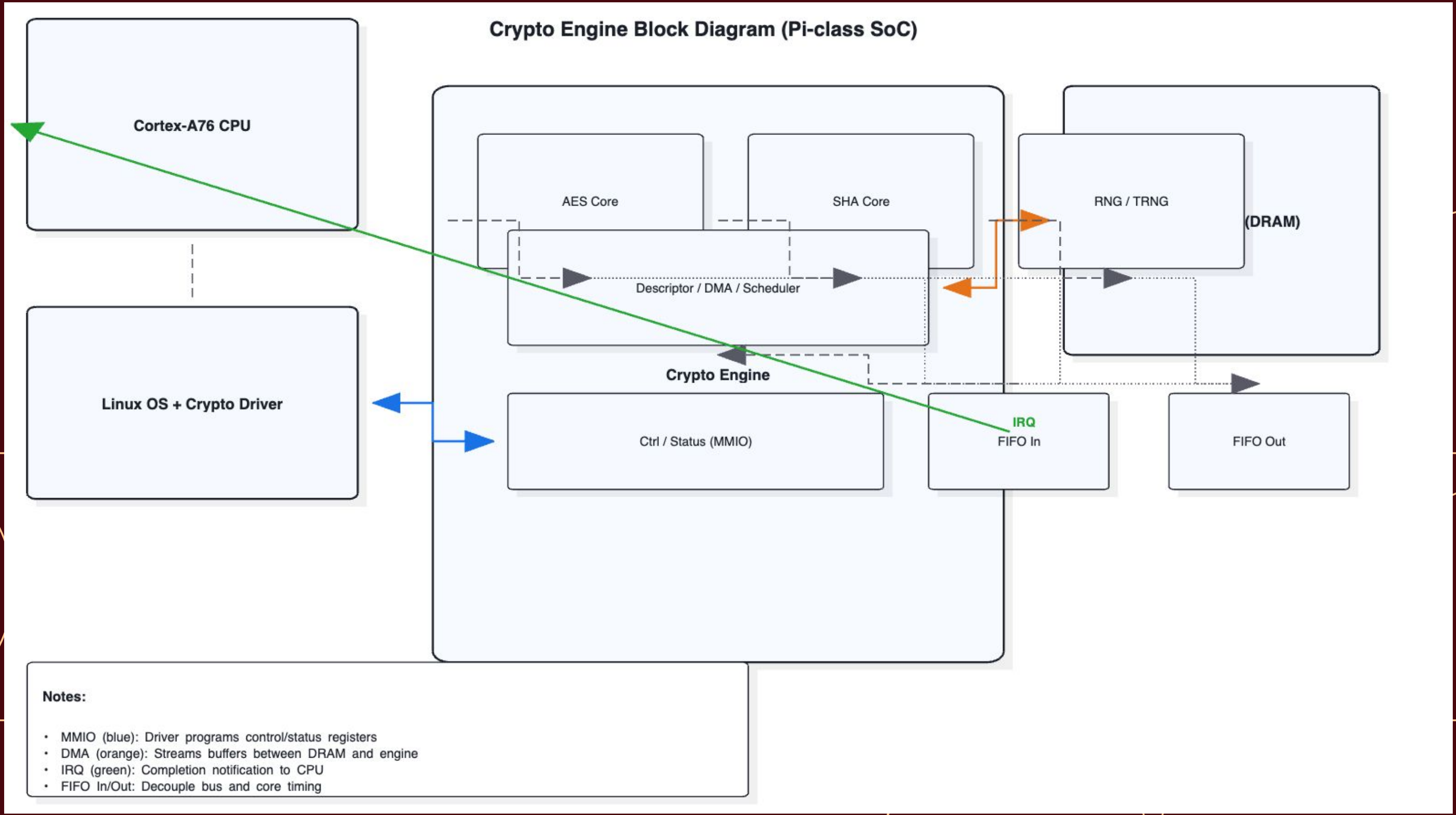
ECE 4301 Quiz 1
Crypto Engine on PI-5

By: Priyanka Ravinder, Arvind Mohanraj and Raul Garcia

What this Presentation Covers:

- Cryptographic workloads such as TLS, disk encryption and blockchain require plenty of power.
- Offloading to acceleration is able to lower the CPU overhead, latency and energy
- Crypto-engine designs for the PI-5 and the overall OS

Block Diagram of Crypto Engine



Notes:

- MMIO (blue): Driver programs control/status registers
- DMA (orange): Streams buffers between DRAM and engine
- IRQ (green): Completion notification to CPU
- FIFO In/Out: Decouple bus and core timing

Interface with Main Processor and OS

Device and Kernel Interface

- In terms of uniformity for the crypto operations, Kernel crypto API provides the uniform interface.
- The driver for the crypto engine is able to map MMIO registers in order to control and obtain the statuses of descriptors, operations, etc.
- Libraries such as OpenSSL detacts and uses hardware support

Data Flow and Control

1. The Kernel or crypto library submits a request.
   - For example, the request can be an algorithm, key, input, or buffers
2. The driver sets up and allocates the descriptors and programs the engine using the registers
3. DMA obtains plaintext and communicates it to the AES/SHA blocks
4. The engine then processes this data, and sends the results to memory using DMA
5. Once this is complete, the driver tells caller

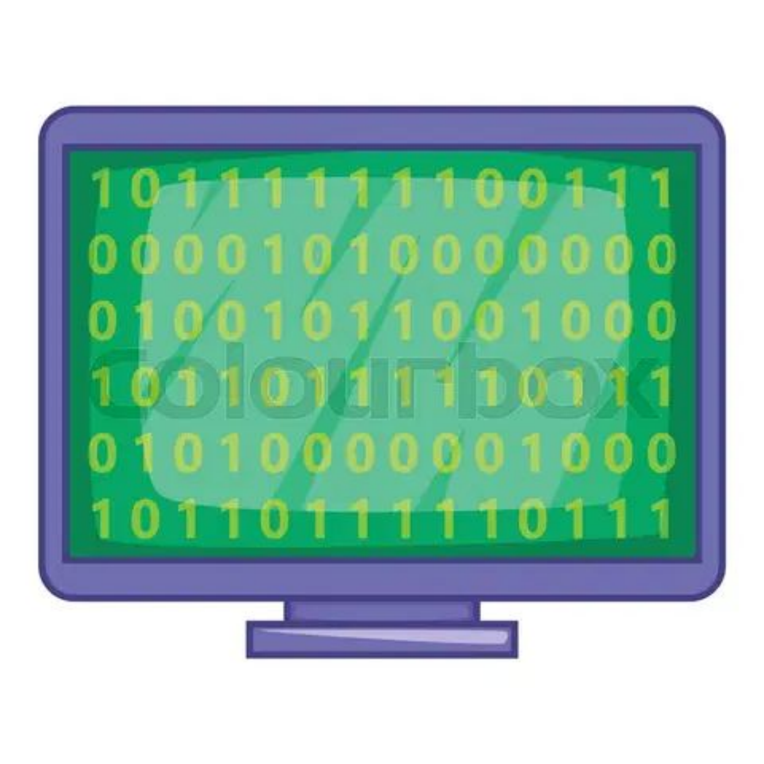
Memory

- Shared memory and buffers must abide by the caching and coherency regulations
- Prior to DMA transfer, there must be disposing of CPU caches so that once DMA is completed, the CPU is able to put in new data
- For the proper buffer layout, alignment must be considered to improve performance

AES Encryption

Here are the steps of encrypting a block of data using AES:

1. The driver loads the AES key into RAM
2. The driver then builds descriptors to point out the buffers in memory
3. DMA engine then streams the plaintext to the AES core
4. The core then performs rounds, for example SubBytes or AddROundKey
5. The signals finish and the driver validates the completion status
6. Once the AES rounds are finishes, the CPU performs its remaining tasks



Trade-offs and Extensions

Overall Benefits

- Low CPU load and high data processing abilities
- Lower latency for various crypto tasks
- Lower energy per bit processed

Trade-offs

- Added hardware complexities
- Non Flexible, newer algorithms can not be implemented and mapped easily
- In the event the core does not have extension support, there is risk of fallback to the software

Extensions

- Side-channel resistance
- Post-quantum crypto accelerators
- Add support for the ECC

References

1. *Documentation – Arm Developer*, developer.arm.com/documentation/101432/r1p2/Functional-description/About-the-Cryptographic-Extension. Accessed 26 Sept. 2025.
2. *Ti*, www.ti.com/lit/an/swra667/swra667.pdf?ts=1726232714031&ref_url=https%253A%252F%252Fwww.google.com%252F. Accessed 27 Sept. 2025.
3. "Board Index." *Raspberry Pi Forums - Index Page*, forums.raspberrypi.com/viewtopic.php?t=371780. Accessed 26 Sept. 2025.
4. Bennet, Tom. "Build & Run a Full Bitcoin Node with a Raspberry Pi 5." *Tom Bennet*, bennet.org/blog/building-a-bitcoin-node-with-raspberry-pi/. Accessed 26 Sept. 2025.
5. *Cryptographic Engine - an Overview | Sciencedirect Topics*, www.sciencedirect.com/topics/computer-science/cryptographic-engine. Accessed 27 Sept. 2025.