Alpha Improvements

Ideas for discussions

F2F Amsterdam 2019-02-19 – 2019-02-21

Joachim Strömbergson

ASSURED

SECURITY CONSULTANTS

Setting / Disclaimer

A collection of ideas for improvements to CrypTech Alpha board

- Performance improvements
- Security improvements

- Divided into changes that can be applied to
  - All Alpha boards – current and future
  - Alpha boards manufactured in the future with the same PCB design
  - A newly designed Alpha board (Alpha NG, v2, Beta...)

- Some ideas are more speculative/wrong than others.

SW and FPGA changes

Can be deployed on current and future Alpha boards

SW and FPGA Changes 1

Possible Performance Improvements

- Pushing to meet 90 MHz, 100+ MHz clock speed for processing cores
  - Use floor planning to better control the routing
    - We have releng releases where the core configuration not changing all the time. We could create floorplan(s) for a stable configuration.

- Hunt latency cycles for 32-bit accesses over FMC bus
  - Move endianess fix to FPGA
  - Collapse function calls, optimize address calculation in SW for read/write
  - Look at possible HW support for auto increment of addresses
    - Minimalistic DMA
  - Use fine grained cycle counters in STM32 to measure access times

Paul has done things here

SW and FPGA Changes 2

Possible Performance Improvements (cont)

- Move SW functions to the FPGA
  - CRT has been identified as prime candidate
    - Dedicated RISC-V core with CRT code in BlockRAM as ROM
      - Can be clocked higher than other cores
    - Add HW accelerator(s) based on profiling

Probably moot after Pavels work

  - Other code that is tightly coupled to the FPGA as part of processing
    - Similar behavior to the AES Key Wrap

RISC-V Cores – Size, Performance

Looked at three cores

- PicoRV32
  - ISC License (similar to MIT)
  - Scalable from very small to something like ARM Cortex-M3
    - Focused on compact implementation above performance
  - No support for protection modes
  - 200 – 400 MHz in Artix-7
  - https://github.com/cliffordwolf/picorv32

Core Variant	Slice LUTs	LUTs as Memory	Slice Registers
PicoRV32 (small)	761	48	442
PicoRV32 (regular)	917	48	583
PicoRV32 (large)	2019	88	1085

RISC-V Cores – Size, Performance

Looked at three cores

- VexRisc
  - MIT-license
  - Scalable from very small to something like ARM Cortex-M4
  - Good DMIPS/MHz
  - Smallest: 481 LUTs, 539 regs, 346 MHz in Artix-7
  - Biggest: 1813 LUTs, 1424 regs, 183 MHz in Artix-7
  - https://github.com/SpinalHDL/VexRiscv

RISC-V Cores – Size, Performance

Looked at three cores

- Western Digital SweRV
  - Apache 2.0 license
  - 9 stage, dual issue superscalar (compared to ARM A15)
  - 30074 LUTS, 14131 regs, 8 RAMB36, 28 RAMB18, 4 DSP48
  - 200+ MHz
  - https://github.com/westerndigitalcorporation/swerv_eh1

More powerful than STM32
Less kitchen sinks

RISC-V Cores – Comparison

Resource usage today in Releng

- 32k of 269k regs (12%)
- 37k of 134k slice LUTs (27%)
- 10k of 33k slices (30%)
- 16 of 366 RAMB36E (4%)
- 109 of 730 RAMB18E (14%)
- 140 of 740 DSP48 (18%)

SW and FPGA Changes 3

Add memory buffers (as cores) to allow SW to store temporary data in the FPGA between operations

- Would probably affect specific cores

Add DMA master that can perform data movements within the FPGA on command by the CPU

- Between processing/utility cores (mkmif to keywrap)
- Between processing cores to buffer cores
- Single, low digit cycles/word. Better than bouncing via FMC bus
- Also reduces the amount of sensitive data in the STM32
- Need to understand access patterns and use cases

Suggesting a break out session to discuss this

Add just read/write buffers with address generators

- A restricted/simplified version of the DMA idea
- Allow decoupling of reading/writing blocks of data (RSA keys for example)

SW and FPGA Changes 4

Possible Security Improvements

- Based on Pauls comments
  - Add support in cores for quick zeroisation of sensitive data

  - We have wires between AVR and FPGA to signal alarms etc
    - We need to connect and use them

Minor Board Changes

Can be introduced for future Alphas with no board redesign

BoM Changes

Change components when manufacturing more Alphas

- Use FPGAs with faster speed grades (-2, -3 instead of -1)
- Artix-7, speed grade -1 costs 193 USD
- Speed grade -2 costs 222 USD
- Speed grade -3 costs 293 USD
- 10-20 % better timing for each speed grade
- Allow us to reach higher core clock speeds
  - Cloced synced with FMC don’t allow for many possible realistic clocks speeds

Change MKM to quad SPI

Need to be mounted on headers and use GPIOs

- MKM read speed used to seem to take significant time
- Quad SPI would cut latency by 4
- Can in fact be mounted on existing Alphas too

- We haven’t really increased SPI clock speed on the existing MKM
- We are implementing a FPGA MKM where we can decide on interface to use

Alpha V2/NG/Beta

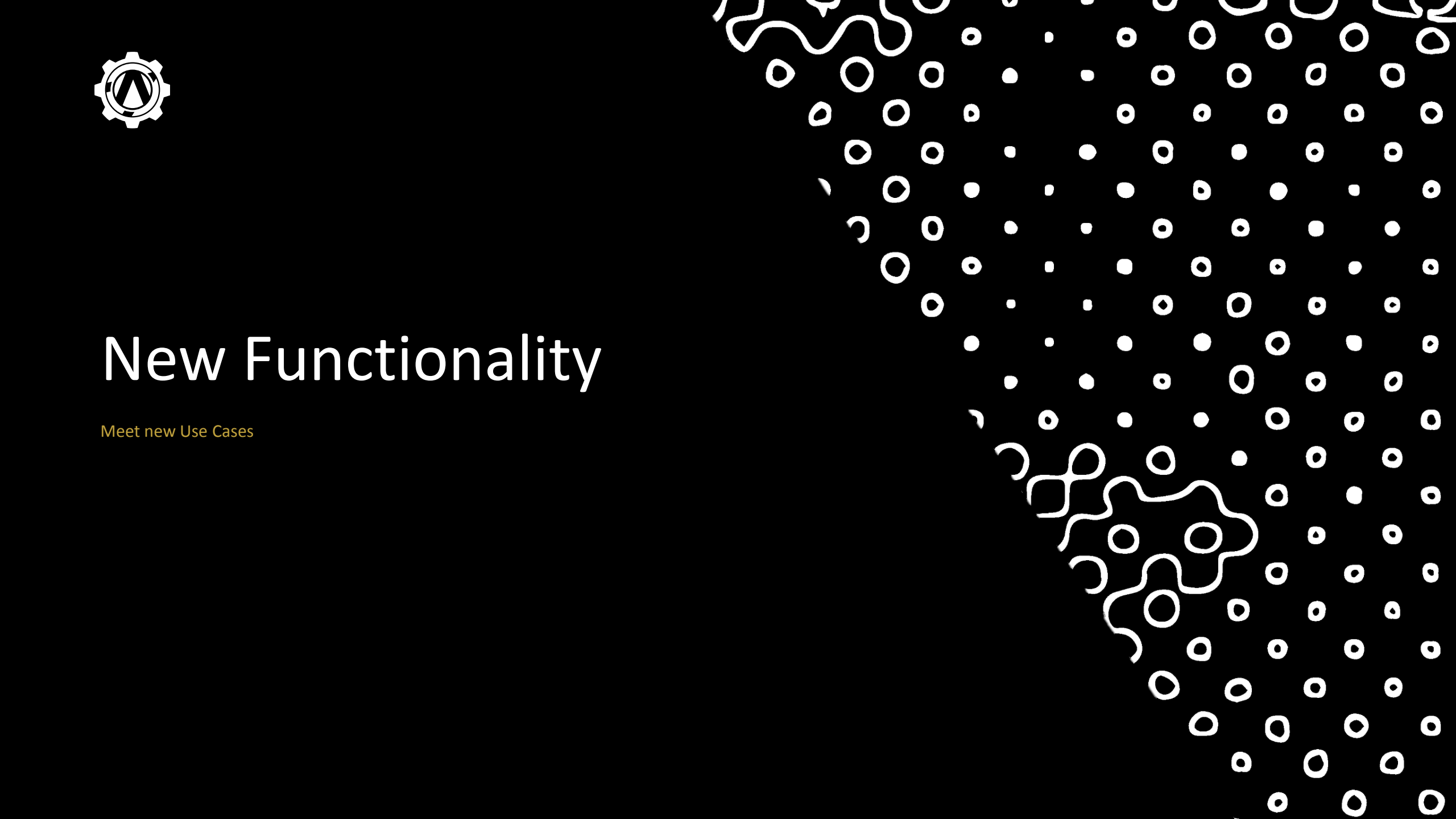
Major redesign to create the next Cryptech HSM

Next Alpha

Replacing the current FPGA

- Using a bigger FPGA and add WD RISC-V cores for processing
  - No need for the STM32
  - We are using the biggest Artix-7 (200T)
  - The Kintex-7 325T is the next biggest chip
    - 934 USD (5x more expensive)
    - Will have to move to Xilinx Vivado

- Use a FPGA with hard ARM cores
  - Single or dual 800 MHz ARM-A53 cores
  - Tightly coupled interface between CPU and FPGA (much less cycles)
  - Wider interfaces
  - DMA support
  - Xilinx UltraScale+ ZU6CG has almost 2x the FPGA resouces, dual cores
    - 1904 USD (10x compared to current FPGA. But no STM32)



New Functionality

Meet new Use Cases

Some New Functionality (1)

- Message Authentication Codes
  - CMAC (Completed. Can be donated to the project)
  - HMAC-SHA256. We have SHA256. HMAC simple to add

  - Poly1305
    - Very interesting core. No HW implementations I know of
    - Would allow CrypTech to provide ChaCha20-Poly1305 in HW
      - Provide HW implementation of RFC 8439
      - Used in TLS 1.3, WireGuard etc

https://tools.ietf.org/html/rfc8439

ASSURED

SECURITY CONSULTANTS