of the master key is saved in the demonstrator.

3.2.3 Main State Machine

The main state machine running on the devices is shown in Fig. 3.4. The device is either empty,

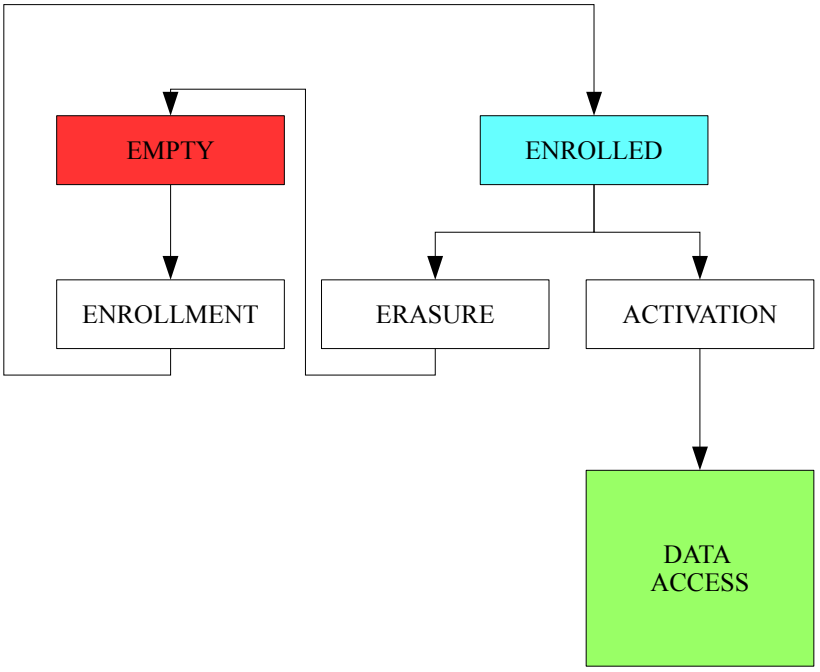

Figure 3.4: Main states of Demonstrator 2.

deleted or enrolled. It is **empty** just after manufacturing. After the delivery the user enrolls it. The embedded non-volatile memory of an **enrolled** device contains PUF helper data which are necessary to reconstruct the PUF response, encrypted key and their authentication tag. The device is **deleted** on user request or after too many unsuccessful attempts to enter the passphrase, at which point the data in the eNVM and on the SD card are zeroized.
The user **activate**s his enrolled device by entering the passphrase. The key is decrypted, verified and up-loaded to the corresponding register. The crypto block is then ready to encrypt or decrypt data during sector write or read operations. The device is ready to be mounted by the operating system. After the user finishes his work, he should securely unmount the device.

3.2.4 Software Structure

The demostrator firmware is composed of modules displayed in Fig. 3.5. After the system initialization the firmware loops in the main state machine. The loop is interrupted on keyboard strokes (user's interaction) and USB communication.

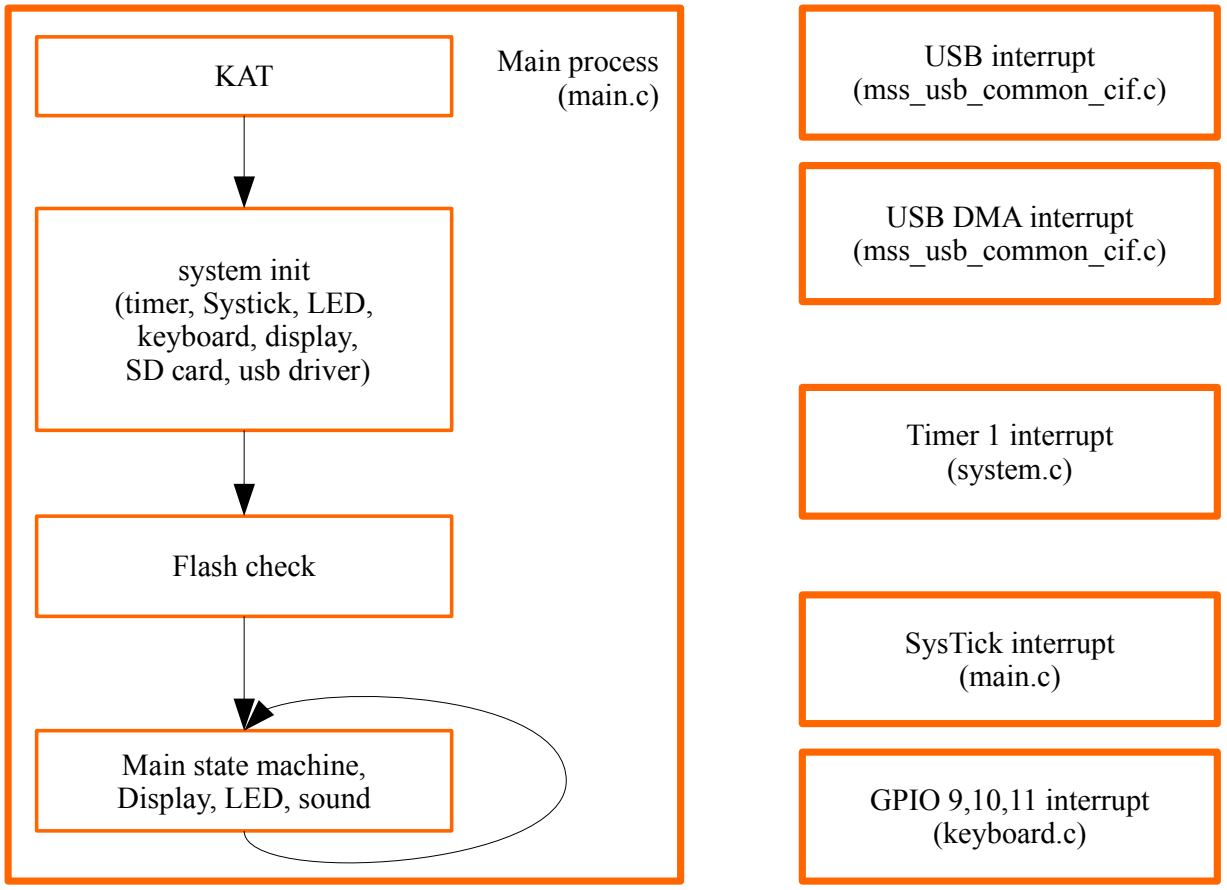

Figure 3.5: Firmware modules.

3.2.5 SoC Resources Usage

The final project was analyzed in Libero 11.8 SmartPower tool and the resulting power consumption of the whole project and the particular blocks is displayed in Tab. 3.2.5. The values are computed for typical operational conditions, where the junction temperature is 25°C.

Power consumption in mW

Demo 2 SoC	Crypto Block	Ascon core	TERO-PUF	PLL-TRNG (PLL0 + PLL1)
268	110	7.8	5.6	12.6 + 9.5

The project uses two oscillators, one driven by an external crystal at 12 MHz and the second one internal RC ocillator at 50MHz. The later one drives clock conditioning circuits (PLLs) of the TRNG. Another CCC/PLL generates a 75 MHz system clock. A MSS and 52 inputs/outputs are

also used. The top hierarchy exploits 12.526 (45.23%) logic elements. Table 3.2.5 summarizes the other resources.

Resources used

Resource	FPGA	Crypto Block	Ascon core	TERO-PUF	PLL-TRNG
4LUT + CC	12072 (44%)	9666	1762	2298	351
DFF	5632 (20%)	3531	324	185	333
RAM1K18	18 (58%)	11	0	0	1
Chip Global	12	7	0	3	4
Row Global	205	27	0	4	23

3.2.6 Data Transfer Rate

The speed of data reads and writes is one of the most important characteristics of a device. The speed of Demonstrator 2 was tested using a notebook Lenovo T460p, Microsoft Windows 7 Professional 64-bit and USB Flash Benchmark application.

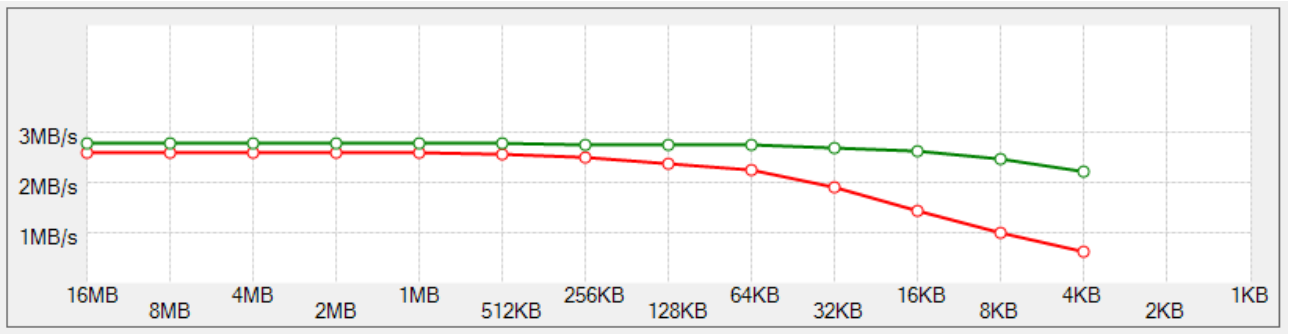

Figure 3.6: Data transfer rate. Reading in green, writing in red.

3.3 Installation and User Manual

Front Panel

Figure 3.7: The front panel of the device

The front panel is the user interface of the device. It consists of the LED, the display and the keyboard. The LED diode indicates the following situations:

- when the light is continuously blue ⇒ the disk is attached, enrolled, but inactive yet;
- when the light is blinking blue ⇒ a key is pressed;
- when the light is continuously green ⇒ the disk is active;
- when the light is blinking green ⇒ a transfer of data is currently occuring;
- when the light is blinking red ⇒ the disk is being deleted;
- when the light is continuously red ⇒ the disk has been erased or an error occured.

Menu key. If pressed for at least 3s, the **menu** key deactivates the device and disconnects it from the computer. This operation is available only if the device is active.

Num char caps. Key **num char caps** switches between the lower-case letters and numerals on the keyboard. The mode is displayed by **(a)** and **(1)** symbols.

Horizontal arrows. Keys to the left of the display allows horizontal movement within the phrase being entered and enable to change the values in menu items.

Vertical arrows. Keys to the right from the display allow vertical movements in the phrase being entered and vertical movements in the menu to make or confirme certain choices.

Bkspc. Key **bkspc** deletes a character to the left from the cursor return without carrying out an operation.

Enter. Key **enter** is a confirmation button.

3.3.1 Disk Enrollment

The device is delivered empty. The empty device has to be enrolled before its use. Two encryption keys are generated during the enrollment. One key is used for data encryption and authentication. The second main key protects the data encryption key when stored in the device. The main key is never stored in the device. It is reconstructed from the user pass-phrase each time the user activates the device. Just an empty device can be enrolled. Use the provided USB cable to attach the disk to a computer. The device turns on automatically. The LED is red. The information about the device appears on the LCD display.

```
HECTOR Demo 2 version: 1.02   .
(c) MICRONIC, a.s.            .
Serial nr:  64EB3104 06AA2609 .
Press "enter" to continue     .
```

Press `enter`.

```
Disk is DELETED               .
                              .
Disconnect the device or      .
Press "enter" to continue     .
```

Press `enter`.

```
>Enrollment                   .
 Device setup                 .
 Evaluation                   .
                              .
```

Choose the option `Enrollment` and press `enter`.

```
Maximal number of tries (1-20).
3                             .
                              .
Enter number and press "enter".
```

Set the limit of attempts to enter the pass-phrase and press `enter`. Exceeding the number makes the disk automatically erase.
The keys are generated during the enrollment by a physical (true) random number generator in the device and will be used to encrypt the data. The pass-phrase representing the main key appears.

```
The passphrase is: flake stall.
 vs mig canary 65 motel pot   .
                              .
enter - accept, bkspc - repeat.
```

If you are not comfortable with the pass-phrase press `bkspc` to generate a new one. Memorize the activation phrase. Any written copy of the pass-phrase must be kept in a safe place. Press `enter` to proceed.

```
(a):                          .
                              .
                              .
                              .
```

Enter the pass-phrase. The character in the brackets indicates the keyboard mode – `(a)` lower-case letters and `(1)` numerals and special characters. The yellow `num-char-caps` key in the bottom left allows you to switch the keyboard mode. Use the arrows for horizontal and vertical moves within the phrase. Press `bkspc` to delete the character before the cursor. Press `enter` when done. If an incorrect phrase has been entered the following notification appears:

```
Phrases are not identical!    .
                              .
                              .
Press "enter" to continue     .
```

Press `enter` to repeat the pass-phrase.
If the phrase is correct, disconnect the device.

```
Disk is enrolled.             .
                              .
                              .
Disconnect the device         .
```

3.3.2 Installation of the Disk

The HECTOR Demonstrator 2 is a Mass Storage Class USB device supported across all operating systems. Thus the installation of the device runs in most operating systems automatically after it has been attached to the computer. If the operating system supports the format of the disk, a portable disk will emerge in the system. The most universal format recognized by operating systems is FAT32.

3.3.3 Activating and Deactivating the Disk

Disk Activation

Only an enrolled device can be activated. Use the provided USB cable to attach the disk to a computer. The device turns on automatically. The LED is blue. The information about the device appears on the LCD display.

```
HECTOR Demo 2 version: 1.00   .
(c) MICRONIC, a.s.            .
Serial nr:  64EB3104 06AA2609 .
Press "enter" to continue     .
```

Press `enter`.

```
>Activate the disk            .
 Device setup                 .
 Delete the content           .
                              .
```

Select the `Activate the disk` option and press `enter`.

```
(a):                          .
                              .
                              .
                              .
```

Enter the pass-phrase. The character in the brackets indicates the keyboard mode – `(a)` lower-case letters and `(1)` numerals and special characters. The yellow `num-char-caps` key in the bottom left allows you to switch the keyboard mode. Use the arrows for horizontal and vertical moves within the phrase. Press `bkspc` to delete the character to the left of the cursor. Based on the settings of the device, there may be a beep or the device vibrates, or the LED diode blinks blue when a key is pressed.

```
(a):flake stall vs mig canary .
65 motel pot                  .
                              .
                              .
```

Press `enter`.
If the correct phrase is entered the LED lights up green on and the disk is connected to the computer.

```
Disk is activated.            .
Before detaching the device   .
unmount the disk in the OS    .
and press "menu".             .
```

A new disk appears in the operating system. If the disk has been just enrolled, it is not formatted and it is necessary to do so.
If something is missing in the pass-phrase the following notification appears:

```
Passphrase too short!         .
                              .
                              .
Press "enter" to continue     .
```

Press `enter` to correct the pass-phrase. If a typo occurred in the pass-phrase the following notification appears:

```
xth word not in dictionary!   .
                              .
                              .
Press "enter" to continue     .
```

Press `enter` to correct the pass-phrase. If an incorrect phrase has been confirmed one of the following notification appears:

```
Wrong phrase!                  .
Remaining number of tries to   .
enter the passphrase:       4 .
Press "enter" to continue      .
```

or

```
Wrong phrase!                  .
The last try to enter          .
the passphrase.                .
Press "enter" to continue      .
```

Press `enter` to correct the pass-phrase. If, however, the correct activation phrase is not entered within the prescribed limit of invalid attempts the disk will be erased.

Deactivation and Safe Removal of the Disk

In order to detach the disk, unmount it from the operating system first. The icon for unmounting portable disks is situated on the toolbar in the right bottom corner of the computer screen. Performing the operation remove safely is crucial to allow the operating system to finish writing all data to the disk. Afterwards, hold the key menu for at least three seconds. After three seconds you can hear a beep and the LED turns from green to blue. This means that the disk has terminated reads and writes to the SD card, deactivated the cipher core and now it can be detached from the computer. If the safe removal procedure is not followed and the device is instantly detached from the computer, there is a risk of data damage! The damage can be so severe, that the whole data contents of the disk goes unavailable and there is no way to restore it. Avoid such a risk and also periodically back up data. The producer is not liable for loss of the data and subsequent incurred damage.

3.3.4 Erasing the Disk

Disk Erasure Initiated by the User

Only an enrolled device can be erased. Use the provided USB cable to attach the disk to a computer. The device turns on automatically. The LED is blue. The information about the device appears on the LCD display.

```
HECTOR Demo 2 version: 1.00    .
(c) MICRONIC, a.s.             .
Serial nr:  64EB3104 06AA2609  .
Press "enter" to continue      .
```

Press `enter`.

```
 Activate the disk             .
 Device setup                  .
>Delete the content            .
                               .
```

Choose the option `Delete the content`.

```
Deleting the disk             .
                              .
Press "arrow down" to continue.
Press "bkspc" to return       .
```

Press `bkspc` to cancel the deletion and return to the main menu. Press arrow down to proceed.

```
Disk will be erased, no data  .
recovery will be possible !!! .
Press "enter" to continue     .
Press "bkspc" to return       .
```

Press `bkspc` to cancel the deletion and return to the main menu. Press `enter` to proceed.
The LED blinks red during the erasure. A notification about the erasure progress is displayed.

```
Cipher key deleting           .
in progress...                .
                              .
User erase request            .
```

First, the encryption keys.

```
Data deleting in progress...  .
                              .
                              .
User erase request            .
```

Then, the data stored in memories of the device are deleted.
After erasure information about the device being erased is displayed and the red LED diode lights up.

```
Disk is DELETED               .
                              .
                              .
Disconnect the device         .
```

Subsequently disconnect the device.

Erasing the Disk when Entering the Activation Phrase

Disk may be also erased when exceeding the limit of attempts to enter the activation phrase. The maximum number of attempts (in the range of 1 to 20) is set during the device initialization. If an incorrect phrase has been confirmed one of the following notification appears:

```
Wrong phrase!                 .
Remaining number of tries to  .
enter the passphrase:       4 .
Press "enter" to continue     .
```

or

```
Wrong phrase!                .
The last try to enter        .
the passphrase.              .
Press "enter" to continue    .
```

Press `enter` to re-enter the pass-phrase. The attempts counter does not reset after detaching the device! If an invalid phrase in the last attempt is entered the disk erasure starts. The last line indicates the reason of the disk erasure. During the disk erasure the LED is blinking red and the two following screens appear:

```
Cipher key deleting          .
in progress...               .
                             .
Max. phrase entering attempts .
```

```
Data deleting in progress...  .
                              .
                              .
Max. phrase entering attempts .
```

After the device has been erased the following screen appears:

```
Disk is DELETED              .
                             .
                             .
Disconnect the device        .
```

A user can exceed the maximum number of attempts deliberately to delete the disk.

3.3.5 Device Setup

Use the provided USB cable to attach the disk to a computer. The device turns on automatically. The information about the device appears on the LCD.

```
HECTOR Demo 2 version: 1.00  .
(c) MICRONIC, a.s.           .
Serial nr:  64EB3104 06AA2609 .
Press "enter" to continue    .
```

Press `enter`.
The main menu of an empty device:

```
 Enrollment                  .
>Device setup                .
 Evaluation                  .
                             .
```

The main menu of an enrolled device

```
 Activate the disk              .
>Device setup                   .
 Delete the content             .
                                .
```

Select the `Device setup` option and press `enter`. The first enable to change the language of notifications used by the device.

```
>Language:          ENG         .
                                .
                                .
Screen 1/2        Exit "bkspc".
```

The option `Language` chooses a language of communication with the device. There are two options: English `ENG` and Slovak `SVK`. The second screen sets the way the device responds to pressing a key.

```
>Beeps:             YES         .
 Vibration:         YES         .
 Blue LED blink:    YES         .
Screen 2/2        Exit "bkspc".
```

Option `Beeps` activates the sound when keys are pressed (`YES` turns the beeps on, `NO` turns them off). Option `Vibration` activates vibration when keys are pressed (`YES` turns the vibrations on, `NO` turns them off). Option `Blue LED blink` activates blinking the LED blue when keys are pressed (`YES` means that when a key is pressed the LED blinks, `NO` means it does not).

3.3.6 Usage Recommendations

Usage Recommendations for Demonstrator 2

1. Write down a copy of your passphrase and keep it in a safe place. Write on a hard surface not on a pad of paper.
2. Memorize the pass-phrase making a mnemonic. First, make sure you know what each word means. Look it up if you have to. Then try to make up a story that uses those words.
3. If your passphrase is a recognizable English sentence or phrase generate a new one.
4. Make sure that by any means no other person can watch you entering the passphrase.
5. Check the enclosure of the device for any modifications.
6. Consult any abnormal operation with the manufacturer or its representative.
7. Periodically back up the data stored on the device.
8. When activated, the device allows full access to the data. Use only a trusted computer with appropriate security measures applied against malware and remote access.

9. Performing the operation remove safely is crucial to allow the operating system to finish writing all data to the disk. If the safe removal procedure is not followed and the device is instantly detached from the computer, there is a risk of data damage!

3.3.7 Other Details

Basic Technical Parameters

Cryptographic algorithm	Symmetric cryptographic algorithm ASCON, key length 128 bits
Communication speed	Data reading/writing up to 2.8 MB/sec
Memory size	Depends on the size of the SD card used, maximum 2 TB
Dimensions (L x W x H)	73 x 144 x 8.2 mm
Weight	150 g
Power supply	5 V DC / 170 mA from the USB bus
Working temperature limits	0 to 50°C
Interface	USB 2.0, micro USB connector

Delivery Contents

- Secure portable USB data storage HECTOR Demonstrator 2
- SD card 32 GB
- USB cable

3.4 Conformance to Requirements and Key Performance Indicators

The following table shows the status of Demonstrator 2 requirements defined in D1.2. Here are reported just the requirements which needed to be justified on the final hardware and software and were not validated in the previous deliverable D4.1.

Conformance to Requirements for Demonstrator 2

Requirement	Current status	Remark
Required building blocks		
TRNG block	PLL-TRNG implemented	**Satisfied**
PUF block	TERO-PUF implemented	**Satisfied**
Error correcting algorithm	Majority voting, dark bit selection, Kang's scheme implemented	**Satisfied**

Hash function (for the PUF and passphrase derivation)	not needed	-
Authenticated encryption algorithm	ASCON	**Satisfied**
Functional requirements & properties		
Once powered on, the device requires less than 3 seconds to reach a ready state after which it can be used for its intended purpose	Compliant – less than 650 ms	**Satisfied**
The device offers protection against passive physical attacks (side channel attacks)	Outside the scope	SCA finally turned-out not to be possible in this context
The device offers protection against key-logger attacks on the host PC	Compliant	**Satisfied**
The device reveals no sensitive information once powered off, in particular no offline dictionary attacks is possible on the user passphrase	Compliant	**Satisfied**
The probability of any authentication failure over the system life is lower than 10^{-4}	$P_{\text{fail}} = 10^{-4}$ (influenced by the applied error correcting code and the resulting bit error probability of the PUF)	**Satisfied**
Block level encryption	Compliant	**Satisfied**
The speed at which data can be read from the host PC must be at least 2 MB/s	Compliant	**Satisfied**
The speed at which data can be written by the host PC must be at least 2 MB/s	Compliant	**Satisfied**
The user does not have to install any extra driver	USB Mass Storage Class used	**Satisfied**
The device does not need any external energy source, runs on power from the USB port	Compliant - The device consumes up to 5V / 170mA	**Satisfied**
Additional IVs & MACs is stored in non-volatile memory	Nonces and tags will be stored along with the encrypted user data on the SD card.	**Satisfied**
IV is provided by the TRNG	Nonces will be derived from TRNG and write counter.	**Satisfied**

Requirements on hardware		
Demonstrator 2 includes a display for user interaction	Non-illuminated display included	**Satisfied**
Demonstrator 2 includes a keyboard for user interaction	Capacitive keyboard included	**Satisfied**
Demonstrator 2 includes at least one flash memory device	One removable SD card included	**Satisfied**
Demonstrator 2 limits EM emissions	Aluminium case used	**Satisfied**
Demonstrator 2 provides tamper evidence	Potting with epoxy	**Satisfied**
Demonstrator 2 is powered from a single USB port	Compliant	**Satisfied**
Requirements on the cryptographic primitives		
The PUF requires less power than 20 mW	Compliant – See Tab. 3.2.5	**Satisfied**
The PUF requires less than 5k logic cells to implement	2298 + 185 – See Tab. 3.2.5	**Satisfied**
The failure rate of the PUF over the system life is lower than 10^{-4}	$P_{\text{fail}} = 10^{-4}$	**Satisfied**
TRNG complies with the Class PTG.2 AIS 20/31 requirements	Compliant	**Satisfied**
TRNG achieves an output data rate of at least 10 kbits/s	$>$ 500 kbits/s	**Satisfied**
TRNG block requires less than 100 mW	Compliant – See Tab. 3.2.5	**Satisfied**
TRNG occupies less than 2 k logic cells	351 + 333 – See Tab. 3.2.5	**Satisfied**
The dedicated startup tests require less than 1 second to complete	$<$ 500 ms	**Satisfied**
AE algorithm is protected against side channel analysis	Outside the scope	–
AE algorithm achieves a throughput of at least 2 MB/s	Compliant – See 3.2.6	**Satisfied**
AE algorithm require less than 10 k logic cells to implement	4434 + 1069 – See Tab. 3.2.5	**Satisfied**
AE algorithm is optimized for 512-byte plaintexts	Compliant	**Satisfied**

3.4.1 Key Performance Indicators

Table 3.4.1 shows the main key performance indicators for Demonstrator 2.

Key Performance Indicators for Demonstrator 2	
Usage	European product
Security	High entropy passphrase
Security	Immune to key logger and screen grabbers
Security	Two-factor authentication
Security	No sensitive parameter stored in plain

Chapter 4

Demonstrator 3: Secure Messaging Device

4.1 Motivation

Enabling the ability for users to communicate securely remains today one of the major use cases of cryptography. Demonstrator 3 aims to provide secure messaging devices enabling high level security communications for high end users: military and diplomatic communications for examples.
Demonstrator 3 makes use of the MICRONIC hardware platform in order to demonstrate that the cryptographic primitives developed within HECTOR can successfully be applied in high-end real-world data security applications in which:

- user and device authentication must be strong,
- confidential messages are both encrypted and authenticated.

To this end, Demonstrator 3 makes use of a PLL based TRNG in order to generate a pass-phrase and a TERO PUF combined with the pass-phrase to generate a protection key with enough entropy. Secure communications and encryption make use of ASCON. Use of the PUF allows to prevent from clonability of the device and a strong pass-phrase randomly generated gives insurance on the overall entropy of the generated key.
In addition, existing solutions for sending messages are usually run on top of an operating system on commodity hardware (like a PC, a tablet or a phone). The HECTOR devices provide a solution whose security does not depend on these systems and which are not prone to key-loggers or screen grabbers.

4.2 Description of the Demonstrator Platform

After an initialization phase, two HECTOR devices will be able to setup a secured communication channel over an untrusted link via untrusted host machines as one can see figure 4.1. This system guarantees confidentiality and integrity of the stream of communication data. This means that not only the plaintext messages typed up by the user on the HECTOR display won't be readable or modifiable until they reach the HECTOR device of the recipient but also that no message can be deleted or duplicated without the recipient noticing the tampering attempt. In top of that, the usage of the PUF coupled with user's passphrase can be used as a strong 2 factor authentication protocol that prevents an outsider from communicating with a peer

even if the whole device is compromised. The use of sponge based primitives dispenses from using ad hoc constructions to guarantee not only that messages are confidential and authenticated but also that the whole *stream of messages* hasn't been tampered with (replay-protection, reorder-protection).

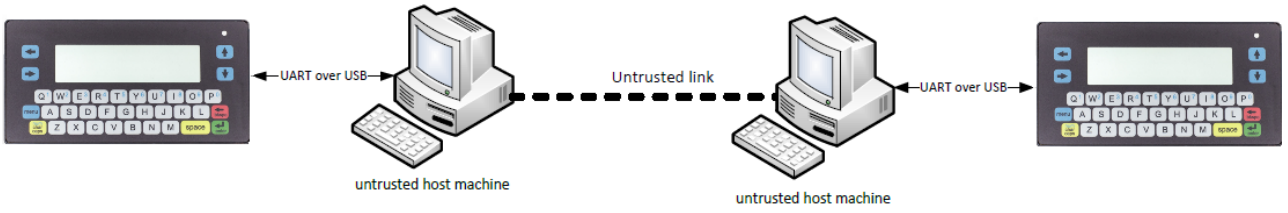

Figure 4.1: HECTOR Demonstrator 3 overview

4.2.1 Hardware Design

The hardware platform of demonstrator 3 is given in Fig. 4.2.

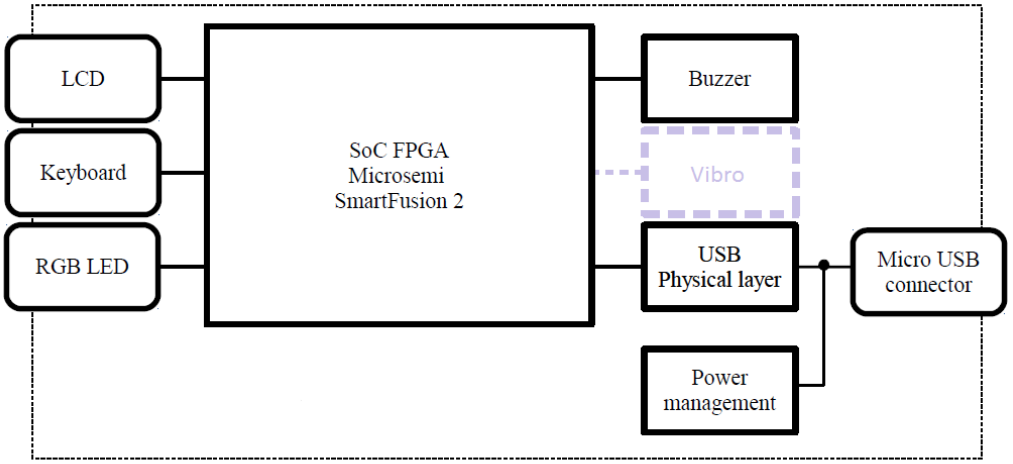

Figure 4.2: Demonstrator 3 hardware platform.

As for Demonstrator 2, the HECTOR Demonstrator 3 integrates a SmartFusion2 SoC FPGA that can be divided into two parts:

- the Microcontroller Subsystem which integrates a Cortex M3 and other sub-systems like SPI, I2C, USB controllers,
- the FPGA fabric which integrates all the cryptographic primitives (PUF, TRNG, and Ascon) which are driven with the software running on the Cortex M3 via an AHB bus.

The software therefore drives the main state machine. The general architecture of the hardware block instantiating the different hardware building blocks is shown in Figure 4.3.

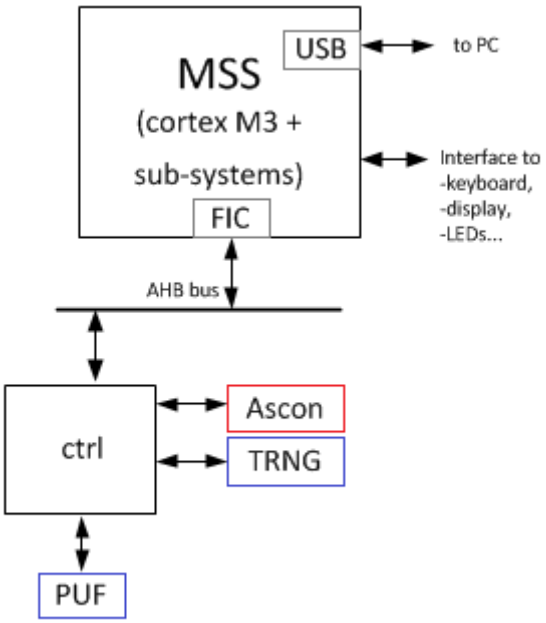

Figure 4.3: Architecture

The interaction between the different hardware blocks and the software running on the Cortex M3 is done by the Control block whose architecture is shown in Figure 4.4.

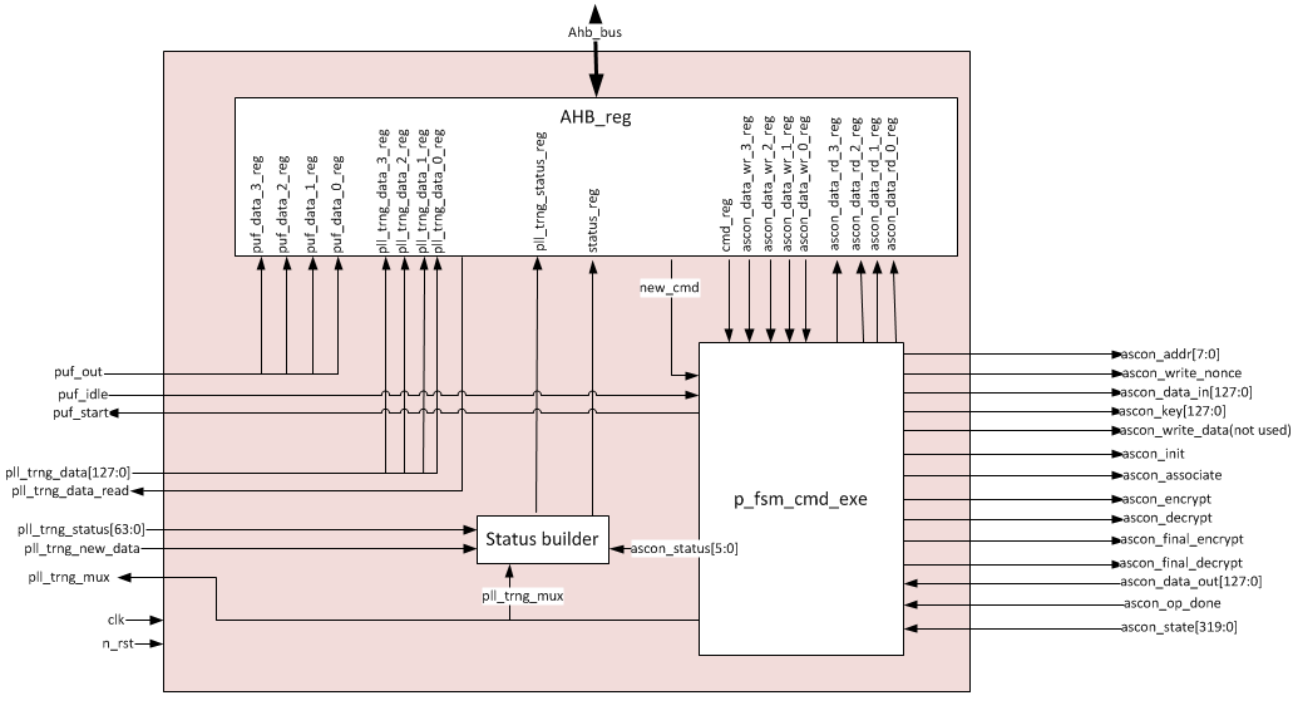

Figure 4.4: Interface of the control block

The control interface communicates with the software through the use of a Command register that activates the different hardware blocks and a Status register that informs the software about the status of the hardware.

Communication with the processor is done via an AHB slave interface. The control module implements a register bank. This register bank is accessible by the processor. All registers are 32 bits wide.

4.2.2 Software Structure

The software package driving the scenario 3 is composed of three modules:

- A python module running on the Secure PC for the initialization phase;
- A python module running on the two Untrusted PCs for the device communication;
- An embedded C module running on the devices that handles the device state machine.

4.2.3 Building Blocks

Non-volatile Memory

The data stored in non volatile memory for usage across device reboots is comprised of several elements:

- **app** : the application binary (read only, start address locked to 0x0);
- – **free** – : a unused memory zone for the possible variation in the application size;
- **data** : the application data (read/write, start address locked to 0x12c00) - initialization flag, helper data, dark bit filter, authentication tags, session ID number,...;
- **dict** : the Diceware dictionary (read only, start address locked to 0x12e00).

The eNVM items over which the designer has controlled are **data** and **dict**. In **data**, the three subitems whose factory settings matter in the scenario are:

- initialization flag : initial value is 0 and becomes 1 when the device has successfully passed the Initialization phase. It can revert back to 0 if the device is cleared;
- Retry counter : initial value is 3 and it is decremented for each unsuccessful authentication attempt. On a successful authentication attempt it is set back to 3;
- Session ID number : initial value is 0 and it is incremented for each session Setup attempt.

The Diceware word list is 7776 words long all shorter than 6 characters. Therefore **dict** is $7776 \cdot 6 = 62208$ bytes long. Each character is ASCII encoded and then padded with 0x0.

PLL-based TRNG with Embedded Tests

As the two-factor authenticaion protocol is the same for both Demonstrator 2 and 3 we refer the reader to section 3.2.2.

TERO PUF and Post-processing

As the two-factor authenticaion protocol is the same for both Demonstrator 2 and 3 we refer the reader to section 3.2.2.

Passphrase, Key Derivation and Retry Counter

A passphrase is a sequence of 8 words picked from the Diceware word list. This passphrase is used as a key to authenticate the PUF associated data[1]. After the PUF reconstruction, it is complemented with the PUF extracted key to form the device key which decrypts and authenticates the communication key.
During the initialization phase, the passphrase is derived from sampling the embedded TRNG on the device. Each call to the TRNG produces 128 bits. Upon the user entry of a passphrase each word is matched to its index in the Diceware word list the indexes are encoded on 13 bits, concatenated and the resulting 117 bits are used as the passphrase key. This transformation clearly does not produce uniform 128 bitstrings but the resulting keys do contain more than the required 96 bits of entropy.
A retry counter is updated each time the PUF reconstruction is initiated and reset to three each time the reconstruction is a success. If more than 3 failures occur in a row the device is reset and a new passphrase must be picked.

Ascon

As in Demonstrator 2, Demonstrator 3 makes use of the AEAD Ascon [1], more precisely of the dubbed "Ascon-128a" version, which is short for $\text{Ascon}_{12,8}$-128-128.
In Demonstrator 3, Ascon is used in 4 ways:

1. Integrity-only mode to protect the authenticity of the helper data;
2. Authenticated encryption of the communication key K_{comm};
3. Authenticated encryption of the `cmd` messages;
4. Stream encryption of the messages exchanged within a session between the peers.

4.2.4 Main State Machine

The main state machine running on the devices is shown on figure 4.5. A brief description of each state is given below.

Initialization The `Initialization` step starts with the first time a device is powered on. It is run only for each device while they are connected to a trusted PC in order to perform key exchange and setting up the device (PUF enrollment, passphrase choice and so on). Once both device have been successfully initialized they are paired and ready to communicate over an untrusted channel.

Authentication The `Authentication` is run each time an initialized `device` is turned. At its end the `device` is on, unlocked and ready to setup a session over an untrusted channel with the `device` it was previously paired with.

LeadSession After a successful `Authentication` a `device` needs to set up a session with its peer by agreeing on a (ever increasing) session number.

[1]The PUF associated data is made of both the Kang scheme helper data and the dark bit filter.

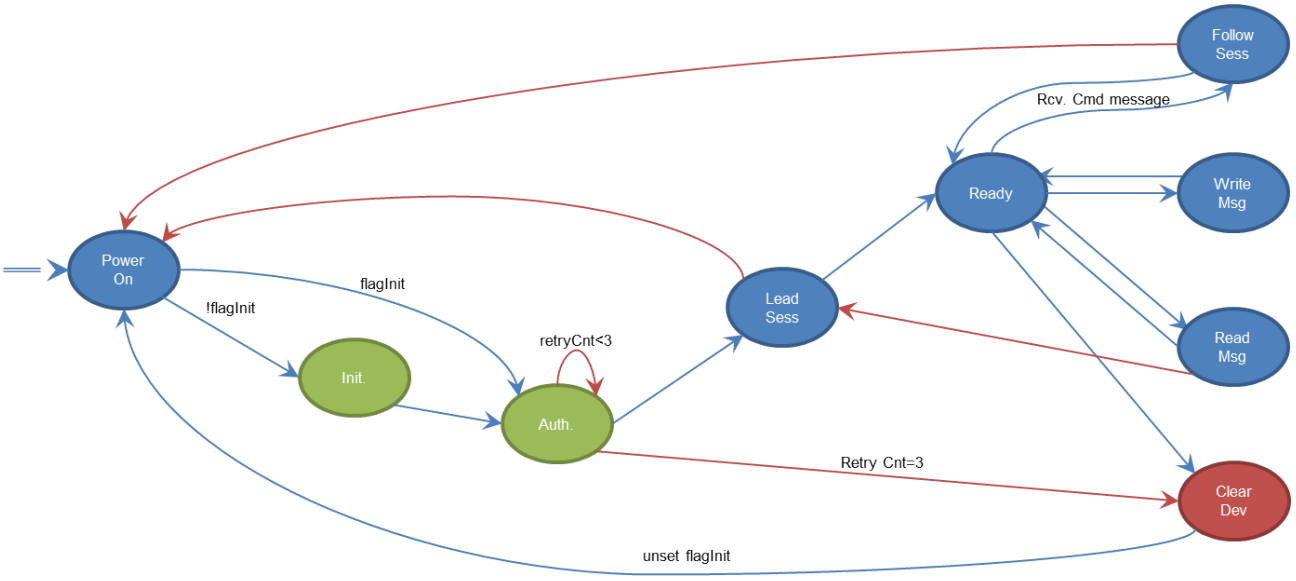

Figure 4.5: Main state machine

Ready After, the establishment of a session the `device` is in a `Ready` state. The `User` can then pick one of the four options:

FollowSession Upon receiving a message of type `cmd` the device will automatically (without user input) proceed to update its internal state and initialize a new stream encryption state.

WriteMessage The length of messages typed up by the `User` is counted in characters and is limited to 160 bytes. To avoid traffic analysis all messages are padded with `0x00` to 160 bytes. It is then wrapped into the current stream encryption state and sent with the associated tag to the peer. If a message has been received it should be read before writing another message otherwise it is going to break the current session.

ReadMessage Upon receiving a message of type `txt` the device will proceed to decrypt and authenticate the message: if it is a valid message then it will be shown to the user, if not then the message will be trashed and the device will be restarted to set up a new stream.

Clear Device This command simply clears the device.

4.2.5 SoC Resources Usage

The final project was analyzed in Libero 11.8 SmartPower tool and the resulting power consumption of the whole project and the particular blocks is displayed in Tab. 4.2.5. The values are computed for typical operational conditions, where the junction temperature is 25°C. The TRNG values include post-processing and PLL TRNG.

Resources used in Demonstrator 3

Resources	FPGA Top	Ascon	Control Block	TERO-PUF	TRNG
4LUT+ Carry chain	7450 (26.9%)	1642	655	2285	2368
DFF	2916 (10.53%)	325	531	178	1678
RAM1K18	11	0	0	0	11
Chip Global	16	0	0	3	2
Row Global	135	0	0	4	8

4.3 Installation and User Manual

The secure messaging device allows to exchange secrets and authenticated messages through an insecure and untrusted channel. It is made in a form of a portable USB disk, like Demonstrator 2. Similarily it has a built-in keyboard and an LCD display which allow to read and enter a pass-phrase and short messages (160 characters at most). Before exchanging messages, two devices must enter in an initialization phase to get their personal secret key which is derived from the generated pass-phrase. Once done, they can be authenticated using the pass-phrase and start a session by constructing a session key which will protect the messages in confidentiality and integrity. The number of wrong authentication is limited using an internal counter which is incremented every time an authentication phase is starting and reset every time an authentication is a success. The disk uses the USB 2.0 interface of the host computer and then must be connected in order to be operated.

4.3.1 Front Panel

Front panel is identical to that of Demonstrator 2. It is the user interface and consists of the LED, the display and the keyboard. The main properties are the following:

LED. The LED diode indicates that the device has successfully complete the different phases by turning green at the end of the phase. When an error occurs (at authentication or when reading a message for example) the diode turns red.

Num char caps. Key **num char caps** switches between the lower-case letters and numerals on the keyboard. The mode is displayed by **(a)** and **(1)** symbols.

Horizontal arrows. Keys to the left from the display allow horizontal movements within the phrase being entered and enable to change the values in menu items.

Vertical arrows Keys to the right from the display allow vertical movements in the phrase being entered and vertical movements in the menu in order to make or confirm choices.

Bkspc. Key **bkspc** deletes a character to the left from the cursor return without carrying out an operation.

Enter. Key **enter** is the confirmation button.

4.3.2 Prerequisites

Demonstrator 3 makes us of:

- one secure PC,
- two untrusted PCs,
- two HECTOR devices and their connections,
- one Ethernet cable to connect the two untrusted PCs.

It also comes with a `Delivery` folder containing three folders:

- `CommunicationPC` which itself contains the python script `comm_Hector.py` to run on the untrusted PCs for communication;
- `SecurePC` which itself contains the python script `SecureInitialization.py` to run on the secure PC for initialization;
- `HectorDevice` which contains the bitstream which is already loaded on the two devices.

4.3.3 Installation

To use the demonstrator, it is necessary to be able to run the python modules on the Secure PC and on the two untrusted PCs - the communication module script must be installed on both untrusted PCs. The disk uses the USB 2.0 interface of the host computer and then must be connected in order to be operated.

4.3.4 Detailed Scenario Description

The two devices are delivered non initialized. It is then necessary to start by an initialization step on both of them.

Initialization

Start by connecting the device to the Secure PC using the provided USB cable. The device turns on automatically.
Initialization is done once on uninitialized devices.

1. **Connect HECTOR device to the Secure PC**

```
************************
* Demonstrator 3       *
* Secure Messaging     *
************************
```

Diode turns green

```
************************
* Initialization       *
* Device               *
************************
```

2. A screen appears with the passphrase consisting of 8 words.

```
/|\ DO NOT FORGET /|\
1:xxxxxx 2:xxxxxx 3:xxxxxx
4:xxxxxx 5:xxxxxx 6:xxxxxx
7:xxxxxx 8:xxxxxx
```

   The passphrase must be memorized. Press `Enter` to proceed.

3. Initialization is processing without intervention from operator. The screen of the device indicates the following steps:

```
   Password generation
   Kang enrollment
   Authenticating PUF data
   Save auth. PUF data
```

   Next screen shows :

```
[x] Derive loc protection key
    Send extract rand to PC
    Master key received
    Encrypt \& auth Master Key
```

4. If the script is not ready to operate, a screen prompt on phase `Send extract rand to PC` asking to:

```
1. Launch PC script
2. Follow indicated steps
[] Press Enter when ready
```

5. **Launch the Python script** `SecureInitialization.py` You might need to change the COM ports.

6. Initialization proceeds until end and diode turns green.

Authentication

1. Connect the two devices to the untrusted PCs. The LED turns blue.

```
************************
* Authenticate         *
* Device               *
************************
```

2. **Enter passphrase** on both devices

```
The expected passphrase is
composed of 8 lower case words
of at most 6 characters each.
```

```
a. 1:        2:
   3:        4:
   5:        6:
   7:        8:       .
```

3. If password is correct, session setup can start and the led turns green.

```
Get ready to start the
start session process
Press Enter when ready
```

4. Press `Enter`

Session Setup

1. Session setup is done right after the authentication phase - **Launch the python script `comm_Hector.py` on both pcs**

2. No intervention from the operator is required on devices - If setup is correctly executed, diode turns green.

```
************************
* StartSession         *
* Successful           *
************************
```

Sending & Receiving Messages

1. Once session setup is done, `main menu` screens appears an diode turns green. Then the following menu is displayed on both devices:

```
(x) Check Inbox (00 new)
    Write message
    Device statistics
        Clear device
```

2. **Menu `Write Message` - To send a message** select `Write message` using the up an down arrow keys and press `Enter` - You can type any type of message you want - Restriction to 160 characters.

3. **Menu `Check Inbox`** - Upon receiving a message `Check Inbox` is updated. **To read a message** select `Read message` using the up an down arrow keys and press `Enter`.

4. **Menu `Device statistics`** - Statistics regarding the device.

5. **Menu `Clear statistics`** - To clear the device and remove all authentication and session data (including keys) select this menu and press `Enter`.

Error Handling

Errors can occur in two cases:

- when a wrong pass-phrase is typed,
- when a new message is send when an unread message is pending in the inbox.

Passphrase errors
If a wrong passphrase is typed a message indicating `Authenticating Failure` is prompted, the led turns red and the device is rebooted. A new authentication phase starts.
In case three wrong passphrases are entered in a row then a message is prompted, the device is rebooted and a new initialization is done. ***New initialization must be done on both devices.***
Message errors
If a user tries to send a message when a new message is pending in the `Check Inbox`, a message prompt on screen. If user enters `Proceed` the message is sent but can not be decrypted by the other device. If an incorrect message is sent to a user (say by an attacker), authentication can not be verified and an error message is displayed on the device screen.
In case of an authentication or decryption error, a new session must be launched: the device which fails restart a session and send a new session message to the other device. Upon reception of this message, the second device will itself engage in a new session.

4.3.5 Usage Recommendations

Usage Recommendations for Demonstrator 3

Disconnection before end of initalization: If you disconnect one of the devices before the end of the initialization, you will have to restart the procedure from beginning on next connection as the pass-phrase will need to be regenerate.

Disconnection of a device: Remember that disconnecting a device shuts it down.

Pass-phrase: Try to make a mnemonic to remember the pass-phrase or keep it in a safe place.

4.4 Conformance to Requirements and Key Performance Indicators

The following table shows the status of Demonstrator 3 requirements defined in D1.2.

Conformance to Requirements for Demonstrator 3

Requirement	Current status	Remark
Required building blocks		
TRNG block	PLL-TRNG	**Satisfied**
PUF block	TERO-PUF	**Satisfied**
Error correcting algorithm	Golay Codes	**Satisfied**

Hash function	-	No hash function required
Authenticated encryption algorithm	ASCON	**Satisfied**
Functional requirements		
"Ready" after 3 seconds	Compliant	**Satisfied**
Protection against side-channel attacks	Out of scope	SCA finally turned-out not to be possible in this applicative scenario
Protection against key-logger on host PC	Compliant	**Satisfied**
Protection against offline password brute-force	Compliant	**Satisfied**
Authentication failure rate $< 10^{-4}$	Compliant - $P_{\text{fail}} = 10^{-4}$	**Satisfied**
160 bytes plaintext	1–160 bytes	**Satisfied**
Doesn't show out of order messages	Compliant	**Satisfied**
Doesn't show replayed messages	Compliant	**Satisfied**
Doesn't show modified messages	Compliant	**Satisfied**
Hardware requirements		
Includes display for user interaction	Non-illuminated display included	**Satisfied**
Includes keyboard for user interaction	Capacitive keyboard included	**Satisfied**
Includes SD-card	Not necessary for demonstrator scenario	–
Limit EM emission	Aluminum case used	**Satisfied**
Provide tamper evidence	Possible potting with epoxy	**Satisfied**
Powered from USB	Compliant	**Satisfied**
Cryptographic requirements		
PUF requires $< 20mW$	Compliant	**Satisfied**
PUF requires $< 5k$ logic cells	Compliant - $2285 + 178$	**Satisfied**
PUF failure rate $< 10^{-4}$	$P_{\text{fail}} = 10^{-4}$ (influenced by the applied error correcting code and the resulting bit error probability of the PUF)	**Satisfied**
TRNG is AIS20/31 PTG.2 compliant	Compliant	**Satisfied**

TRNG output data rate $> 10kbits/s$	$> 500kbits/s$	**Satisfied**
TRNG require $< 100mW$	Compliant	**Satisfied**
TRNG requires $< 2k$ logic cells	$351 + 333 < 1k$	**Satisfied**
TRNG startup tests require < 1 second to complete	$< 500ms$	**Satisfied**
AE is protected against side channel analysis	Out of scope	SCA finally turned-out not to be possible in this architecture
AE output data rate is $> 2Mbits/s$	Compliant	**Satisfied**
AE requires $< 10k$ logic cells	$1642 + 325k$	**Satisfied**
AE is tailored for 1-160 bytes plaintext	Compliant	**Satisfied**

4.4.1 Key Performance Indicators

Table 4.4.1 shows the main key performance indicators for Demonstrator 3.

Key Performance Indicators for Demonstrator 3	
Usage	European product
Usage	Highly sensitive usage
Security	Transfer of encrypted and authenticated messages
Security	Immune to key logger and screen grabbers
Security	Device and user authentication

Chapter 5

Discussion on the Compliance with Recommendations R4 and R5 Given by Reviewers During the Second Review Meeting in October 2016

5.1 Recommendations R4

Recommendation R4 consists of two questions.

5.1.1 R4.1 - LFSR use

Currently, the challenge value is computed by an LFSR generator. Would the utilization of non-linear LFSR (a) reduce the prospects of successful attacks, and (b) would this have implications for modeling and the feasibility to spot that the RNG is under attack?

Answer

We do not use LFSRs in the context of our TRNG design. Therefore we understand your recommendation as a comment to the work on the PUF designs. To describe the situation related to PUFs, some background information is necessary:
PUFs are physical functions and produce a response when queried with a challenge. Responses and challenges are both binary vectors at the highest abstraction level. PUFs are often subdivided in two classes, namely strong and weak PUFs, depending on the number of Challenge-Response Pairs (CRPs). Guajardo et al. define a PUF being strong in [3] if the number of challenge-response pairs is exponentially large. Ideally, when a PUF is challenged, its response is unique. For a given challenge, which is repeated several times, all the responses of the same PUF should be the same, up to a small error. However, for a given challenge, the responses of different PUFs must differ with high probability. Moreover, for a given challenge, the response of a PUF should be unpredictable and uniformly distributed. Due to this property, one use case of PUFs is device authentication. In this setting, particularly strong PUFs are within interest, as the high amount of CRPs prevents an attacker from learning all the responses to all possible challenges having therewith a negligible probability of success. Research work has however shown that strong PUFs are prone to modeling or machine learning attacks. Rührmair et al. show in [14] that all examined strong PUF candidates under a given size are vulnerable to

machine learning attacks with success rates above their in-silicon stability. This contradicts the unpredictability requirement of PUFs, which is required for device authentication. Potential countermeasures against such attacks are indeed increasing the length of the PUF or adding nonlinearity (for example, AND and OR gates [9]). In this case the recommendation would be more than true and vital.
A second use case of a PUF is to use the response a PUF to protect a secret value (e.g. a cryptographic key). Within the HECTOR project, we use the PUF in this setting. We therefore do not work with strong PUFs and their respective assumptions. Our focus is on weak PUFs, where the number of CRPs is rather small. A weak PUF can be understood as a special scheme for non-volatile key storage. Within the HECTOR project we have analyzed two main PUF instantiations, namely the TERO and the RO PUF. Both belong to the group of delay-based PUFs, as the response is influenced by delays of the signals on the IC. The selection of the challenge in our case had purely practical reasons, as we carried out statistical investigations. The randomness of the PUF output is solely depending on the structures on the IC, which are influenced by (non-linear) manufacturing variations. Changing the selection of the challenge does not contribute at all to the randomness of the PUF output and is therefore not relevant for our work on PUFs within the HECTOR project.

5.1.2 R4.2 - Cryptographic Algorithms of the Project

Which of the cryptographic algorithms developed and validated as part of the project (5 of which being candidates in the CESAR competition) will be employed in practice, which of them will be implemented in hardware, and how will the demonstrators make use of these implementations?

The CAESAR competition and HECTOR. The CAESAR competition for authenticated encryption reflects the necessity for a portfolio of efficient and secure symmetric algorithms that protect both the authenticity and confidentiality of data (AEAD). The competition encouraged design approaches dedicated to a wide range of different application scenarios, including (1) lightweight applications, (2) high-performance applications, and (3) defense in depth. The competition started in 2014, one year before the start of the HECTOR project, and the community's efforts in analyzing the 57 submitted designs have lead to the selection of 15 candidates for the current third round of evaluation. Of the 5 CAESAR candidates co-authored by HECTOR partners, 4 are still in the game. The designer teams had to develop hardware (and optimized software) implementations of all these candidates, and had the chance to improve both the algorithms and the implementations based on the public evaluation. We discuss the developments and learnings within the CAESAR competition in more detail in the deliverables of WP5.
Due to the performance requirements identified in the HECTOR project (WP1) and the following evaluation (WP3), 2 candidates in particular emerged as the main focus within the HECTOR project: Ascon and Ketje. Hardware implementations for both were developed, adapted, optimized, and evaluated for the HECTOR project in WP3. In addition, we were able to improve details of the algorithmic design of Ascon. Both appear very well-suited for the lightweight requirements within HECTOR. We refer to WP3 (D3.1, D3.2) and D4.1 for more details on the evaluation and conclusions. Their sponge design was also adapted in WP3 (D3.3) for the cryptographic postprocessing, significantly improving the efficiency of this part.
Based on this evaluation, the HECTOR project selected one AEAD mode to use in its demonstrators: Ascon. Thus, the demonstrators could profit from the concentrated efforts on a single target, both for the security analysis and the implementation efficiency.

The Ascon cipher in the context of HECTOR. Ascon [1] is a family of authenticated encryption designs Ascon-$128_{a,b}$-k-r designed by Dobraunig et al in 2014. In August 2016, Ascon was selected as one of the candidates participating in round 3 of the CAESAR competition [12]. In the framework of Work Package 4 the consortium agreed on the use of a instantiation dubbed "ASCON-128a" which is short for $\text{Ascon}_{12,8}$-128-128.
Demonstrator 2 and Demonstrator 3 make both use of the Microsemi Smartfusion2 SoC, which is a low power, reconfigurable FPGA device with a 32-bit ARM processor and implements Ascon.
In demonstrator 2 we have the following needs:

- all data on the SD card must be encrypted,
- a secure authentication protocol for both the device and the user must be implemented to ensure that no critical data like confidential keys and passwords are saved in the device in plain,
- when at rest, on the move, lost or stolen the data on the removable media must be securely protected by a strong authenticated encryption algorithm.

AEAD mode Ascon is used for all those applications. It is used for the two-factor authentication protocol which allows authentication of the user, the device and the helper data, and the authenticated encryption of the data encryption key. It is also used to encrypt the data on the USB device.
Ascon is used in full confidentiality and integrity mode. The Nonces N and the tags T will be stored along with the encrypted user data on the SD card.
In demonstrator 3, Ascon is used in the initialization step to protect the passphrase on the device. It is then used on each device to protect the communication key. As a consequence it is also used when the two devices are powered on and need to recover the communication key. It is also used to encrypt and decrypt the messages exchanged between the two devices. Ascon is only deployed on the devices and need not to be implemented on the PCs. Ascon is used:

- for integrity only on the helper data,
- for confidentiality and integrity to protect the communication key,
- for stream encryption of the messages exchanged within a session between the peers.

5.2 Recommendation R5

Demonstration scenarios and use should be described in detail and in terms of the key performance indicators assessments defined resp. performed by WP4, not only the individual components, but the overall system implementing the use cases. To this end, WP4 should produce a list of well-defined key performance indicators (KPI) against which the assessment of the use cases can be carried out. Preferably, these KPIs should be applicable for other (existing) solutions, and if ever possible, the results of the HECTOR demonstrators should be compared with and benchmarked against such solutions.

The three demonstrators aim to demonstrate that the provided designs and design approaches of selected cryptographic primitives (TRNG, PUF, AE) are suitable to cover three different problems:

- Demonstrator 1 as a high performance TRNG can be used as a **stand-alone security peripheral** achieving high security level and high output bit rate,
- Demonstrator 2 is a portable device aimed at **protecting data at rest** as a secure USB stick,
- demonstrator 3's goal is to **protect data in motion** with a secure messaging device.

However, it is important to note that the primary purpose of the demonstrators is not to provide a single finished product, but to demonstrate how the innovative components developed and optimized within WP2 and WP3 of the HECTOR project can be integrated to a fully functional system to solve practical problems.
We describe the demonstration scenarios and uses in detail in the motivation part of each chapter: Section 2.1 for Demonstrator 1, Section 3.1 for Demonstrator 2 and Section 4.1 for Demonstrator 3.
Key Performance Indicators are also described in the Conformance to Requirements and Key Performance Indicators section of each chapter: Section 2.4.1 for Demonstrator 1, Section 3.4.1 for Demonstrator 2 and Section 4.4.1 for Demonstrator 3.

Chapter 6

Summary and Conclusion

This deliverable accompanies the three HECTOR demonstrators developed in T4.1 and T4.2 in order to provide a complete understanding of their design and their use. This work advances Deliverable 4.1 where software and hardware specifications of the demonstrators platform were described.
The primary purpose of the demonstrators is not to provide a single finished product, but to demonstrate how the innovative components developed and optimized within WP2 and WP3 of the HECTOR project (TRNG, PUF, AE) can be integrated to a fully functional system to solve practical problems. The three demonstrators we described are suitable to cover three different problems.
Demonstrator 1 as a high performance TRNG can be used as a **stand-alone security peripheral** achieving high security level and high output bit rate. It is compliant with the major recommendations (AIS20.31, NIST, DGA) and we demonstrated that its bitrate is considerably higher than existing solutions. Also, all the requirements listed in D4.1 are satisfied.
Demonstrator 2 is a portable device aimed at **protecting data at rest** as a secure USB stick. It is the first European solution to provide such a device with a high level of security. It is immune to key logger, screen grabber and clonability and no sensitive data are stored in plain. With this demonstrator we showed that, in the lost and found scenario, HECTOR components can be successfully applied. The list of requirements from D4.1 for this demonstrator is also achieved.
Demonstrator 3's goal is to **protect data in motion** with a secure messaging device. This is also the first European solution to provide such a high secure device to exchange messages when communications are really sensible, as in military or diplomatic usages. This messenger is also not prone to key logger and screen grabbers and its two-factors authentication prevent from lost and found and clonability issues.
The hardware modules integrated in the three demonstrators come from output of WP2 and WP3, therefore the integration of the three demonstrators benefited a lot from the collaboration of the majority of the partners.
This work will continue with T4.4 and the security evaluation and testing of the demonstrators.

Chapter 7

List of Abbreviations

List of Abbreviations

AEAD	Authenticated Encryption with Associated Data
AE	Authenticated Encryption
AES	Advanced Encryption Standard
ATX	Advanced Technology eXtended
COM	Communication
CRP	Challenge-Response Pair
DC	Delay Chain
DFF	D Flip-Flop
DMA	Direct Memory Access
DRNG	Deterministic Random Number Generator
eNVM	Embedded Non-Volatile Memory
GUI	Graphical User Interface
FIPS	Federal information processing standard
FPGA	Field Programmable Gate Array
HDA	Helper data algorithm
IP	Intellectual property
IV	Initialization Vector
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MSS	Microcontroller sub-system
PC	Personal Computer
PLL	Phase-Locked Loop
PTRNG	Physical True Random Number Generator
PUF	Physically Unclonable Function
RAM	Random Access Memory
RNG	Random Number Generator

RO	Ring Oscillator
SD card	Secure Digital card
SF2	Microsemi SmartFusion2
SoC	System on Chip
TERO	Transient Effect Ring Oscillator
TOE	Target Of Evaluation
TRNG	True Random Number Generator
USB	Universal Serial Bus
UART	Universal Asynchronous Receiver/Transmitter
VCP	Virtual Communication Port

Bibliography

[1] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schlaffer. Ascon v1.2, Submission to the CAESAR Competition. Technical report, Institute for Applied Information Processing and Communications Graz University of Technology, Infineon Technologies Austria AG, September 2016.

[2] Gross, H. CAESAR Hardware API reference implementations. Technical report, Institute for Applied Information Processing and Communications Graz University of Technology, June 2016.

[3] Guajardo, J., Kumar, S.S., Schrijen, G.J., Tuyls, P. FPGA intrinsic PUFs and their use for IP protection.

[4] Rémi Audebert JeanMichel Picot and Elie Bursztein. Attacking encrypted usb keys the hard(ware) way. *Black Hat USA*.

[5] Hyunho Kang, Yohei Hori, Toshihiro Katashita, Manabu Hagiwara, and Keiichi Iwamura. Cryptographie key generation from puf data using efficient fuzzy extractors. In *Advanced Communication Technology (ICACT), 2014 16th International Conference on*, pages 23–26. IEEE, 2014.

[6] W. Killmann and W. Schindler. A proposal for: Functionality classes for random number generators. AIS 20 / AIS31. Technical report, 2011.

[7] Lattacher, S., Deutschmann, M., Rožić, V., Yang, B., Singelée, D., Fischer, V., Mureddu, U., Anzala-Yamajako, A., Melzani, F., Kleja, M., Laban, M., Eichlseder, M., van Battum, G., Wakker, M. D4.1 Demonstrator Specification. Technical report, TEC, KUL, UJM, TCS, STI, MIC, TUG, BRT, August 2017.

[8] Lattacher, S., Lipnik, G., Deutschmann, M., Iriškić, L. TERO PUF Evaluation. Technical report, Technikon, August 2017.

[9] Daihyun Lim. Extracting secret keys from integrated circuits.

[10] National Institute of Standards and Technology. FIPS 140-1: Security Requirements for Cryptographic Modules, 1994.

[11] Alin Suciu and Tudor Carean. Benchmarking the true random number generator of TPM chips. *Computer Research Repository (CoRR)*, abs/1008.2223, 2010.

[12] The CAESAR committee. CAESAR: Competition for authenticated encryption: Security, applicability, and robustness. http://competitions.cr.yp.to/, 2014.

[13] Meltem Snmez Turan, Elaine Barker, John Kelsey, Kerry A. McKay, Mary L. Baish, and Mike Boyle. Recommendation for the entropy sources used for random bit generation. NIST Special Publication 800-90B (second draft), 2016.

[14] Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. Modeling attacks on physical unclonable functions.