Securing IoT Applications with mbed TLS

Hannes Tschofenig

Part#3: Random Number Generators (RNGs)

July 2018
Munich

Why should we talk about RNGs?

- The importance of random numbers for security protocols was already discussed in the first webinar.
- As a reminder, watch Paul Bakker talk about "Entropy Requirements in IoT" on the Arm mbed Youtube channel.
- Example code so far did not use an RNG because of the following "development only" settings in the mbedTLS_config.h file:
  - `#define MBEDTLS_NO_DEFAULT_ENTROPY_SOURCES`
  - `#define MBEDTLS_TEST_NULL_ENTROPY`
- A compile-time warning is the result:
  `THIS BUILD IS *NOT* SUITABLE FOR PRODUCTION USE`
- We will enhance our code to make use of a hardware-based random number generator.

arm

Our Hardware

arm

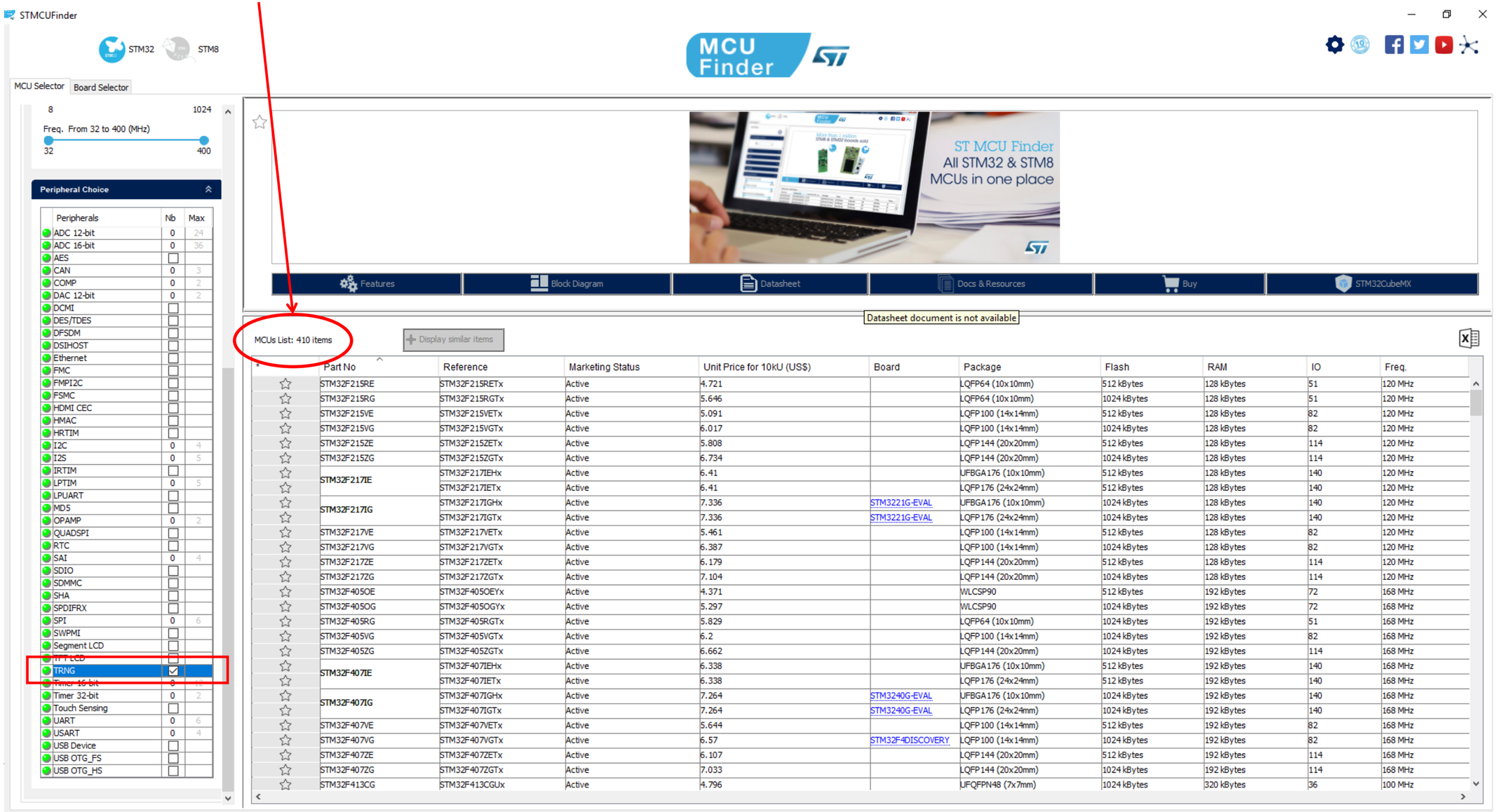
Keil Evaluation Board

- We are still using the Keil MCBSTM32F400 Evaluation Board, which uses the STM32F407IG MCU.
- This MCU uses an Arm Cortex M4 processor and offers a hardware-based random number generator.

arm

RNG Availability?

- ST alone offers 410 MCUs with RNG support. This is only ST and other vendors have similar offers.



Accessing the RNG

arm

STM32F407IG RNG

- The STM32F407IG product webpage contains pointers to reference manuals, including RM0090.
- Section 24 describes the RNG functionality. In a nutshell:
  - The RNG delivers a 32-bit random number with every access.
  - The peripheral is attached to the AHB2 bus and is memory mapped.
  - The RNG can be operated in two modes; in an Interrupt-driven mode and in a polling mode.
  - There are three registers dedicated to the use with the RNG, namely (1) a control register (CR), a status register (SR), and a data register (DR).

RNG Register Map

Offset	Register name reset value	Register size																													
		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	RNG_CR 0x0000000	Reserved																												IE	RNGEN	Reserved	
0x04	RNG_SR 0x0000000	Reserved																									SEIS	CEIS	Reserved		SECS	CECS	DRDY
0x08	RNG_DR 0x0000000	RNDATA[31:0]																															

arm

Steps for accessing the RNG

Enable RNG peripheral via RNGEN bit in the RNG_CR register

RNG working

Check for errors and random number availability

Perform three checks via the RNG_SR register:

1. Is there a seed error? (SEIS bit)
2. Is there a clock error? (CEIS bit)
3. Is a random number ready? (DRDY bit)

No error and data is ready

Read the random number

Note: The first random number generated after setting the RNGEN bit should not be used, according to FIPS (Federal Information Processing Standard Publication) PUB 140-2.

Note: Each generated random number has to be compared with the previously generated number. The test fails if any two compared numbers are equal.

arm

Hands-On:
Testing the RNG Hardware

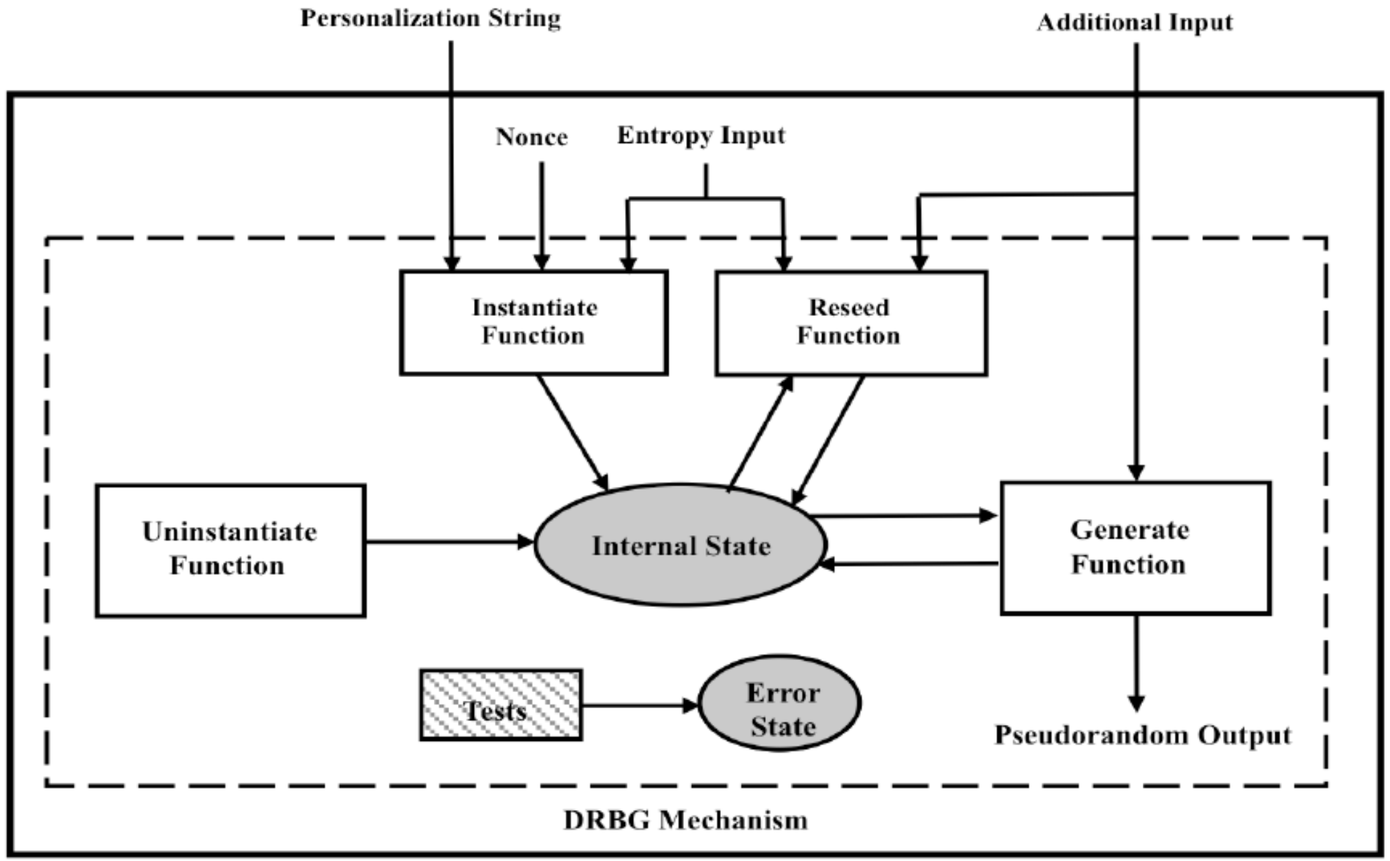
arm

NIST SP 800-90A rev 1

arm

Deterministic random bit generator (DRBG)

- NIST SP 800-90A rev1 specifies the generation of random bits using deterministic methods based on either hash functions or block cipher algorithms.



DRBG Functional Model

Mbed TLS Entropy & DRBG API

Entropy accumulator implementation

// Entropy context structure
mbedtls_entropy_context entropy;

// Entropy context initialization
mbedtls_entropy_init(…)

// Adds an entropy source to poll
mbedtls_entropy_add_source(…)

Add our hardware-based RNG

// Free entropy context
mbedtls_entropy_free(…)

DRBG Mechanism Based on Block Ciphers

// CTR_DRBG context structure
mbedtls_ctr_drbg_context ctr_drbg;

// Initializes the CTR_DRBG context
mbedtls_ctr_drbg_init(…)

// Seed the CTR_DRBG
mbedtls_ctr_drbg_seed(…)

Link with entropy context

// Turns prediction resistance on or off
mbedtls_ctr_drbg_set_prediction_resistance(…)

// Clears CTR_CRBG context
mbedtls_ctr_drbg_free(…)

// Configure Mbed TLS with RNG callback
mbedtls_ssl_conf_rng(…)

Link with CTR_DRBG context

arm

Hands-On: Integrating the RNG functionality

arm

Summary

- Random numbers are important for security.
- Pick the appropriate hardware for your task. For our security application we need a hardware-based random number generator.
- Use your favorite MCU and follow the steps to integrate your RNG into mbed TLS.