UNC CHARLOTTE

SECURE BOOT FOR FPGAS

HEADS HARDWARE AND EMBEDDED DESIGN AND SECURITY LAB

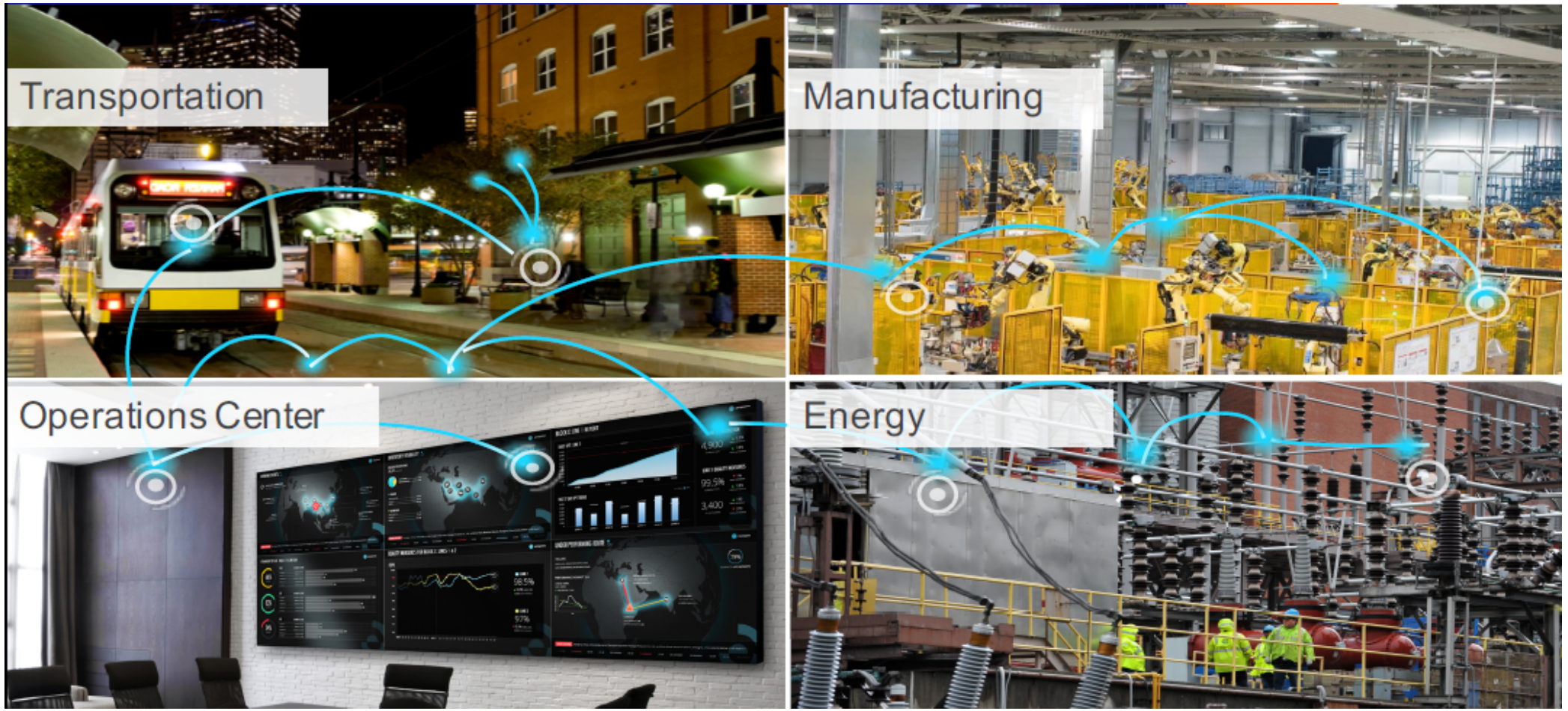
Lab Director
Fareena Saqib, Assistant Professor
Email: fsaqib@uncc.edu
Office: EPIC 2164
Phone: 704-687-8098

Why Engineering

"Women are often under-represented in the academic and professional fields of engineering." Wikipedia Women in Engineering

This is an exciting time to participate to push Women for change

Electronics: The Heart of Digital



Business Operations ⟺ Enterprise Culture ⟺ 3rd Party Ecosystem

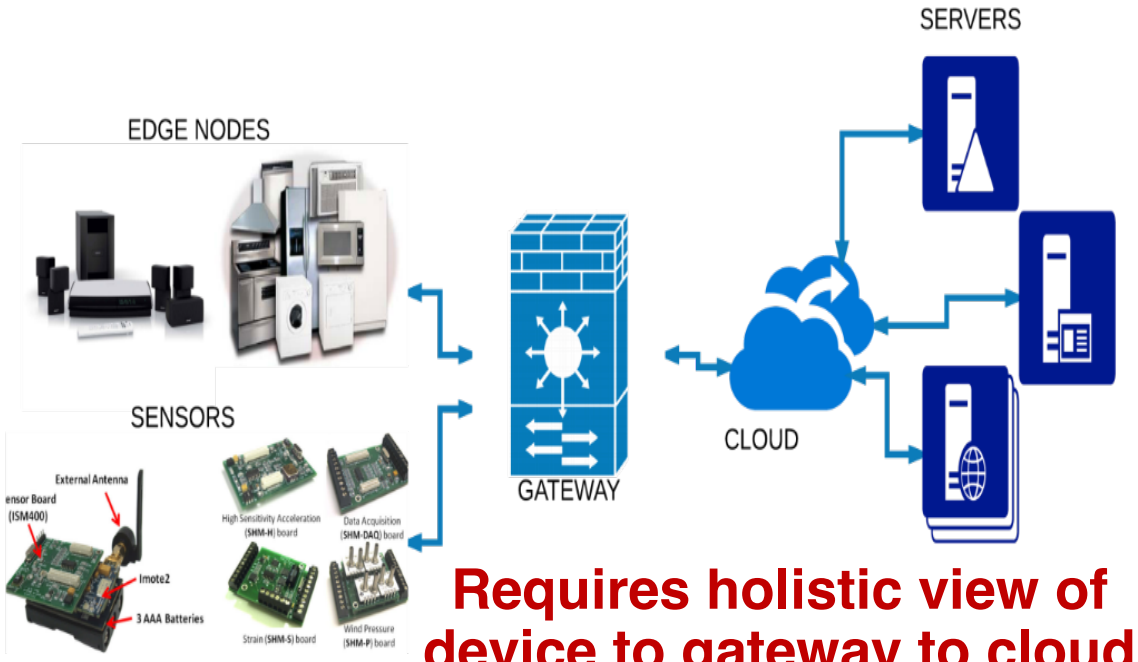
Hardware Security

Cyber security traditionally meant software, network and data security considering hardware as **root of trust**. This assumption is no longer true with evolving semiconductor business landscape.

IOT Challenges of connected devices

Connecting devices, that deliver value through smart interfaces and user experience

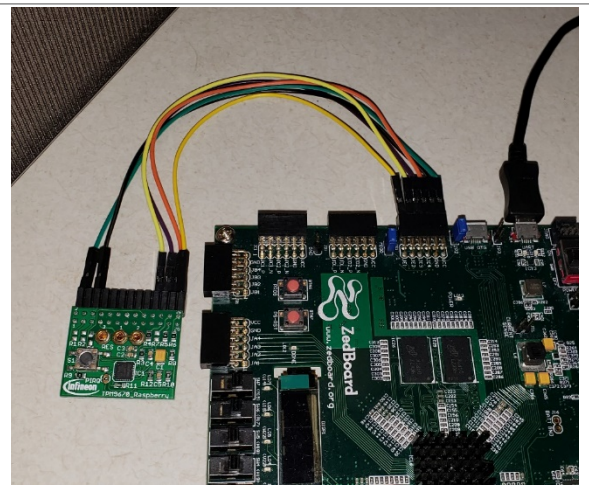
- Long life cycles of IoTs
- Provisioning keys and key management life cycle
- Security assessment of equipment that were never intended to be connected
- Device identification for device-to-device communication
- Availability, Scalability and system resilience
- Firmware updates

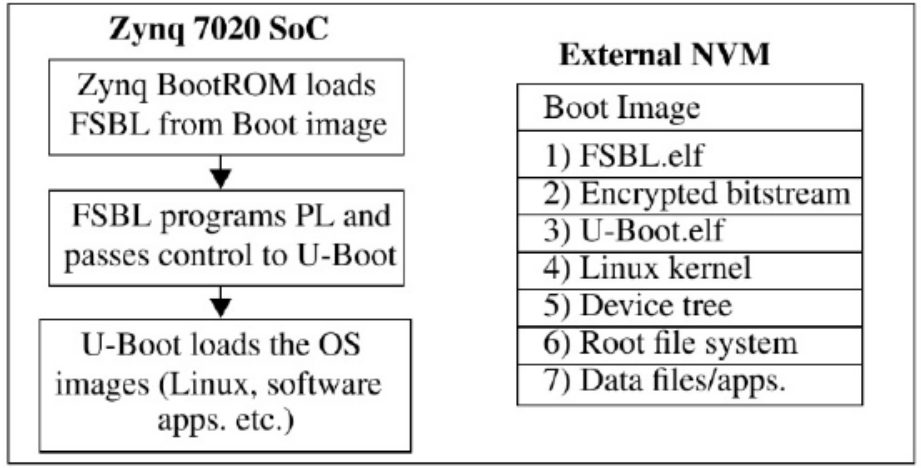


Requires holistic view of device to gateway to cloud and the communication between them.

Secure Boot Process

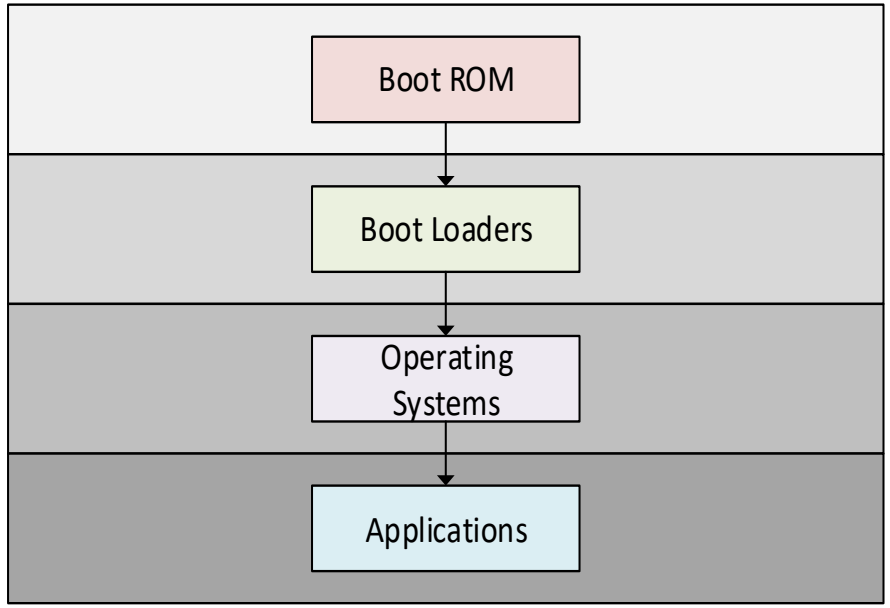
- An Autonomous, Self-Authenticating, and Self-Contained Secure Boot Process for Field-Programmable Gate Arrays using PUF.
- TPM based secure hardware framework for boot process and over the air update for reconfigurable computing.
- There are physical as well as remote threats. In this work we study invasive and non invasive attacks on keys, data and boot process.
- The research covers mutual authentication, key management and secure boot process for FPGAs.





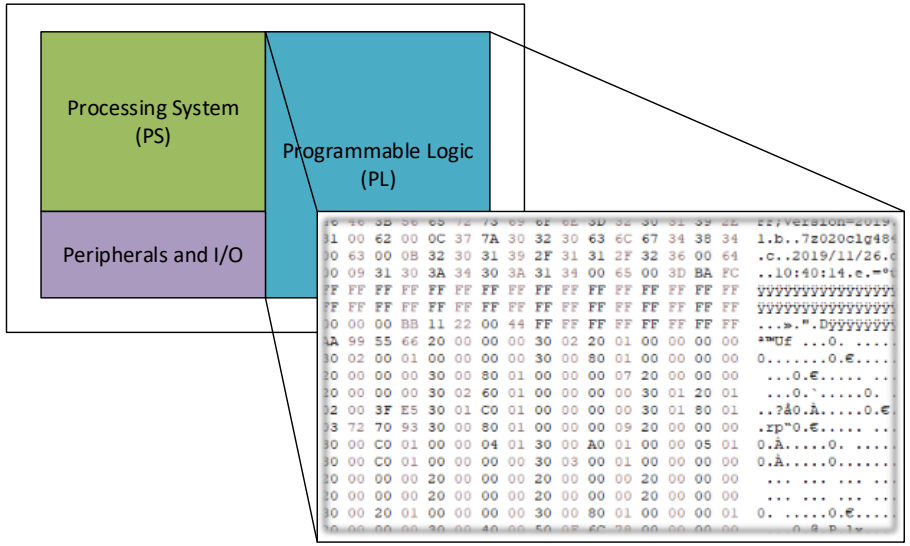
Device Boot

- A device boot-up can be divided into sequence of processes:
  - Firmware (Boot ROM)
  - Boot Loaders
  - Operating System
  - Applications
- An adversary having access to a process can exploit all the layers above it.



Boot Process

- FPGA based SoCs have reconfigurable fabric, processors, buses and interconnects.
- Programmable Logic (PL) fabric in an FPGA is composed of:
  - Look-Up Tables (LUTs)
  - Memory elements
  - Computation elements, e.g., DSP and ALU
  - Interconnects
- In SRAM based FPGAs, the input configuration is stored in a bitstream.
- At boot, the FPGA will configure the PL fabric with the bitstream configuration.
- Some FPGAs also allow runtime partial reconfiguration of the PL fabric.



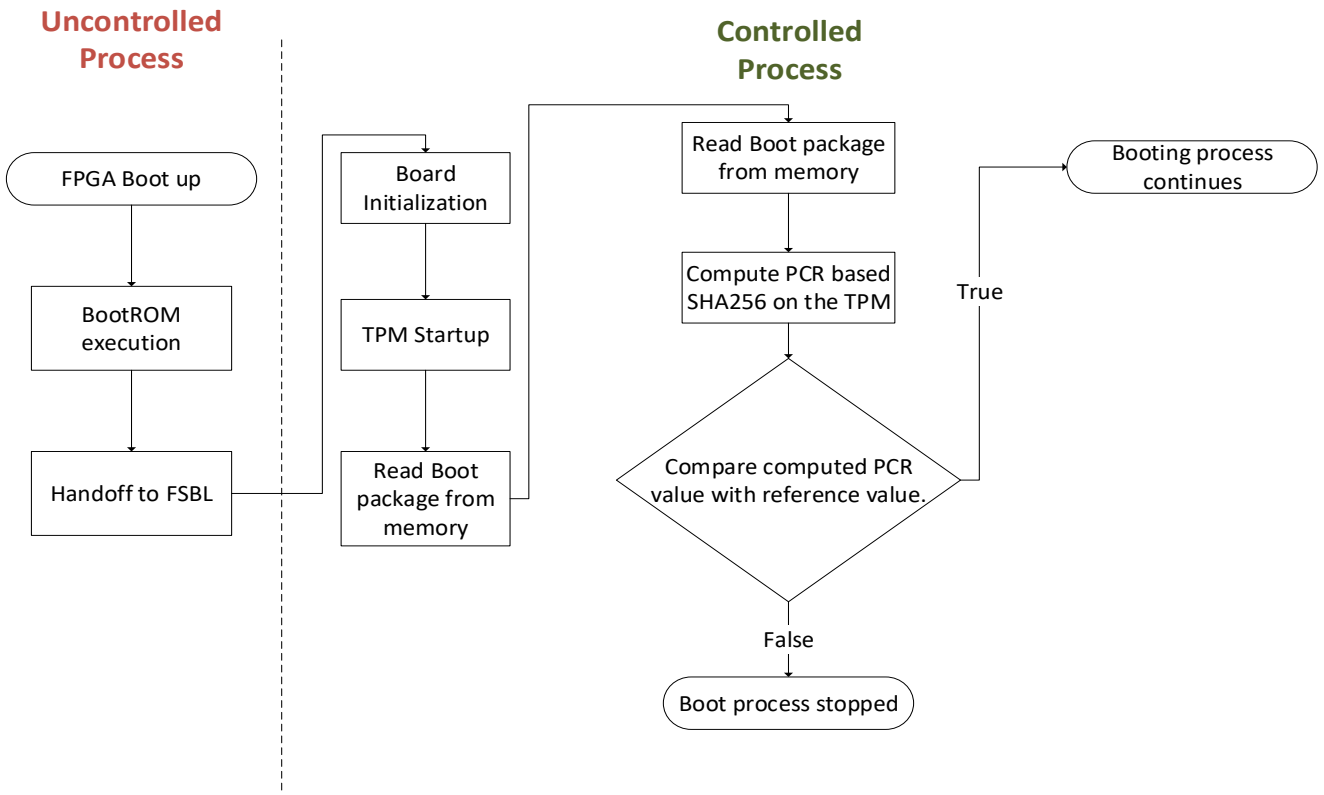
Boot Process

- Modification of the bitstream in a SRAM based FPGA leads to modification of the underlying architecture.
- In the hands of an adversary that can result in addition of malicious logic or even formation of information leaking side channels.
- Commercial FPGAs implement Secure Boot by implementing an on-board AES block and HMAC block.
- There are two zones for storing keys:
  - One Time Programmable eFuses
  - Battery Powered RAM

Secure Boot in FPGAs

- The FSBL and the bitstream are encrypted using the symmetric key. The Boot ROM decrypts the FSBL whereas the FSBL uses the AES core to decrypt the bitstream.
- Additionally, there is software support to implement RSA based authentication.
- There are certain shortcomings with this implementation:
  - BBRAM is not practical since it requires indefinite power supply for operation.
  - Efuses once programmed cannot be changed, therefore if the key is once discovered, it cannot be modified.
  - Once the encrypted payload has been decrypted and has been brought into the main memory, it is susceptible to Time of Check to Time of Use attacks (TOCTTOU).
  - The provided cryptographic cores are only usable by the Boot ROM and the FSBL. These cores cannot be used for any additional purposes.

TPM Based Secure Boot for FPGAs



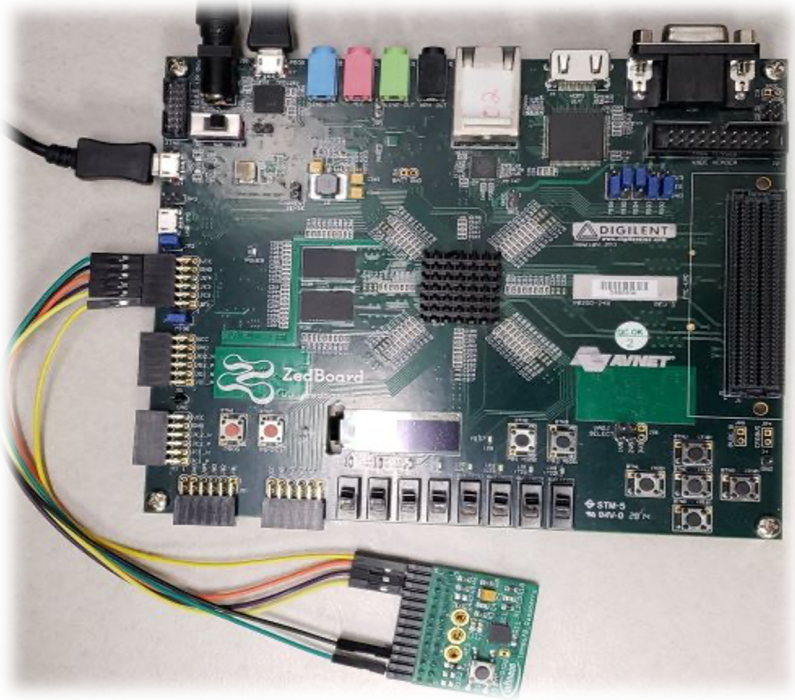


Figure: TPM providing secure boot to FPGA

TPM based integrity verification

```
Inside ComputeHash hook:
Image Start Addr: 00100000
Image End Addr: 001F6EC0
Acquiring Locality 4: 0
Reading Image:
tpm_pcr_read: INFO Command bytes: 80.01.00.00.00.14.00.00.01.7E.00.00.00.01.00.0
B.03.00.00.02.
tpm_pcr_read(): INFO tpm_send() return: 20
tpm_pcr_read(): INFO return size: 62
tpm_pcr_read(): INFO PCR_READ(REG:17): 80.01.00.00.00.3E.00.00.00.00.00.00.00.1A
.00.00.00.01.00.0B.03.00.00.02.00.00.00.01.00.20.0F.EC.53.25.1A.43.A4.65.6A.23.F
F.68.0A.26.40.AC.8E.B0.8F.CB.03.64.57.FC.BE.FC.9C.41.DD.8F.0D.70.
Computed Bitstream Hash: 0FEC53251A43A4656A23FF680A2640AC8EB08FCB036457FCBEFC9C4
1DD8F0D70
Reference Bitstream Hash: 0FEC53251A43A4656A23FF680A2640AC8EB08FCB036457FCBEFC9C
41DD8F0D70
Bitstream check complete. Boot process continuing normally...
```

Figure: Secure Boot Process completes successfully

```
tpm_pcr_read(): INFO PCR_READ(REG:17): 80.01.00.00.00.3E.00.00.00.00.00.00.00.1A
.00.00.00.01.00.0B.03.00.00.02.00.00.00.01.00.20.0F.EC.53.25.1A.43.A4.65.6A.23.F
F.68.0A.26.40.AC.8E.B0.8F.CB.03.64.57.FC.BE.FC.9C.41.DD.8F.0D.70.
Computed Bitstream Hash: 0FEC53251A43A4656A23FF680A2640AC8EB08FCB036457FCBEFC9C4
1DD8F0D70
Reference Bitstream Hash: 0FEC53251A43A4656A23FF680A2640AC8EB08FCB036457FCBEFC9C
41DD8F0D71
Reference Hash is not equal to computed hash. Image compromised
Performing Fallback.
PARTITION_MOVE_FAIL
```

Figure: Bitstream could not be verified successfully

Physical Unclonable Functions

- PUFs are embedded test structures to extract and digitize the process variations in the features, that are unique just like a finger print.
- PUFs rely on physical properties such as path delays, response behavior to glitches, initial boot-time values, threshold voltage of transistors.
- Strong or weak PUFs, and their applications.
- Physical Unclonable Functions (PUF) provide on-the-fly tamper resistant authentication.
- Strength of a PUF can be measured using metrics:
  - Randomness, Uniqueness, Reliability and entropy.

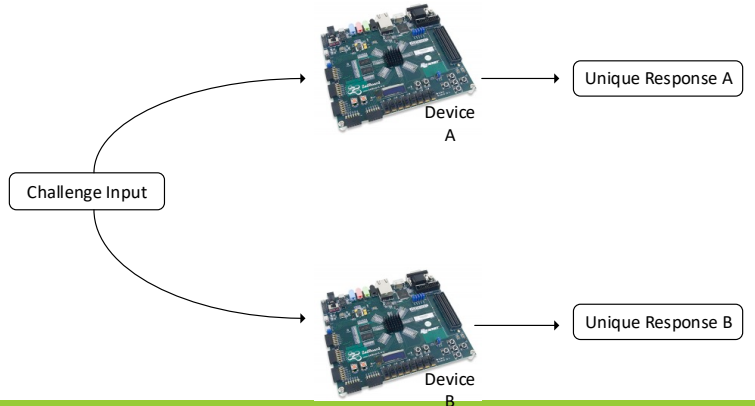
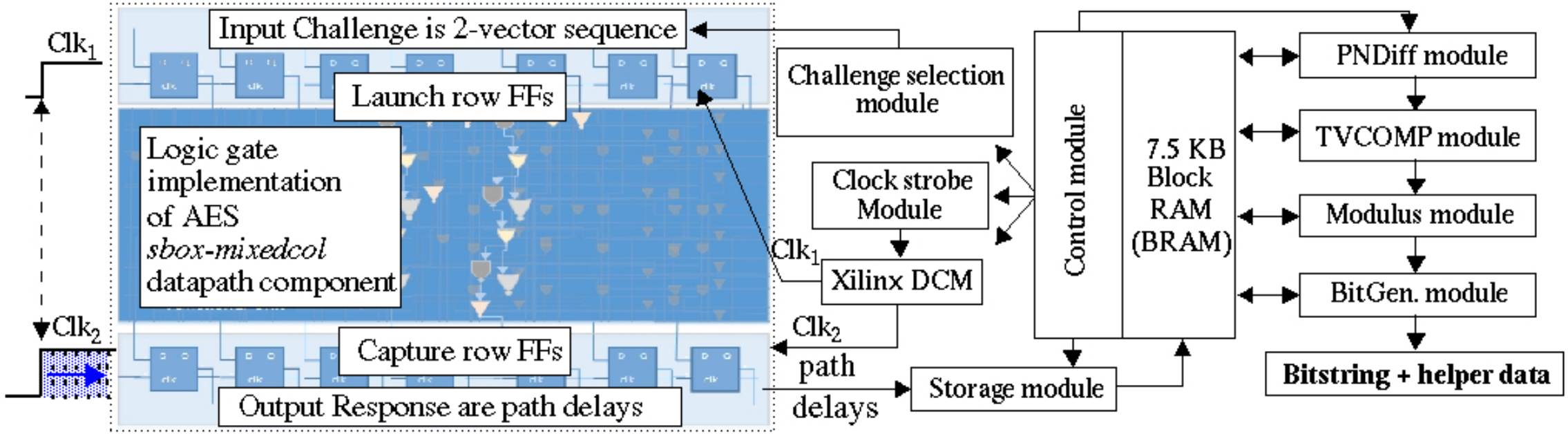


Figure: PUF generates unique responses when implemented on different devices.

Security Research: Physical Unclonable Functions PUFs

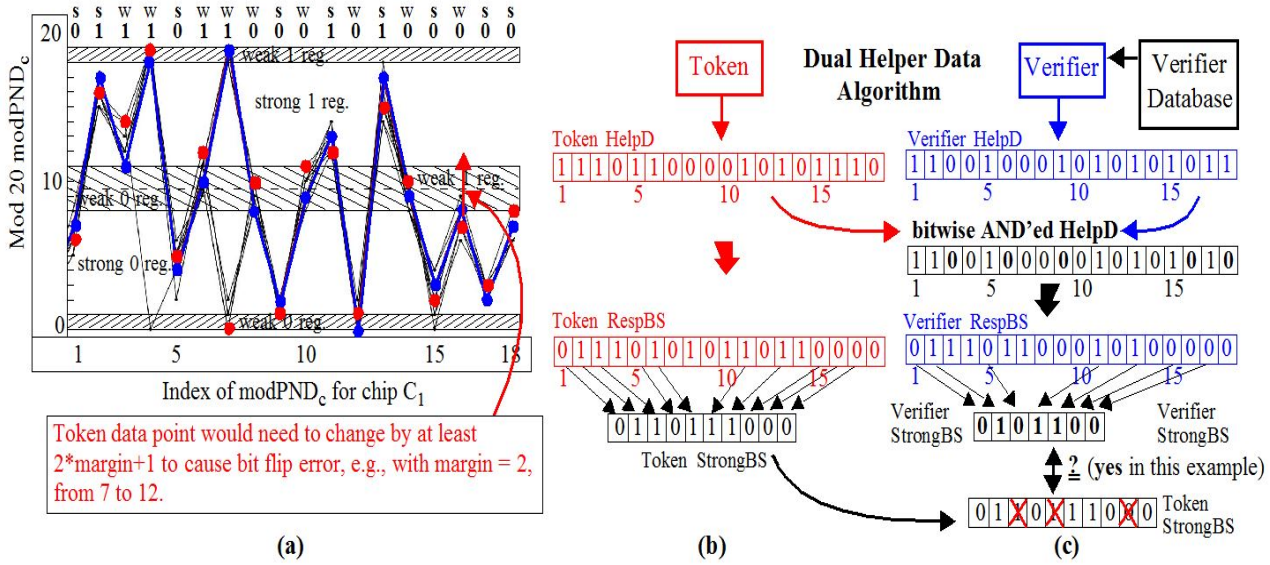
HELP entropy is path delays of existing functional units.

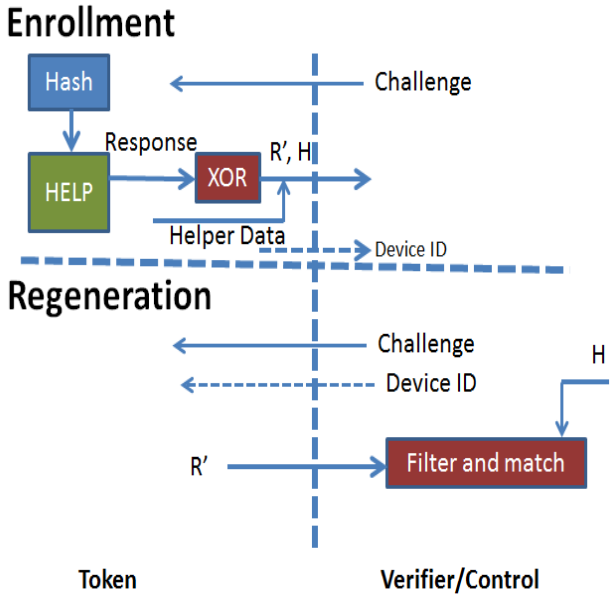
On-chip bitstring generation provides real-time identification.



Privacy Preserved Authentication in Distributed Environment

A privacy-preserving, mutual authentication protocol using dual helper data

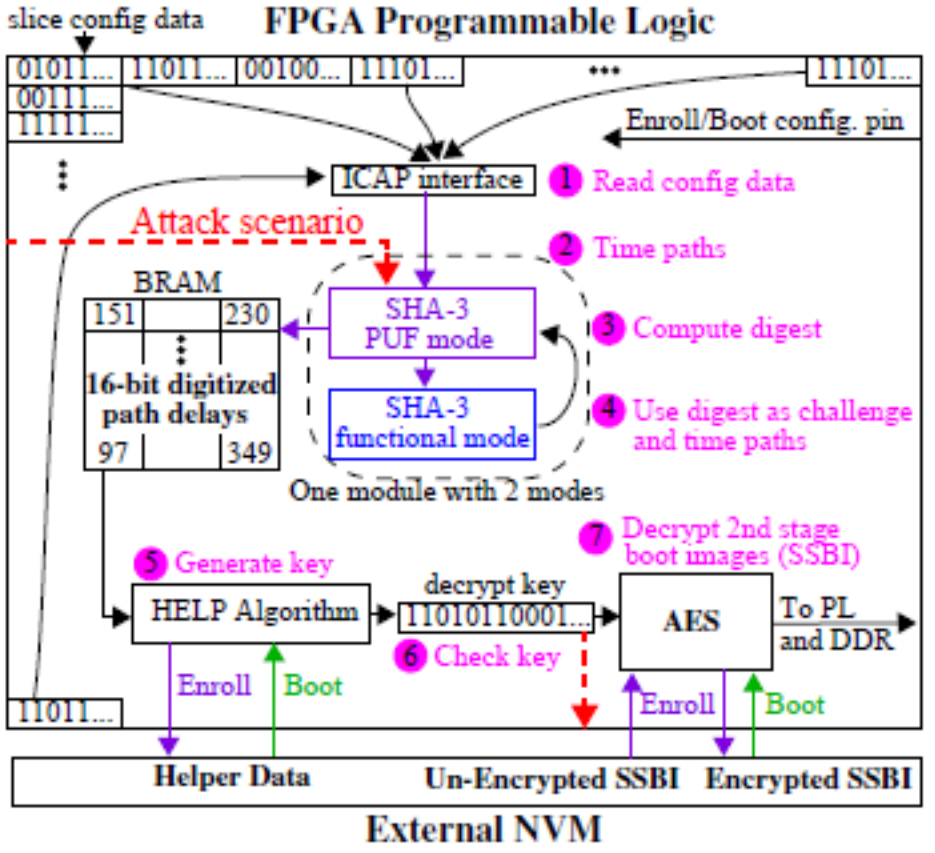




Sponsored by NSF

Secure Boot for FPGAs: An Autonomous, Self-Authenticating and Self-Contained Secure Boot Process

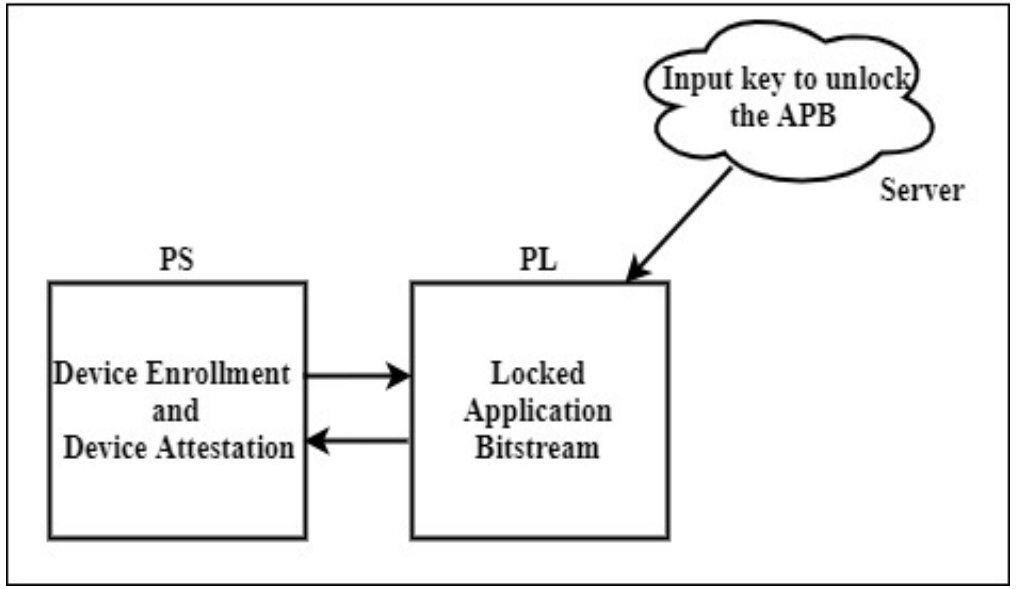
- Existing work for security of bitstream security in FPGAs provides limited security.
- In [1] and [2], the First Stage Boot Loader (FSBL) is assumed to be trustworthy.
- The FSBL loads a FPGA bitstream consisting of two partitions, static and dynamic.
- The static partition consists of a Physical Unclonable Function implementation.
- The PUF response acts as the key for decrypting encrypted dynamic partition which contain application logic.



[1] D. Owen Jr., D. Heeger, C. Chan, W. Che, F. Saqib, M. Areno and J. Plusquellic, An Autonomous, Self-Authenticating and Self-Contained Secure Boot Process for FPGAs, Cryptography, MDPI, 2018.
[2] G. Pocklassery, W. Che, F. Saqib, M. Areno and J. Plusquellic "Self-authenticating secure boot for FPGAs," in *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2018, pp. 221–226

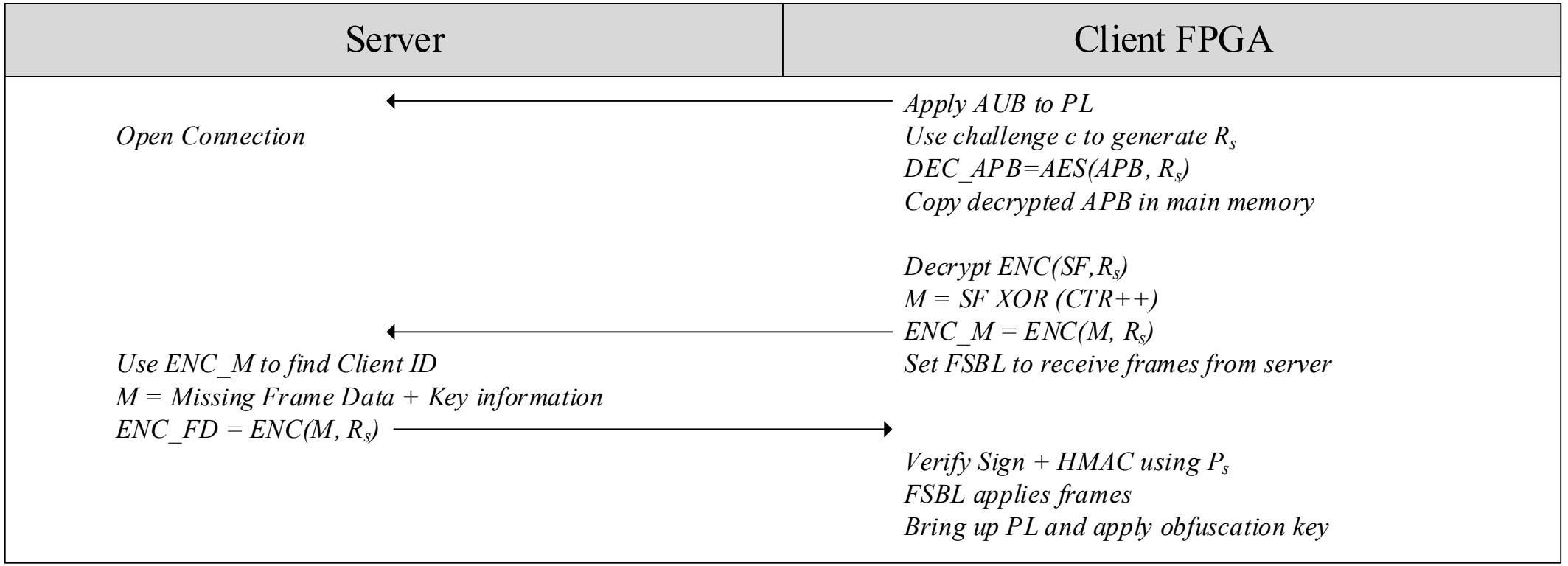
Multilayer camoflauged Secure Boot for SoCs

- The secure boot is a multilayered process, The enrollment in done in trusted environment.
- The first stage Authentication Bitstream (AUB) has a PUF to generate unique per device keys to decrypt the 2nd stage application bitstream.
- A unique bitstream (APB) is generated per device which consists of stripped and corrupted frames at the LUT granularity.
- The correct LUT configuration for the device is sent by the server after successful light weight device authentication.



[1] Ali Shuja Siddiqui, Geraldine Shirley Nicholas, SAM Reji Joseph, Yutian Gui, Marten Van Dijk, Jim Plusquellic, Fareena Saqib , Multilayer camouflaged secure boot for SoCs, MTV 2019

Boot and Device Attestation



LUT configuration

Comparison of before and after.

```
Readback XDevcfg Status Register
Original frame contents at address: 0x0000239b
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000
Modified PL frame at address: 0x0000239b
0000000000000000000000000000000000008080000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000100800000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000
PL Brought UP
CTL0 Register value: 0x00000501
FAR Register value: 0x0000239d
```

Thank you !!!