**HARDWARE SECURITY FOR FPGA’S**

Siva Prashanth B

University of Windsor

Electrical and Computer Engineering Dept.

**INTRODUCTION:**

Mobile and embedded devices are becoming ubiquitous, interconnected platforms for everyday tasks. any such tasks require the mobile device to securely authenticate and be authenticated by another party and/or securely handle private information. Indeed, smartphones have become a unified platform capable of conducting financial transactions, storing a user’s secure information, acting as an authentication token for the user, and performing many other secure applications. The development of powerful mobile computing hardware has provided the software flexibility to enable convenient mobile data processing. However, comparable mobile hardware security has been slower to develop.

The current best practice for providing such a secure memory or authentication source in such a mobile system is to place a secret key in a non volatile electrically erasable programmable read-only memory (EEPROM) or battery backed static random-access memory (SRAM) and use hardware cryptographic operations such as digital signatures or encryption. This approach is expensive both in terms of design area and power consumption. In addition, such non volatile memory is often vulnerable to invasive attack mechanisms. Protection against such attacks requires the use of active tamper detection/prevention circuitry which must be continually powered.

Physical unclonable functions (PUFs) are a promising innovative primitive that are used for authentication and secret key storage without the requirement of secure EEPROMs and other expensive hardware described above. This is possible, because instead of storing secrets in digital memory, PUFs derive a secret from the physical characteristics of the integrated circuit (IC). For example, this paper will discuss a PUF that uses the innate manufacturing variability of gate delay as a physical characteristic from which one can derive a secret [1].

**Types of PUFs:**

PUFs are modelled as CRP’s (Challenge Response Pairs), which means that when we give a challenge(c) to the PUF we get a response(r). So, depending on the challenge we can we get different response. The mathematical representation is given as:

r = f(c), where the function represents the unique property of PUF.

Broadly PUFs can be classified into two categories:

**Weak PUF**: Used For key storage. A weak PUF (also called as Physically Obfuscated PUF’s) has a very small range such that we can have collisions with one or other challenges that give the same response. We define the Weak PUF function as r=F(.) where F(.) means it has only very small domain. Even though it can be used as a fingerprint to generate cryptographic functions, the number of responses of the weak PUF is related to the number of components subject to manufacturing variation. A weak PUF can have only some secret keys (i.e the domain range is less).

**Strong PUF**: Commonly used for authentication. Strong PUf’s have a large domain and supports a lot of CRP’s that means it can be used effectively and securely. Not feasible to manufacture two PUFs with the same responses. The readout only reveals the response r = f(c) and no other data about the internal functionality of the PUF [1].

**Error Correction**:

We know that circuits response changes not only because of challenge but also because of the temperature and other environmental conditions. So, these affects the digital signature of the circuit to fail.

To tackle these problems, we use **differential design techniques** to filter the first order environmental dependencies. To improve the performance of the PUF we use soft coding technique and for reliability we make the PUF to give repeated outputs to get a stable output.

**Strong PUF Architectures:**

**OPTICAL PUF:**

Optical PUF was found by pappu et al. he found three components:

1) a laser directed along the Z-axis that can be moved in the XYplane

and whose polarization can be modified.

2) a stationary scattering medium that sits along the path of the laser beam.

3) an imaging device that records the

output ‘‘speckle’’ pattern of laser light exiting the scattering

medium.

Challenge: laser XY Location and polarization.

Response: Speckle pattern.

**ARBITER PUF:**

Although the capabilities of the above optical PUF are significant, and they represented a significant step forward in the understanding and construction of PUFs, the practical applications are limited due to the macroscopic optical nature.

Silicon implementations of strong PUFs were described in the literature beginning in 2002 using manufacturing variability in gate delay as the source of unclonable randomness.In one implementation, a race condition is established in a symmetric circuit. This is shown in Fig. 1. An input edge is split to two multiplexors (muxes). Depending on the input challenge bits (x[0] - x[127]), this path will vary. Although the layout is identical (propagation time should be the same for each edge no matter what challenge bits are chosen), manufacturing variability in the gate delay of each mux will result in one edge arriving at the latch first, and the latch acts as the ‘‘arbiter.’’ The output will, therefore, depend on the challenge bits.

In Fig. 1, there are 128 challenge bits and one response bit. Of course, one typically operates multiple identical circuits in parallel to achieve 128 response bits. In this way, the arbiter PUF can be scaled to an almost arbitrary number of CRPs.

The security of the arbiter PUF, like the optical PUF before it, is based on assumptions regarding manufacturing capabilities and ultimately metrology of the individual gate delays. Because the design is symmetric, the design does not contain any ‘‘secret’’ information. An adversarial manufacturer that has the PUF design cannot manufacture a duplicate PUF, because the behaviour of the PUF is defined by the inherent variability in the manufacturing process. Even the original manufacturer of the PUF could not produce two identical PUFs, since this would require a significant improvement in manufacturing control.

The second security assumption is that the individual gate delays are difficult to measure directly. It assumes that an invasive attacker would have difficulty in extracting the individual delays even with physical access. This assumption is based on the hypothesis that an invasive attacker would destroy the gate delay properties using his/her measurement techniques.

**Authentication of a Strong PUF**:

The following steps are used for authentication of a PUF.

1. PUF of its own fingerprint is manufactured.
2. Server generates all possible CRPS’s and stores them as a secret message.
3. PUF is given to a client.
4. Client submits request to the server.
5. Server picks a known CRP and submits the challenge to the client.
6. The client runs the challenge on the PUF and returns the response to the server.
7. Server checks to see that the response is correct and marks the CRP as used.

When we use a challenge, the server must make sure that the same challenge is not used again.

**Intra PUF Variation**:

Its defined as the number for bits in a PUF response that change when the same challenge input is given to the PUF. Usually, this happens due to environmental variation and statistical noise.

**Inter-PUF Variation**:

Its defined as the number of bits in a PUF response that vary between different devices for the same challenges. This is due to differences in IC Fabrication. It’s the measure of the uniqueness

of a PUF circuit.

A good PUF must have low values for Intra PUF Variation and a high value for Inter PUF variation.

**Weak PUF Architecture**:

**Ring Oscillator PUF**:

The design of a Ring Oscillator PUF has a not gate with 4 Not Gates connected with each other and the output is given as a feedback to the input.

We concatenate all Ring oscillators and use a decoder to select one oscillator at a time to use that intrinsic variability of that ring oscillator PUF.

**SRAM PUF**:

The SRAM has two states (either 0 or 1 state) , these state values differ when the SRAM is powered on because it displays the initial values which are in the SRAM those SRAM values are used to generate the PUF design.

**Cryptographic key generation: weak PUFs:**

Due to their limited challenge–response space, weak PUF architectures are typically used for cryptographic key generation. In this case, a weak PUF will replace a secure nonvolatile memory that would have stored the cryptographic key. Once the key is derived from the weak PUF, it is stored in secure volatile memory during the device’s operation. This key can then be used for authentication, encryption, and other cryptographic protocols. Due to the fact that one or very few keys can be generated by the PUF, the security of this key during operation is of paramount importance. If the secure key is revealed, any device can emulate the weak PUF.

**KEY GENERATION IN WEAK PUF’S**:

The cryptographic key that is derived from the weak PUF , once its derived the weak PUF its stored in the secure volatile memory for device’s operation.

PROTOCOL:

The key generated by weak PUF is used for authentication. By supplementing the weak PUF with hardware HMAC/AES implementation we can achieve authentication capability that embody the HMAC/AES protocol.

EMERGING PUF CONCEPTS:

Although existing PUF technology has been successful in addressing applications in low cost authentication and secure key generation, PUF technology still has significant untapped potential.

**MODEL BASED PUF’S**:

While considering the application of low cost authentication, one of the primary drawbacks of strong PUF architectures is the establishment of the secret message challenge. This secret model PUF still requires both the “secure bootstrapping” phase as well as the secure storage. This model must be kept secret, as it exactly describes the PUF behavior and this can be spoofed in the authentication process. So, we can directly compute the random challenge and computing the response.

**TIMED AUTHENTICATION AND PUBLIC MODELS**:

As the model of the PUF has large storage requirement so we go to new type of concept on PUF called as timed authentication PUF’s, PPUF’s, SIMulation Possible but Laborious (SIMPL) systems.

A PPUF has a model that is public (i.e known to everyone). The authentication scheme works as follows (Where a server is authenticating a client):

1. Server obtains the desired PPUF model from a trusted third-party storage.
2. Server generates a challenge and computes the response using the PPUF model.
3. Server sends response to the client and begins the timer.
4. The client uses its PPUF hardware to compute a response and send it back to the server.
5. Server measures the client response time T.
6. Server accepts if T<T0 and client’s response is correct.

Eventhough the PUF model is public, the model is resistant against tampering. This can be done using the traditional public key infrastructure(PKI). The server must establish a value T0 as described in the above mechanism.

1. The time must be greater than the PPUF hardware to compute the response and allow for roundtrip network latency.
2. Short enough that no model could emulate the PPUF hardware and correctly produce a response in that time.

**Proposed Research:**

The generic concept of a physically unclonable function was presented little over a decade ago, but the major research contributions on this topic are situated in the last couple of years. PUFs are hence a relatively new subject in the field of physical security, and several interesting future research directions, as well as some encountered open problems can be identified.

**ANDERSON PUF FOR ALTERA:**

**Planning to write about it.**

**REFERENCES:**

[1] Charles Herder, Meng-Day (Mandel) Yu, Farinaz Koushanfar, and Srinivas Devadas, Physical Unclonable Functions and Applications: A Tutorial”, Proceedings of the IEEE, Volume: 102, Issue: 8, Publication Year: 2014.

[2] Anderson, J.H.; A PUF design for secure FPGA-based embedded systems; 2010 IEEE 15th Asia and South Pacific Design Automation Conference (ASP-DAC).

[3] Miodrag Potkonjak, Vishwa Goudar Public Physical Unclonable Functions: A Tutorial”, Proceedings of the IEEE, Volume: 102, Issue: 8, Publication Year: 2014.