TUTORIALS 7269

CRYPTOGRAPHY: UNDERSTANDING THE BENEFITS OF THE PHYSICALLY UNCLONABLE FUNCTION (PUF)

By: Zia Sardar

Abstract: This tutorial is part of a series that is designed to provide a quick study guide in cryptography for a product development engineer. Each segment takes an engineering rather than theoretical approach on the topic. In this segment, you'll learn how the physically unclonable function (PUF) provides one of the most advanced protections in cryptography. A similar version of this tutorial originally appeared on May 13, 2020 on Electronic Design.

Introduction

In cryptography and within embedded security ICs, the PUF is used to create keys that are generated on-demand and instantaneously erased once used. PUF is dependent on random physical factors (unpredictable and uncontrollable) that exist natively and/or are incidentally introduced during a manufacturing process. That's why it is virtually impossible to duplicate or clone. PUF technology natively generates a digital fingerprint for its associated security IC, which can be utilized as a unique key/secret to support cryptographic algorithms and services including encryption/decryption, authentication, and digital signature.

Except for the momentary duration of a cryptographic operation, the PUF key value never exists in digital form within the circuitry of the security IC. Further, since the key is derived and produced on-demand from physical characteristics of circuit elements, they are never present in the device's nonvolatile memory. Any

attempt to discover the PUF key through micro-probing or other invasive techniques will disrupt the sensitive circuitry used to construct the PUF key and render the output useless. Thus, they provide a level of security that is very desirable in today's embedded systems.

In this short lesson, you'll gain a better understanding, via a simplified hypothetical PUF architecture and its usage, of why PUF-based key generation can provide such excellent protection in cryptographic applications.

A PUF Example

Figure 1 shows two separate example devices, each with a 64-bit PUF-based key (*this is a simplified general view*).

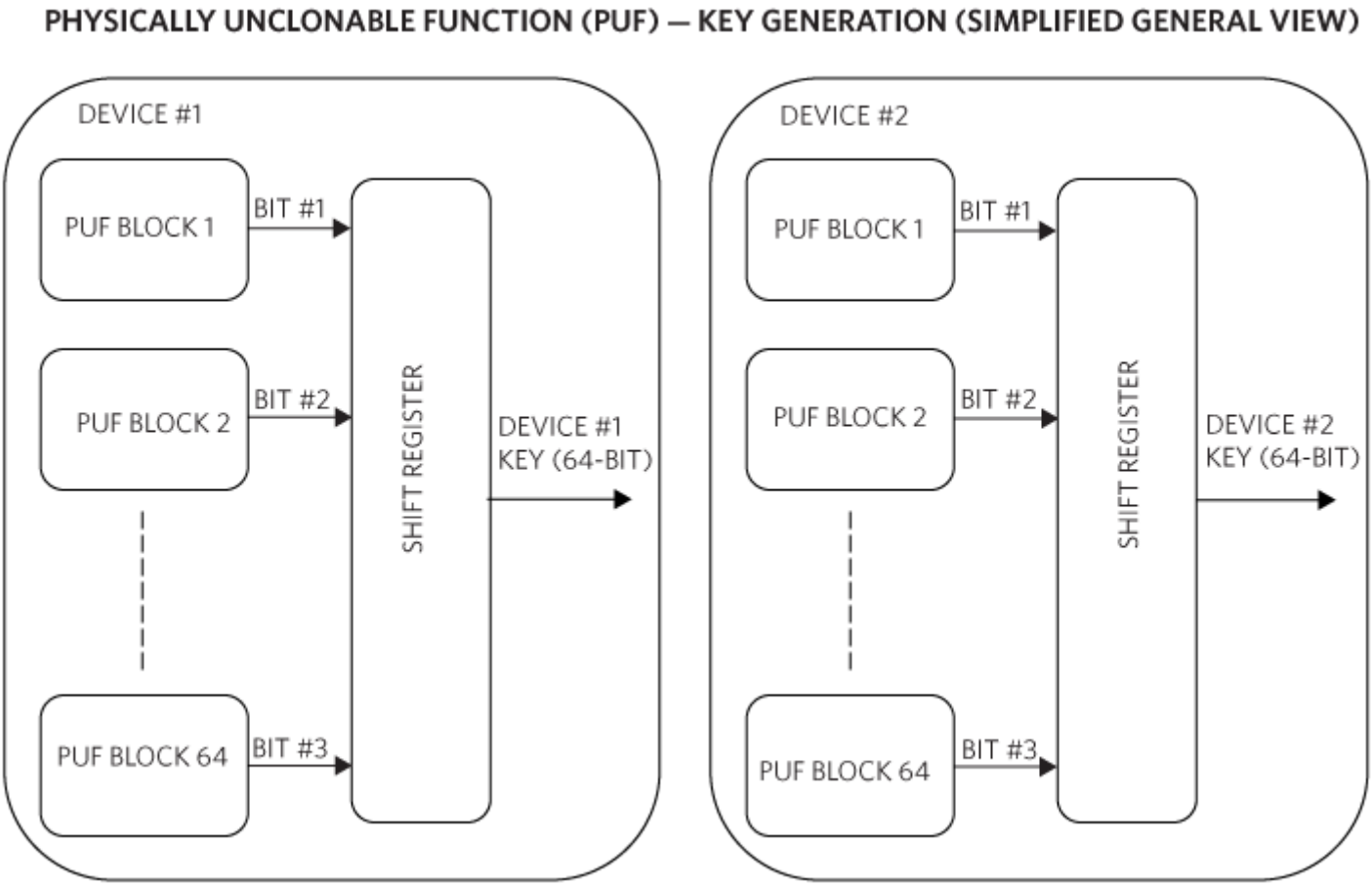

Figure 1. This figure shows two devices and their PUF key generation blocks.

PUF– Key Generation

Each device in Figure 1 has 64 different PUF blocks that generate 1 bit of data. The bits are then shifted to create the 64-bit key. Now our target is to have independent keys for each of these devices that are repeatable over voltage, temperature, and age. **Device 1** will produce a key that will have sufficient number

of bits that are different from the key produced by **Device 2**. Each of the device keys, however, will stay constant over the specified voltage and temperature range.

Let's consider a potential implementation of the PUF blocks of one of the devices in detail.

Data Bit Generation (Simplified General View)

In **Figure 2**, we show a simple PUF implementation scheme based on the frequency variation of ring oscillators. Later in this application note, we will expand on ring oscillators and learn why they produce slightly different frequencies for each instance of the block.

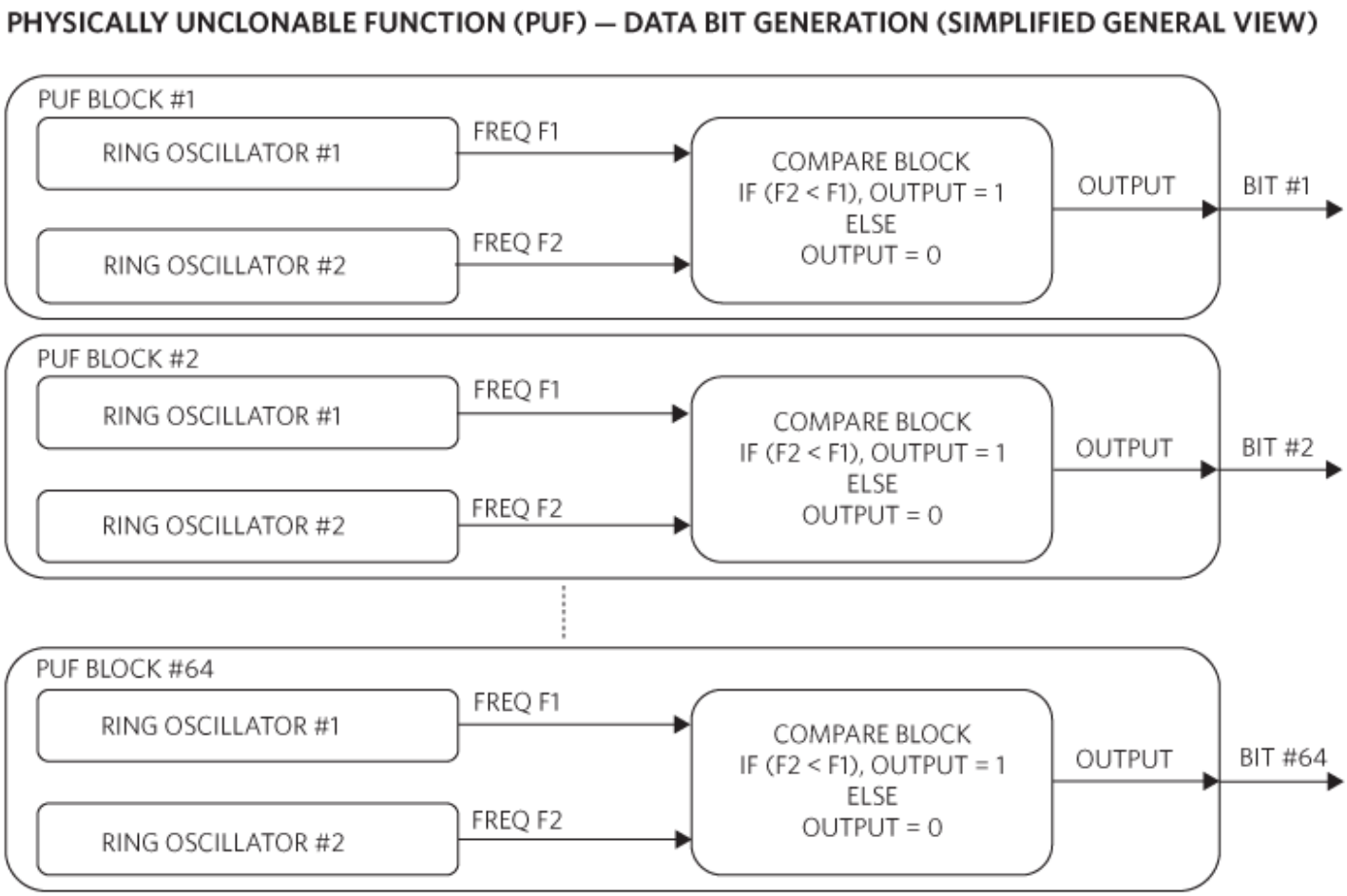

Figure 2. PUF data bit generation uses ring oscillators.

For now, let's assume that each of the PUF blocks has two ring oscillators that produce slightly different frequencies. In PUF block 1, F1 will be slightly different than F2 and this will let the compare block produce a bit 0 or bit 1 based on whether or not F2 is faster than F1.

How does this design help with voltage, temperature, and age variations? We'll compare two values to generate the bits rather than basing it on one frequency output. Thus, if with a higher voltage F2 increases, F1 will also increase but the delta between the two values should stay very much the same. This results in the same bit value produced with a different applied voltage. The effects of temperature and aging can be mitigated in a similar way.

As PUF blocks 2 to 64 are instantiated, the ring oscillator blocks inside them will produce slightly different frequencies from each other in an unpredictable way. This results in an unpredictable bit pattern for bits 1 to 64. Although the overall bit patterns can't be predicated, the actual bit pattern produced is repeatable as each block always produces the same bit.

Now let's look at a simplified ring oscillator design.

Simplified PUF Element – Ring Oscillator

A ring oscillator is constructed from an odd number of inverters arranged as shown in **Figure 3**. The output frequency of the oscillator is dependent upon the delay of each of the stages of the ring. From an IC perspective, this delay is dependent on wafer characteristics including oxide thickness, capacitance, and threshold voltage of each transistor that makes up the inverter stages. Due to imperfection and variation in the semiconductor manufacturing process, all these parameters vary slightly and with complete randomness. No matter how controlled the process is, it is impossible to avoid this random variation. All of this turns out to be a good thing for a PUF implementation because the key produced needs to have cryptographic-quality randomness.

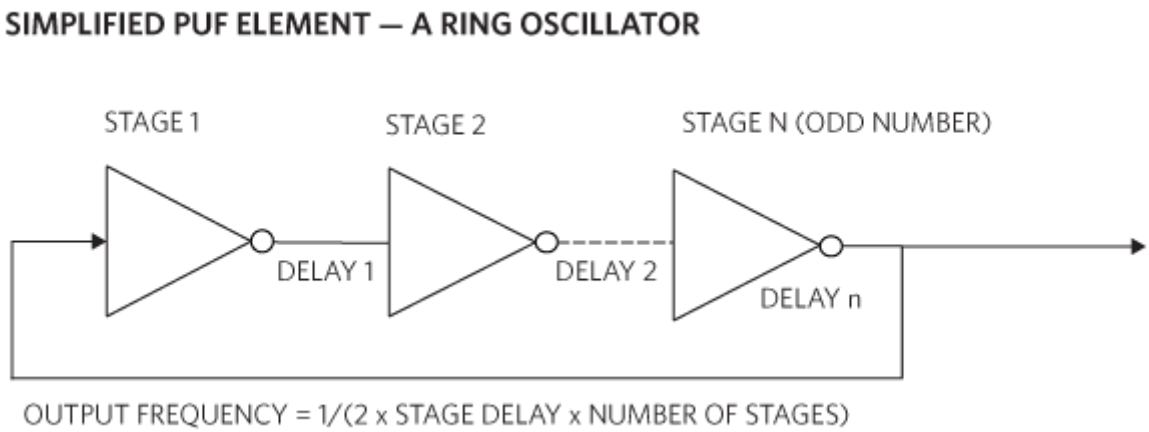

Figure 3. A simplified PUF element-ring oscillator constructed from an odd number of inverters.

A primary value that a PUF brings to cryptography is the inherent protection of the key/secret value that it derives and produces from the physical characteristics of circuit elements. As described in the ring oscillator example, electrically sensitive parameters such as threshold voltages, capacitance, and gate oxide thickness directly influence delays through the circuit. Probing or modifying the PUF, in an effort to obtain the key, permanently modifies and disrupts these sensitive characteristics. This results in a change to the

key value the PUF would output, making it useless and rendering the crypto IC permanently nonfunctional. Similarly, any attempt to reverse-engineer and clone a PUF results in a circuit that cannot output a key value which is correct for the system environment in which the security IC is deployed.

ChipDNA™ PUF Technology

Maxim Integrated has produced an implementation of PUF technology called ChipDNA. It is not ring-oscillator-based like the hypothetical example. Instead, ChipDNA fundamentally operates from the naturally occurring random variation and mismatch of the analog characteristics of MOSFET semiconductor devices. This randomness originates from factors similar to those previously described: oxide variation, device-to-device mismatch in threshold voltage, interconnect impedances, and variation that exists within wafer manufacturing through imperfect or non-uniform deposition and etching steps. ChipDNA also operates from a patented approach to ensure that the unique binary value generated by each PUF circuit has high cryptographic quality and is guaranteed to be repeatable over temperature, voltage, and the device's lifetime.

ChipDNA Applications

Figures 4 and **5** show use cases for ChipDNA PUF technology that is integrated into a secure authenticator cryptographic IC.

Securing Stored Data

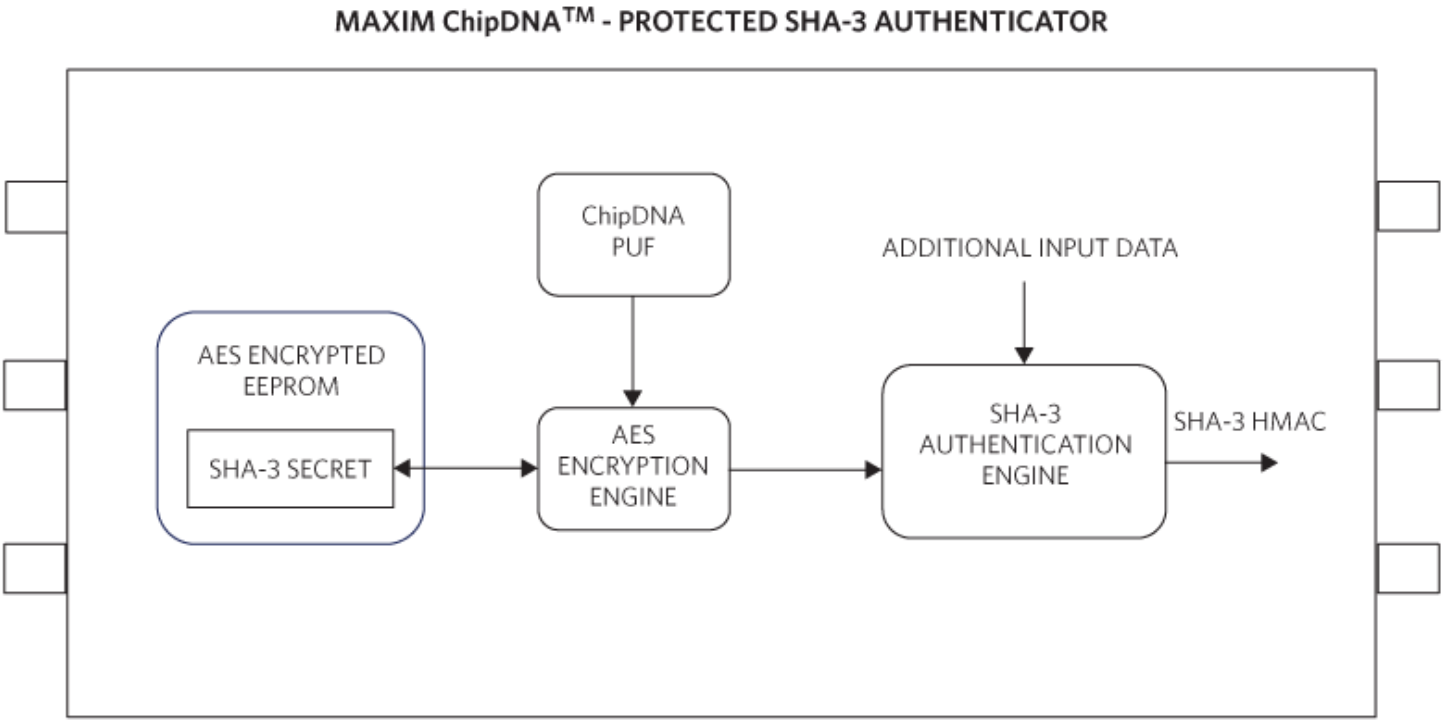

Figure 4. A ChipDNA-protected SHA-3 authenticator secures stored data.

In this case, we are using the ChipDNA PUF key, when needed, to decrypt the EEPROM-stored SHA3 secret used for an authentication sequence. Any attempt to obtain the secret from EEPROM will result in AES-encrypted data that is useless to an attacker. Additionally, any effort to probe the PUF will result in permanent disruption of its operation, resulting in the device being inoperable and again, useless to the attacker. However, if the device is not tampered with, it will serve its purpose faithfully by generating the PUF key on-demand and decrypting the SHA-3 secret for a momentary authentication operation.

Key Generation for Authentication

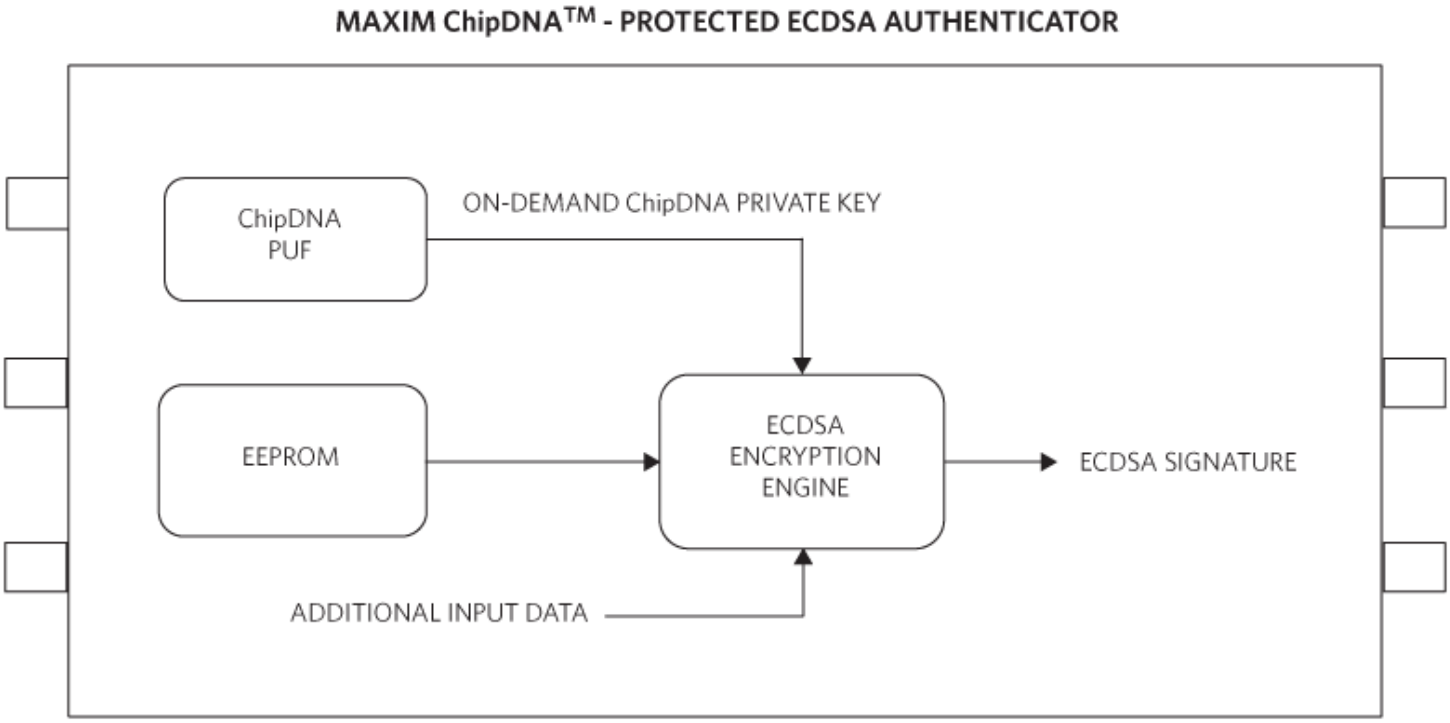

Figure 5. A ChipDNA-protected ECDSA authenticator provides on-demand key generation.

In Figure 5, the ChipDNA PUF key is used directly as the private key for an ECDSA signature computation. With ECDSA, by definition, the private key is a random number. ChipDNA is ideal in that it produces a high cryptographic quality random number. With asymmetric ECDSA there is also a public key required to verify signatures and it is mathematically associated with the private key. Since the PUF key and, in this example, the ECDSA private key, are never exposed outside the IC, the public key would be computed by the ECC engine within the part and stored in EEPROM for transmission to a host controller when requested.

This second example also demonstrates the use of ChipDNA PUF within the scope of benefitting key infrastructure and management. In more complex use cases, ChipDNA becomes the immutable root key of the security IC.

Learn more about specific symmetric and asymmetric key-based hardware authenticators from Maxim Integrated that can be used to accomplish all the concepts discussed in this chapter. Watch for other segments in our series of cryptography tutorials to continue deepening your understanding of this important security technique.

ChipDNA is a trademark of Maxim Integrated Inc.

Related Parts		
DS2477	DeepCover Secure SHA-3 Coprocessor with ChipDNA PUF Protection	Free Sample
DS28E39	DeepCover Secure ECDSA Bidirectional Authenticator with ChipDNA PUF Protection	Free Sample
DS28C39	DeepCover Secure ECDSA Bidirectional Authenticator with ChipDNA PUF Protection	Free Sample
DS28E50	DeepCover Secure SHA-3 Authenticator with ChipDNA PUF Protection	Free Sample
DS28C50	DeepCover I^2C Secure SHA-3 Authenticator with ChipDNA PUF Protection	Free Sample

Next Steps	
EE-Mail	Subscribe to EE-Mail and receive automatic notice of new documents in your areas of interest.

APP 7269: 15 Jun, 2020
TUTORIALS 7269, AN7269, AN 7269, APP7269, Appnote7269, Appnote 7269