An Evaluation of the Multi-Platform Efficiency of Lightweight Cryptographic Permutations

Luan Cardoso dos Santos[1] and Johann Großschädl
SnT and DCS, University of Luxembourg

[1]supported by the Luxembourg National Research Fund through grant PRIDE15/10621687/SPsquared.

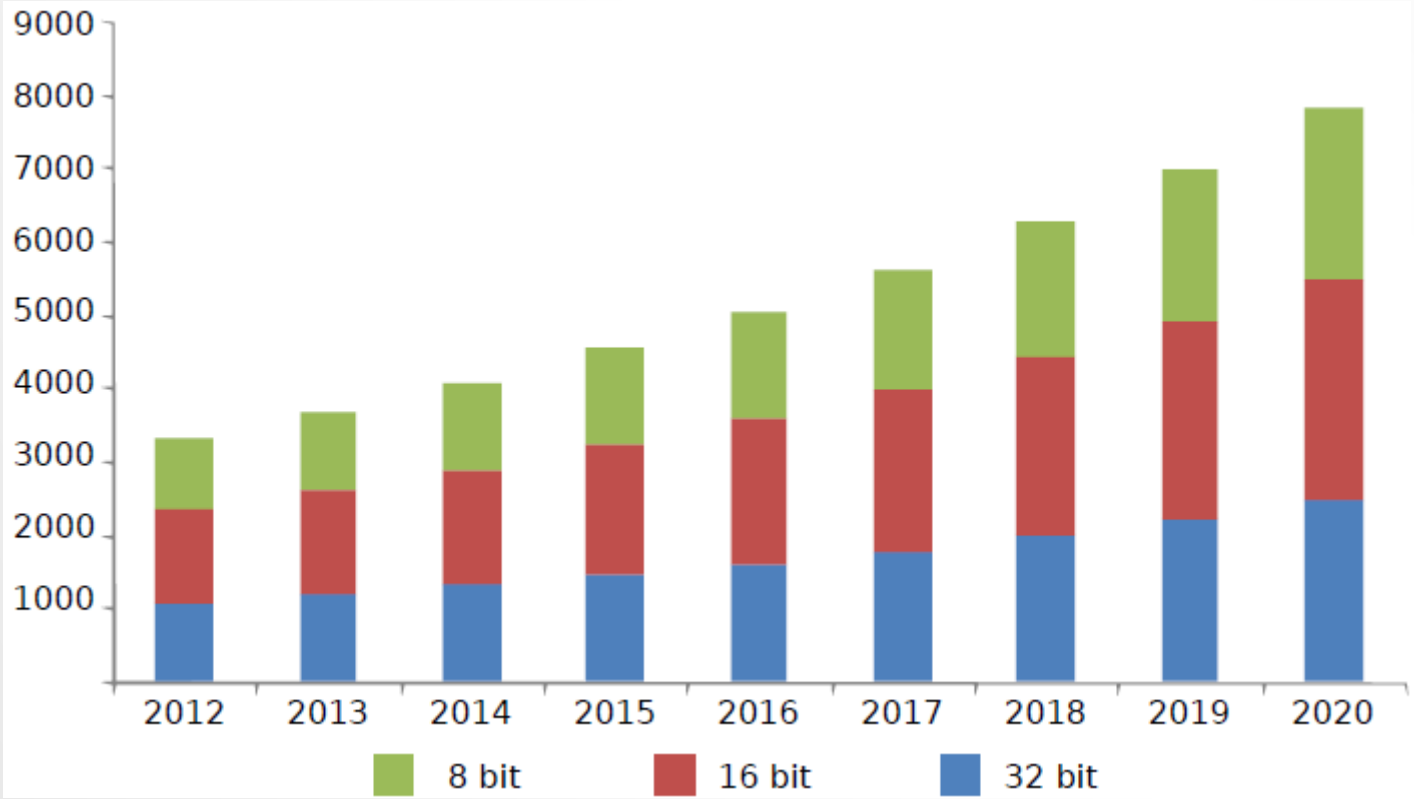
Overview

- Permutations of 4 AEAD Algorithms
  - Ascon128a, Gimli, Schwaemm256-128, Xoodyak
  - Similar characteristics (state size, simple operations)
  - Efficient in software
- Highly-Optimized Assembler Implementations
  - Speed-optimized (fully unrolled) for 32-bit ARM Cortex-M3
  - Size-optimized for 8-bit AVR Atmega128
- Simulated and Measured Execution Times
  - Simulation: Keil Microvision 5.24, Atmel Studio 7.0
  - Measurements: Cortex-M3 boards with 0, 2, 5 flash wait states

Motivation

- Why Multi-Platform Efficiency?
  - IoT is populated by highly diverse and heterogeneous devices
  - No single dominating platform!
  - Not only “smart” but also “dumb” devices (e.g. sensor nodes)
  - 8/16/32-bit microcontrollers have very different characteristics
  - Register space: 56 bytes on ARM, 32 bytes on AVR
  - Multi-bit rotations: cheap or “free” on ARM, costly on AVR
- Why Only Permutations?
  - Reference performance/size figures for designers
  - Throughput of AEAD algorithms is bounded by throughput (in cycles per rate-byte) of underlying permutation!

8/16/32-bit Microcontroller Market



North American microcontroller market by product (8-bit, 16-bit, 32-bit) in million units
source: https://www.radiantinsights.com/research/microcontroller-market

Ascon

- Part of CAESAR's final portfolio
- Main components of the Ascon128a AEAD are two 320-bit permutations: p^a (12 rounds) and p^b $(8\ \mathrm{rounds})$
- State is organized in five 64-bit words
- Substitution layer (bit-sliced 5-bit Sbox): AND, NOT, XOR
- Linear diffusion layer: XOR, rotations of 64-bit words
- Rate used by Ascon128a AEAD for encryption: 16 bytes

Gimli

- Gimli AEAD targets 256-bit security level!
- A 384-bit permutation, designed to achieve high performance on a wide range of hardware and software platforms
- State organized as 3x4 matrix of 32-bit words
- Main components: non-linear 96-bit SP-box, linear mixing layer, constant addition (every 4^{th} round)
- Rate used by Gimli AEAD for encryption: 16 bytes

Sparkle384

- 384-bit permutation, used by the main instance of Schwaemm AEAD and Esch hash function
- Smaller and larger variants exist: Sparkle256, Sparkle512
- Classical ARX design, operations on 32-bit words
- Six instances of a 64-bit ARX-box (Alzette) and a linear diffusion layer per step
- Rotation distances suitable for 8/16-bit microcontrollers
- Rate used by Schwaemm256-128 for encryption: 32 bytes

Xoodoo

- 384-bit permutation for efficient symmetric crypto (e.g. Xoodyak, Xoofff) on a wide range of platforms
- State is organized in 3 horizontal planes, each one consisting of 4 parallel 32-bit lanes
- Round function consists of five components: two plane shifts, a mixing layer, a constant addition, and a non-linear layer
- Rate used by Xoodyak for encryption: 24 bytes

Results 32-bit ARM Cortex-M3

Permutation	Code size bytes	Exec. Time clock cycles	Throughput cc/rate-byte
Ascon128a (8 rounds)	1928	466	29.13
Gimli (24 rounds)	3950	1041	65.06
Sparkle384 (7 steps)	2820	781	24.40
Xoodoo (12 rounds)	2376	627	27.38

Code size, execution time, and throughput (in cycles per rate-byte) of speed-optimized ARMv7-M assembly implementations of the four permutations. The execution times were determined using the cycle-accurate instruction set simulator of Keil MicroVision 5.24 using a generic Cortex-M3 as target device.
The implementations of Ascon, Gimli, and Xoodoo are from the designers; Sparkle384 was implemented by us (not yet publicly available).

Results 8-bit AVR ATmega128

Permutation	Code size bytes	Exec. Time clock cycles	Throughput cc/rate-byte
Ascon128a (8 rounds)	898	6442	402.63
Gimli (24 rounds)	778	23699	1481.19
Sparkle384 (7 steps)	702	8318	259.94
Xoodoo (12 rounds)	954	13091	545.46

Code size, execution time, and throughput (in cycles per rate-byte) of size-optimized AVR assembly implementations of the four permutations. The execution times were determined using the cycle-accurate instruction set simulator of Atmel Studio 7.0 using an ATmega128 as target device.
The implementations of Gimli is from the designers; Ascon128a, Sparkle384, and Xoodoo were implemented by us (not yet publicly available).

Simulation vs Reality

- Keil Simulator is not 100% Cycle-Accurate
  - Simulator assumes "ideal" conditions for memory accesses
  - External components like flash not cycle-accurate!
- Flash Wait States
  - Cortex-M microcontrollers can have frequencies above 200 MHz
  - Flash is usually clocked at much lower frequencies (20-30 MHz)
- Wait States Slow Down Execution Time
  - Number of wait states depends on frequency of core
  - Some devices have a "flash accelerator", i.e. a buffer between microcontroller core and flash memory

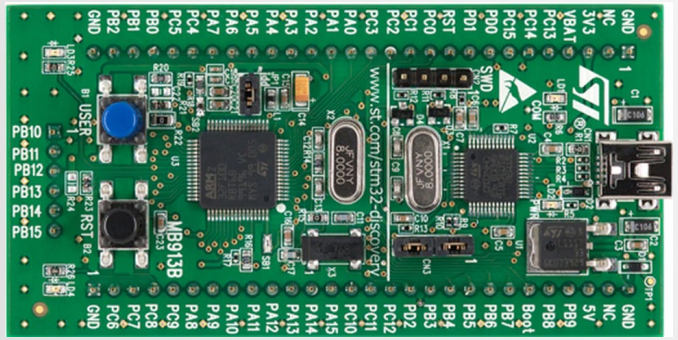
Cortex-M3 Boards

STM32VL Discovery

STM32F100RBT6B Cortex-M3

Nominal core freq: 24 MHz

No flash wait states

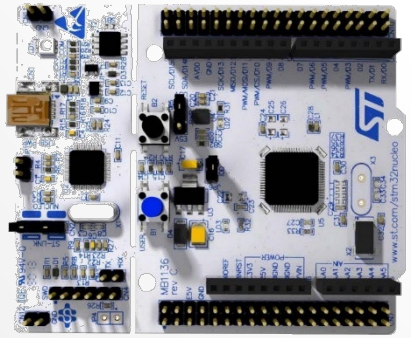


STM32 Nucleo-64

STM32F103RBT6 Cortex-M3

Nominal core freq: 72 MHz

2 flash wait states



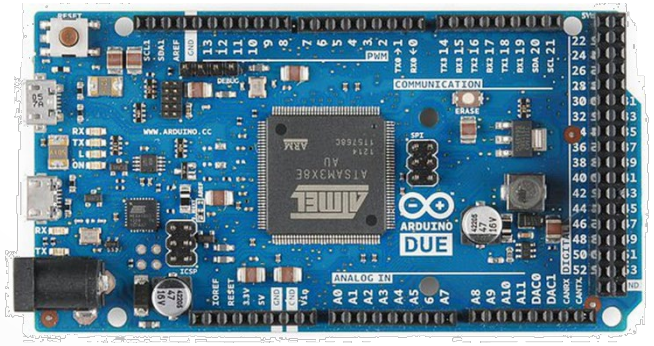
Arduino Due

Atmel SAM3X8E Cortex-M3

Nominal core freq: 84 MHz

5 flash wait states

"Flash accelerator"



Results on ARM Cortex-M3 Boards

Permutation	Keil μVision simulation	VL Discovery 0 wait states	Nucleo-64 2 wait states	Arduino Due 5 wait states
Ascon128a (8 rounds)	466	467	748 (1.60)	571 (1.22)
Gimli (24 rounds)	1041	1043	1656 (1.59)	1287 (1.23)
Sparkle384 (7 steps)	781	782	1196 (1.53)	936 (1.20)
Xoodoo (12 rounds)	657	659	1014 (1.54)	795 (1.21)

Execution time of the four permutations determined by simulation with Keil MicroVision using a generic Cortex-M3 model and measurement on CortexM3 development boards with 0, 2, and 5 flash wait states (values in parentheses are the performance penalties versus the execution time on the VL Discovery, which has 0 flash wait states).
When the Nucleo and the Arduino are clocked down to around 20 MHz and configured to have 0 wait states then the execution times are very similar to that of the VL Discovery.

Gimli Versus the Others

- Gimli AEAD (and permutation) aims for 256-bit Security
  - Ascon128a, Schwaemm256-128 and Xoodyak designed for 128-bit security
  - Comparison not really fair!
- Schwaemm256-256
  - Designed for 256-bit security
  - Uses Sparkle512 permutation (8 steps)
  - 512-bit state does not fit in register file of ARM
  - Rate used by Schwaemm256-256 for encryption: 32 bytes

Results for 256-bit Security-Level

Permutation on ARM	Code size bytes	Exec. Time clock cycles	Throughput cc/rate-byte
Gimli (24 rounds)	3950	1041	65.06
Sparkle512 (8 steps)	4464	1314	41.06

Permutation on AVR	Code size bytes	Exec. Time clock cycles	Throughput cc/rate-byte
Gimli (24 rounds)	778	23699	1481.19
Sparkle512 (8 steps)	702	12454	389.19

Conclusions

- Multi-Platform Efficiency is Important
  - No single dominant platform in the IoT
  - 8/16/32-bit microcontrollers have very different characteristics
- Throughput of Ascon, Sparkle Xoodoo similar on ARM
  - Between 24 and 29 cycles per rate-byte
- Throughput differs significantly on AVR
  - Sparkle outperforms Ascon and Xoodoo by a factor of 1.55 and 2.10, respectively
  - Ascon: rotation-distances not ideal for 8/16-bit microcontrollers
  - Xoodoo: register allocation very challenging, difficult to optimize