Keywords: DeepCover, master/slave/mutual authentication, security, authentication protection

APPLICATION NOTE 5785

IMPLEMENT HEIGHTENED SECURITY WITH A SHA-256 MASTER/SLAVE AUTHENTICATION SYSTEM

By: Bernhard Linke, Principal Member Technical Staff

Abstract: The DS28C22 is a DeepCover® secure authenticator with I²C interface that uses the SHA-256 algorithm for bidirectional authentication. Additional features, including a 3Kb user EEPROM array, multiple memory protection methods and advanced physical security, combine to provide the ultimate in cost-effective IP protection, clone prevention, and authentication. This document describes operating principles of the device, its special features, and its typical application environment.

A similar version of this article appeared on *Electronic Products* May 16, 2014.

Introduction

For more than 10 years, SHA-1 authentication was used to effectively protect intellectual property (IP) from counterfeiting and illegal copying. But now, as computer technology information processing has advanced, customers want an even higher level of security.

Today a new secure authenticator and a companion secure coprocessor implement SHA-256 authentication. This new technology provides advanced physical security to deliver unsurpassed low-cost IP protection, clone prevention, and peripheral authentication. This article explains the general logistics of a SHA-256-based security system and discusses the bidirectional authentication functionality that the authentication system utilizes.

A Secure Authentication System

Implementing a secure authentication system requires linking a host system with a sensor/peripheral module. The system presented in Figure 1 consists of a SHA-256 secure authenticator plus a SHA-256 secure coprocessor. The host communicates with the authenticator and the coprocessor over the industry-standard I²C bus.

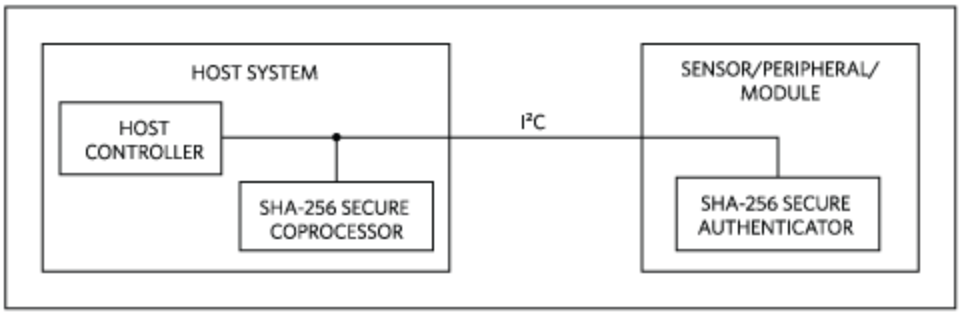

Figure 1. Implementation of a SHA-256 secure authentication system. This system features the DeepCover® DS2465 SHA-256 secure coprocessor and the DeepCover DS28C22 secure authenticator.

The SHA-256 Secure Authenticator

The SHA-256 secure authenticator in this system supports a challenge size of 256 bits and uses a 256-bit secret. The authenticator in Figure 1 is an I^2C slave with a unique 64-bit ROM ID, which serves as a fundamental data element for authentication computations. The system designer can partition the 3Kb user EEPROM into areas with open (unprotected) access, areas where the master must authenticate itself for write access, and areas where the read and write access involves data encryption. Encryption can be combined with authentication to further increase the data security. Table 1 shows the available protection modes.

Table 1. Secure Authenticator Protection Options**

RP	Read Protection. If activated, the data is only accessible for internal use, e.g., like a secret.
WP	Write Protection. If activated, the data cannot be changed.
EM	EPROM Emulation Mode. If activated, individual bits can only be changed from 1 to 0.
AP	Authentication Protection. If activated, write access to the memory requires master authentication.
EP	Encryption Protection. If activated, the data is encrypted on its way to the host controller and when sent to the secure authenticator for write access.

**The system default is no protection with RP, WP, EM, AP, and EP not activated. Protection is cumulative.

The SHA-256 Secure Coprocessor

The SHA-256 secure coprocessor in Figure 1 relieves the host processor from performing SHA-256 computations. More importantly, the secure coprocessor embeds protected memory that securely stores a master secret. Additional memory is set aside to store and protect other data elements used to compute unique slave secrets. From the host's perspective, the SHA-256 secure coprocessor appears as a 256-byte read/write memory with certain regions (data elements) assigned for special purposes.

Security Logistics

SHA-based security relies on message authentication codes (MACs) computed from open data and a secret. To verify authenticity, both sides, i.e., the host or coprocessor and the authenticator, must know the

secret, which shall never be exposed. Moreover, for maximum security the secret in each authenticator must be unique. In this way the security of the entire system is not affected if the secret of a single authenticator is ever compromised.

At first glance, it may appear impossible to meet these requirements. There is, however, a simple solution: compute the secret from known “ingredients” and install it into the secure authenticator in a trusted/controlled manufacturing environment. The ingredients for a unique secret are a master secret; the binding data; a partial secret; the ROM ID of the secure authenticator; and padding/formatting (“other data”). **Figure 2** illustrates the process. Although the ingredients are exposed at one point in time, for example, in a trusted manufacturing environment, the computed secret is never exposed and always remains hidden.

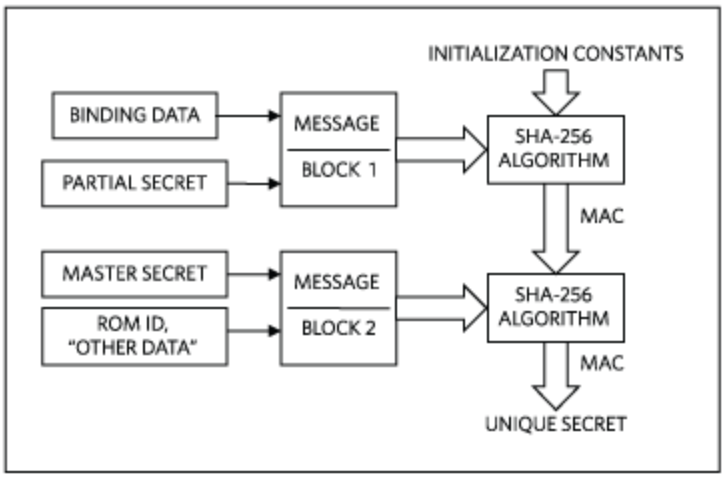

Figure 2. Creating a unique secret.

For security and storage space reasons, the unique secrets of all secure memories in a system cannot be stored in the secure coprocessor or host. Instead, the coprocessor stores only the master secret and the binding data in a protected memory section. The partial secret is a system constant that can be coded in the host processor’s firmware and communicated openly. After having read the authenticator’s ROM ID, the coprocessor can compute a unique secret, as shown in Figure 2. With both authenticator and coprocessor now sharing the unique secret, the system is ready to operate.

Challenge-and-Response Authentication

The primary purpose of the secure authenticator is to furnish proof that the object to which it is attached is genuine. Symmetric key-based authentication uses a secret key and the to-be-authenticated data (message) as input to compute a MAC. The host performs the same computation using the expected secret and the same message data; it then compares its version of the MAC to the one received from the secure authenticator. If both MAC results are identical, the secure authenticator is part of the system.

In this SHA-256 authentication system, the message is a combination of host challenge and data elements stored in the secure authenticator. It is crucial that the challenge is based on random data. A never-changing challenge opens the door to replay attacks using a valid, static MAC that is recorded and replayed instead of a MAC that is instantly computed.

The secure authenticator computes a MAC from the challenge; its secret; memory data; and additional data that together constitute the message (**Figure 3**). If the secure authenticator can generate a valid MAC for any challenge, it is safe to assume that it knows the secret and, therefore, can be considered authentic.

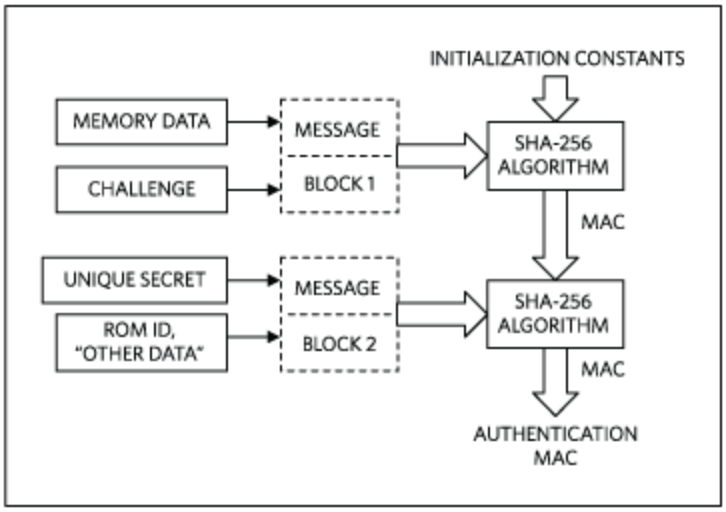

Figure 3. Computing a challenge-and-response authentication MAC.

Data Security (Authenticated Write)

Beyond proving authenticity, it is highly desirable to know that the data stored in the secure authenticator can be trusted. For this purpose, some or all of the EEPROM in the secure authenticator can be authentication protected. With authentication protection activated, memory write access requires that the host presents proof of its authenticity by providing a host authentication MAC to the secure authenticator (**Figure 4**).

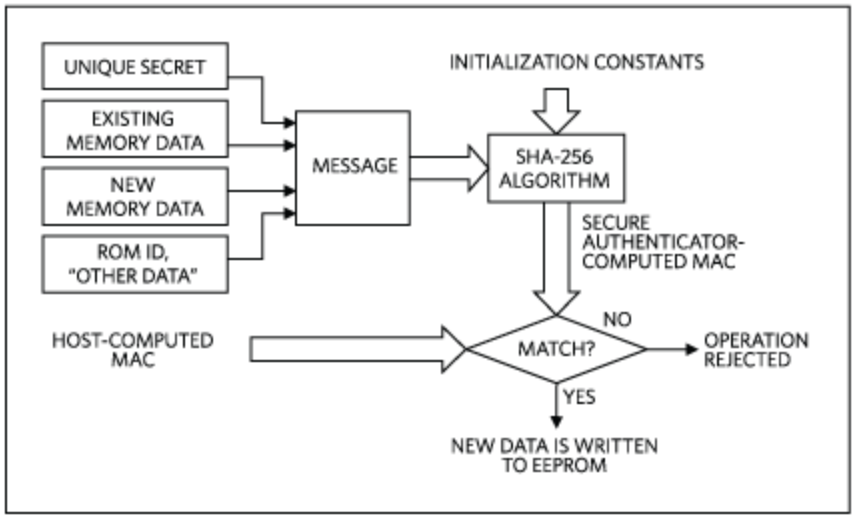

Figure 4. Authenticated write access (host-authentication MAC).

The host-authentication MAC is computed from the new memory data; the existing memory data; the secure authenticator's unique secret plus ROM ID; and other data that together constitute the message. The secure authenticator computes a MAC in the same way.

An authentic host has recreated the secure authenticator's secret and can generate a valid write-access MAC. When receiving the MAC from the host, the secure authenticator compares it to its own result. Data is written to the EEPROM only if both MACs match. User memory areas that are write protected cannot be modified, even if the MAC is correct.

Data Security (Encrypted Read and Write)

Going beyond SHA-256 authenticators in general, where the secret is never exposed, the DS28C22 secure authenticator can be configured so it does not even expose its memory data during memory read and write access. This heightened protection is achieved through data encryption during transit. Inside the chip, the data is stored in the clear as is needed for authentication purposes.

The write-access encryption uses a One-Time-Pad (OTP) that is computed from a host-supplied encryption seed; the secure authenticator's secret; a portion of the authenticator's ROM ID; and other data (padding, formatting, and data address related data). As shown in **Figure 5**, these data elements form a message that is processed according to the SHA-256 algorithm. The resulting message authentication code is the OTP. The host XORs the new memory data with the corresponding data in the OTP before sending it to the authenticator. The authenticator again performs the XOR, restoring the original data that is then programmed to the user EEPROM. The host provides the encryption seed, which should be a random number. This way, even if the host writes the same data over and over again to someone eavesdropping on the I^2C bus, the encrypted data always looks different.

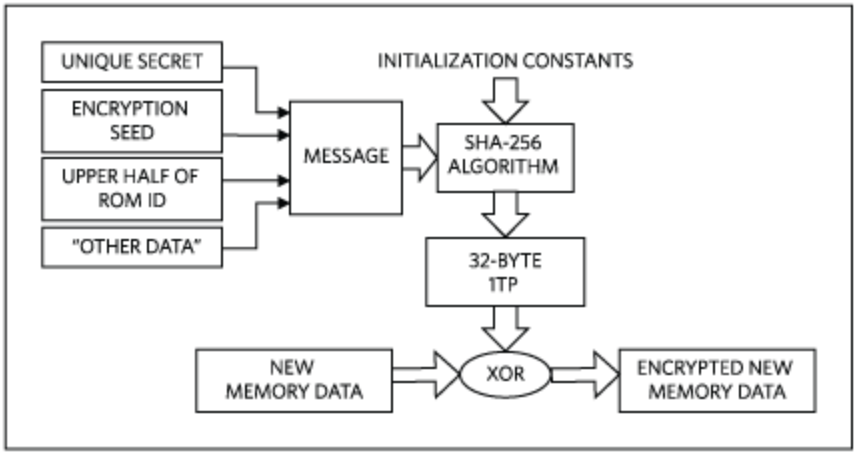

Figure 5. Encrypted write access.

The read-access encryption is most similar to the write-access encryption. Although the data elements of the message are essentially the same, there are differences in the "other data" that cause the read-access OTP to be different from the write-access OTP, even if the other ingredients are identical. As shown in Figure 6, the secure authenticator takes the data from the user memory, XORs it with the OTP, and makes it read accessible to the host. The host then performs the XOR using its version of the OTP. If the host can compute the secure authenticator's secret and the OTP used for encryption, the XOR step successfully

decrypts the data. Again, the host provides the encryption seed, which should be a random number. Now even if the host reads the same data repeatedly to someone eavesdropping on the I^2C bus, the data always looks different.

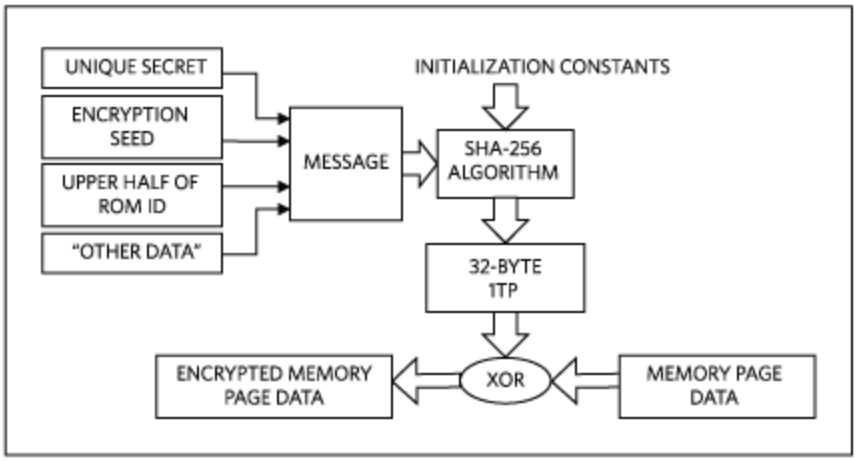

Figure 6. Encrypted read access.

Security (Encrypted Authenticated Write)

Encrypted writing does not prevent a host processor that does not know the secure authenticator's secret from writing to the memory. The data actually written to the memory will be of no use, though. Admittedly, one could maliciously wear out the memory and in this way compromise the authenticator. To prevent this from happening, memory areas that are set up for encryption should either be write protected after the initial writing or authentication protected to allow changes. Then only an authentic host can modify the memory data.

Encrypted authenticated write access consists of two steps. In the first step, the host encrypts the new data as in Figure 5, and then sends it to the secure authenticator. In the second step, the host computes a write authentication MAC as in Figure 4, and then sends it to the secure authenticator. In contrast to authenticated write without encryption, the MAC is now computed from both the existing decrypted memory data and the encrypted new memory data.

Secret Protection

The secure authenticator's secret and the secure coprocessor's master secret are read protected by hardware design. If desired, the secret can be write protected, which prevents tampering with the authenticator's data by replacing an unknown secret with a known secret. After installation, the binding data, typically stored in the coprocessor's memory, should be read protected. This level of protection is effective as long as the coprocessor and authenticator are set up for the application at a trusted production site.

DeepCover Ultimate Security

The deployment of DeepCover technologies from Maxim Integrated provides the strongest affordable protection against any die-level attacks that attempt to discover the secret key. DeepCover technologies include numerous circuits to actively monitor for die-level tamper events, advanced die routing and layout

techniques, and additional proprietary methods to counter the sophisticated capabilities of attackers.

Bidirectional Authentication

The secure authenticator in the example system supports both challenge-and-response authentication and authenticated writes (host authentication). The entire user memory can be used for challenge-and-response authentication. Bidirectional authentication applies to memory areas configured for secure data storage (authenticated write). Data encryption does not impede challenge-and-response authentication. The authentication MAC is always computed from the unencrypted data in the user EEPROM.

Summary

With a size of 256 bits each for the secret, challenge, and MAC, SHA-256 is a significant improvement over older SHA-1 authentication. This article presented a modern, secure authentication system that matches a host system (host controller with SHA-256 secure coprocessor) with a sensor/peripheral module (the SHA-256 secure authenticator). SHA-256 security has never been easier.

General References

1. A general introduction to mutual authentication is found in Maxim Integrated application note 3675, "Protecting the R&D Investment with Secure Authentication".

Related Parts		
DS2465	DeepCover Secure Authenticator with SHA-256 Coprocessor and 1-Wire Master Function	Free Samples
DS28C22	DeepCover Secure Memory with I^2C SHA-256 and 3Kb User EEPROM	Free Samples
DS28C22DEMOK	Authentication Demo Stick Using the DS28C22 authenticator and DS2465 coprocessor	Free Samples

More Information
For Technical Support: http://www.maximintegrated.com/en/support
For Samples: http://www.maximintegrated.com/en/samples
Other Questions and Comments: http://www.maximintegrated.com/en/contact

Application Note 5785: http://www.maximintegrated.com/en/an5785
APPLICATION NOTE 5785, AN5785, AN 5785, APP5785, Appnote5785, Appnote 5785