19-5870; Rev 0; 5/11

Memory-Mapped SHA-1 Coprocessor

General Description

The DSSHA1 coprocessor with 64-byte RAM is a synthesizable register transfer level (RTL) implementation of the FIPS 180-3 Secure Hash Algorithm (SHA-1), eliminating the need to develop software to perform the complex SHA-1 computation required for authenticating SHA-1 devices. The DSSHA1 can compute SHA-1 message authentication codes (MACs) for use with Maxim SHA-1 devices, such as the DS1963S, DS1961S, DS28E10, DS28E02, DS2460, DS28CN01, and DS28E01-100. The device can output the 20-byte MAC result from registers required for comparison against SHA-1 slave devices. When incorporated into a design, DSSHA1 also provides an offloading function, relieving a microcontroller of performing the SHA-1 computation.

Applications

Printer Cartridge Authentication

Clone Prevention of Systems and PCB Designs

License Management

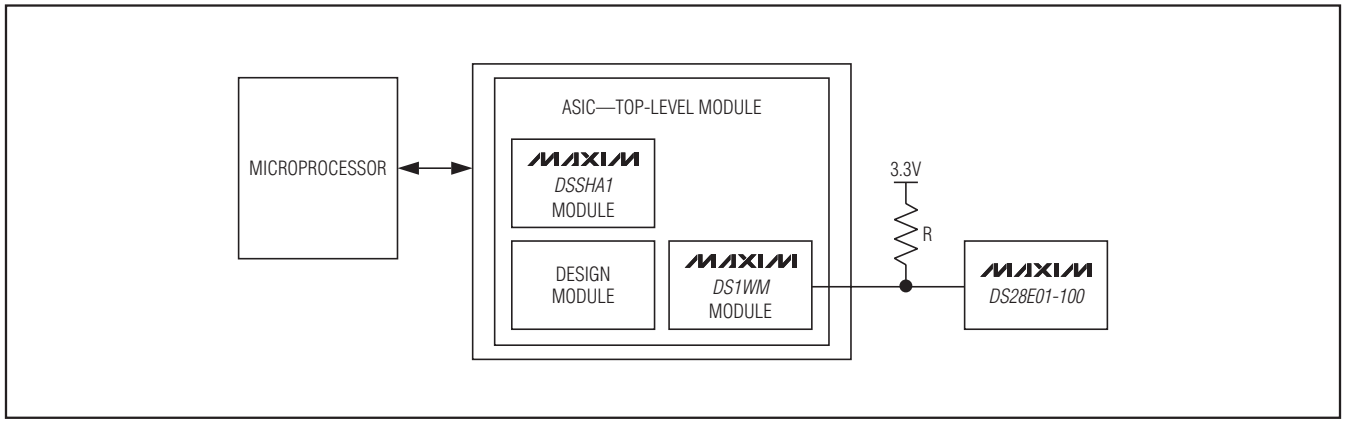
Secure Feature Control of Systems

Network Appliance Authentication

Features

- **SHA-1 Computations Within 670 Clock Cycles (13.4µs at a Typical Frequency of 50MHz)**
- **Area Estimate is 102,256µm^2 in TSMC CL018G (0.18µm Generic Process)**
- **Dedicated Hardware-Accelerated SHA-1 Engine for Generating MACs**
- **64-Byte RAM for Message Input**
- **Five 32-Bit Registers to Read MAC Result**
- **Available in Synthesizable Verilog®**
- **Made as a Low-Level Module to be Instantiated by a Top-Level Module**
- **Includes Test Bench**

Typical Operating Circuit



Verilog is a registered trademark of Gateway Design Automation Corp.

Memory-Mapped SHA-1 Coprocessor

Description

The DSSHA1 is a synthesizable, memory-mapped SHA-1 coprocessor that includes a 64-byte general-purpose RAM that stores the 64-byte message. The input message is used to compute the SHA-1 MAC. The DSSHA1 input and output port signals are designed to internally connect to a 32-bit bus. Implementation in an ASIC or FPGA provides a SHA-1 hash only to be compared to the SHA-1 hash of Maxim devices. By a positive comparison result, authentication security is achieved between a host system and slave accessories. Maxim has numerous SHA-1 slave devices such as the DS28E01-100 and DS28E10 1-Wire® devices, the DS1961S and DS1963S iButton® devices, and the DS28CN01 I^2C device.

Figure 1 shows how the 64-byte SHA-1 message is inserted into the RAM. Triggering the input signal RUN_SHA to logic-high starts the SHA-1 computation. The output BUSY signal indicates an occurring computation. Upon completion of the BUSY signal, the result registers contain the 20-byte message digest for reading.

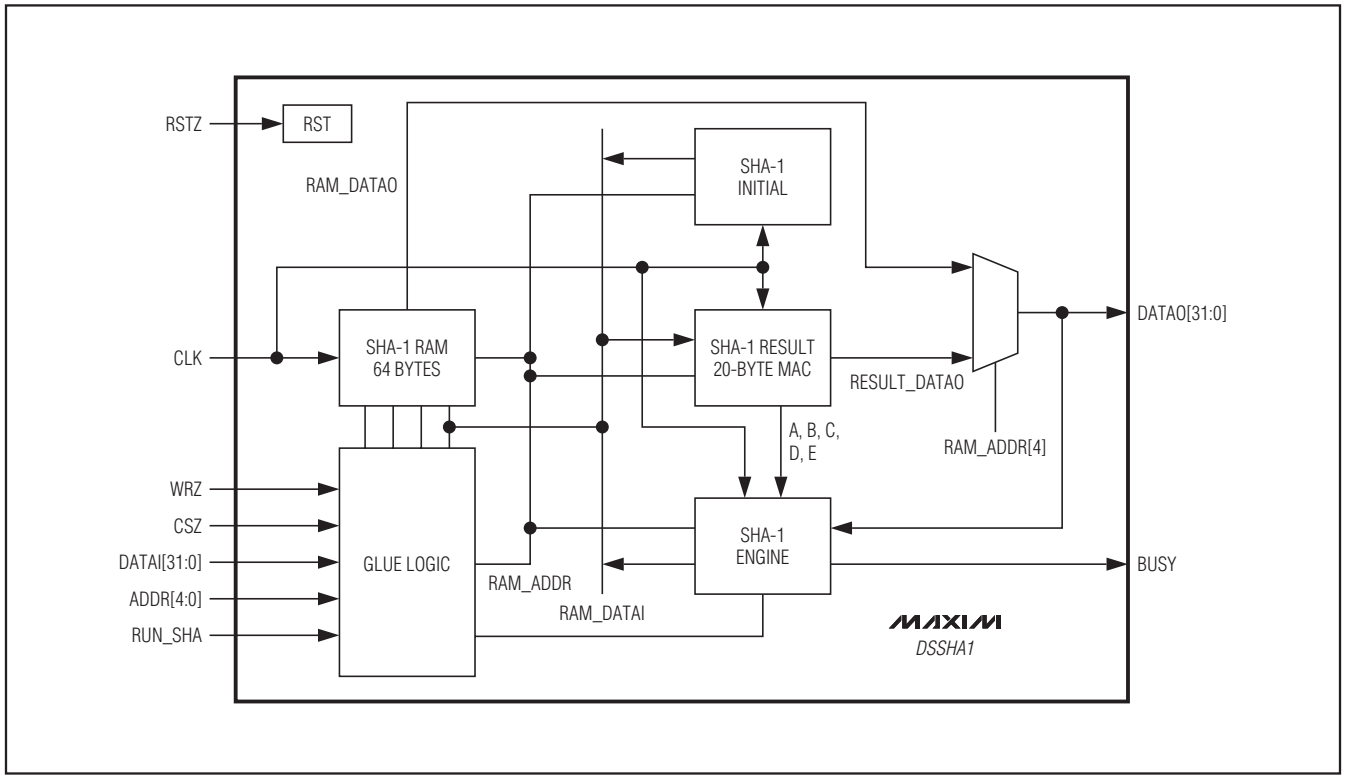


Figure 1. Block Diagram

1-Wire and iButton are registered trademarks of Maxim Integrated Products, Inc.

Memory-Mapped SHA-1 Coprocessor

Signal Description

NAME	TYPE*	FUNCTION
CLK	I	Clock. On the positive edge, data on signals DATAI[31:0] and DATAO[31:0] are clocked in and out.
RSTZ	I	Active-Low Reset. The RSTZ signal is evaluated at each interval of the positive edge of the CLK signal. It is necessary to do a reset before every load of a 512-bit message and MAC computation.
CSZ	I	Active-Low Chip Select. This signal must be low for all accesses to registers and memory.
WRZ	I	Active-Low Write Enable. This signal must be low during all write operations.
ADDR[4:0]	I	Address[4:0]. These five signals are the address signals.
DATAI[31:0]	I	Data Bus Input. These 32 signals are the input data bus.
DATAO[31:0]	O	Data Bus Output. These 32 signals are the output data bus.
BUSY	O	Busy. When high, this signal indicates that the SHA-1 coprocessor is busy performing a computation. There should be no data accesses while this signal is high.
RUN_SHA	I	Run SHA-1. This signal must only be one clock period wide and initiates a SHA-1 computation upon the positive edge of the CLK signal.

**I = Input, O = Output.*

Memory Map

ADDRESS (HEX)	TYPE	ACCESS	FUNCTION
0x00 to 0x0F	RAM	Read/Write	64-Byte Buffer Input. This is the 512-bit input block that usually includes the 64-bit slave device secret and a 448-bit input message consisting of a random challenge and various data.
0x10 to 0x14	Registers	Read	20-Byte Result. This is the MAC for comparison to the received MAC of the SHA-1 slave device.

Detailed Register Description

Input Buffer (00h to 0Fh)

The SHA-1 engine receives the data to be processed through the 64-byte input buffer. This buffer holds the 512-bit message that the SHA-1 engine processes to generate a MAC. Secret and other message data are contained in the input buffer. Security of the secret is a task left for the designer. The format of the data is defined by each Maxim SHA-1 slave device.

MAC Result (10h to 14h)

A 20-byte MAC of a SHA-1 computation resides in the MAC result address space.

Device Operation

The typical use of the DSSHA1 in an application involves writing, reading, and running the SHA-1 engine, and using the MAC result to externally compare this block to the MAC of a 1-Wire SHA-1 device. All these activities are controlled through the 32-bit interface with separate data input and output lines to easily connect to the internal bus inside an ASIC or FPGA. The *SHA-1 Engine Control* section explains the data input and output format and how to instruct the SHA-1 engine to perform a MAC computation.

Memory-Mapped SHA-1 Coprocessor

SHA-1 Engine Control

The DSSHA1 performs the job of a SHA-1 engine. The input buffer accepts the message. The MAC output buffer receives the resultant SHA-1 computation. Figure 2 illustrates data flow into and out of the SHA-1 engine.

Applying a power reset initiates the first step of using the SHA-1 engine. Next, a message is loaded into the input buffer in the format of Table 1. Upon completion of a message load, the user pulses the RUN_SHA input signal. For the duration of the SHA-1 computation, the BUSY signal goes and remains logic-high. A BUSY signal goes logic-low again when the SHA-1 computation completes. All five of the MRR registers (see Table 2) contain the MAC result for reading.

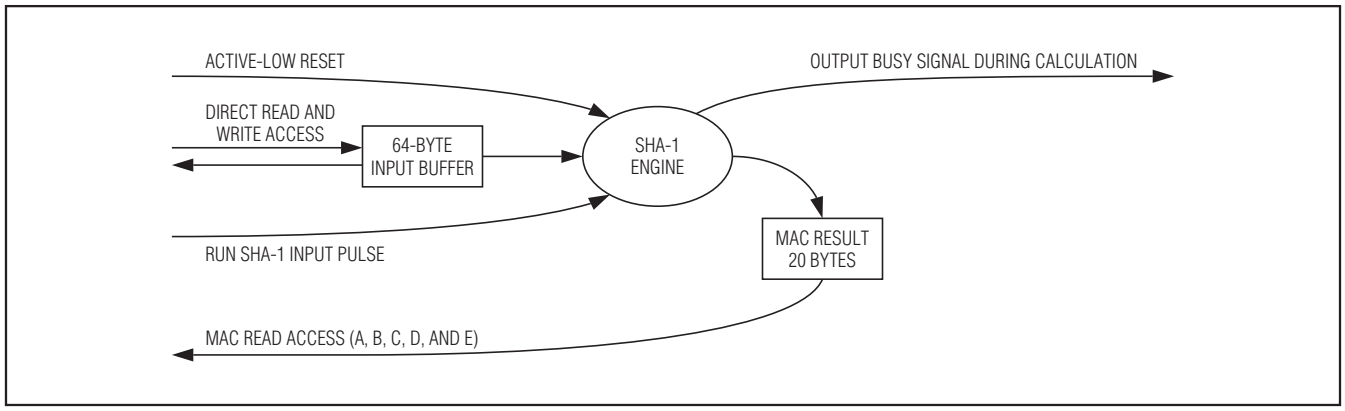


Figure 2. Data Flow Diagram

Table 1. Input Message Format

M0[31:24] = (IB + 0)	M0[23:16] = (IB + 1)	M0[15:8] = (IB + 2)	M0[7:0] = (IB + 3)
M1[31:24] = (IB + 4)	M1[23:16] = (IB + 5)	M1[15:8] = (IB + 6)	M1[7:0] = (IB + 7)
M2[31:24] = (IB + 8)	M2[23:16] = (IB + 9)	M2[15:8] = (IB + 10)	M2[7:0] = (IB + 11)
M3[31:24] = (IB + 12)	M3[23:16] = (IB + 13)	M3[15:8] = (IB + 14)	M3[7:0] = (IB + 15)
M4[31:24] = (IB + 16)	M4[23:16] = (IB + 17)	M4[15:8] = (IB + 18)	M4[7:0] = (IB + 19)
M5[31:24] = (IB + 20)	M5[23:16] = (IB + 21)	M5[15:8] = (IB + 22)	M5[7:0] = (IB + 23)
M6[31:24] = (IB + 24)	M6[23:16] = (IB + 25)	M6[15:8] = (IB + 26)	M6[7:0] = (IB + 27)
M7[31:24] = (IB + 28)	M7[23:16] = (IB + 29)	M7[15:8] = (IB + 30)	M7[7:0] = (IB + 31)
M8[31:24] = (IB + 32)	M8[23:16] = (IB + 33)	M8[15:8] = (IB + 34)	M8[7:0] = (IB + 35)
M9[31:24] = (IB + 36)	M9[23:16] = (IB + 37)	M9[15:8] = (IB + 38)	M9[7:0] = (IB + 39)
M10[31:24] = (IB + 40)	M10[23:16] = (IB + 41)	M10[15:8] = (IB + 42)	M10[7:0] = (IB + 43)
M11[31:24] = (IB + 44)	M11[23:16] = (IB + 45)	M11[15:8] = (IB + 46)	M11[7:0] = (IB + 47)
M12[31:24] = (IB + 48)	M12[23:16] = (IB + 49)	M12[15:8] = (IB + 50)	M12[7:0] = (IB + 51)
M13[31:24] = (IB + 52)	M13[23:16] = (IB + 53)	M13[15:8] = (IB + 54)	M13[7:0] = (IB + 55)
M14[31:24] = (IB + 56)	M14[23:16] = (IB + 57)	M14[15:8] = (IB + 58)	M14[7:0] = (IB + 59)
M15[31:24] = (IB + 60)	M15[23:16] = (IB + 61)	M15[15:8] = (IB + 62)	M15[7:0] = (IB + 63)

Mx = Input buffer of SHA-1 engine; 0 ≤ t ≤ 15; 32-bit words with a start address at 00h and ending address at 0Fh.
IB = Input buffer.

Table 2 shows how the five 32-bit variables A to E that hold the MAC are mapped to the respective locations.

Table 2. Output Message Format

ADDRESS (HEX)	MAC RESULT REGISTERS (MRR)
10	MRR[31:0] = A[31:0]
11	MRR[31:0] = B[31:0]
12	MRR[31:0] = C[31:0]
13	MRR[31:0] = D[31:0]
14	MRR[31:0] = E[31:0]

MAC Comparison

The master has the requirement to test the slave MAC against the DSSHA1 MAC. Authenticity is verified if the slave MAC and the DSSHA1 MAC are equal in value. A fraud is verified if the slave MAC and the DSSHA1 MAC are different.

Functional Verification

To test the DSSHA1, the test message "abc" can verify functionality. This test message with proper padding can be translated into an input block of:

W[0] = 61626380 W[8] = 00000000
W[1] = 00000000 W[9] = 00000000
W[2] = 00000000 W[10] = 00000000
W[3] = 00000000 W[11] = 00000000
W[4] = 00000000 W[12] = 00000000
W[5] = 00000000 W[13] = 00000000
W[6] = 00000000 W[14] = 00000000
W[7] = 00000000 W[15] = 00000018

Using the table format from Table 1, the input block of this test message will be the values in Table 3.

The output of the computation of this block is:

Output[0] = 42541B35
Output[1] = 5738D5E1
Output[2] = 21834873
Output[3] = 681E6DF6
Output[4] = D8FDF6AD

The Maxim devices take these words as most significant word first and the individual bytes as least significant byte (LSB) first. So the device ordering of the output would be:

ADF6FDD8 F66D1E68 73488321 E1D53857 351B5442

Table 3. SHA-1 Input for "abc" Test Packet

M0[31:24] = 61h	M0[23:16] = 62h	M0[15:8] = 63h	M0[7:0] = 80h
M1[31:24] = 00h	M1[23:16] = 00h	M1[15:8] = 00h	M1[7:0] = 00h
M2[31:24] = 00h	M2[23:16] = 00h	M2[15:8] = 00h	M2[7:0] = 00h
M3[31:24] = 00h	M3[23:16] = 00h	M3[15:8] = 00h	M3[7:0] = 00h
M4[31:24] = 00h	M4[23:16] = 00h	M4[15:8] = 00h	M4[7:0] = 00h
M5[31:24] = 00h	M5[23:16] = 00h	M5[15:8] = 00h	M5[7:0] = 00h
M6[31:24] = 00h	M6[23:16] = 00h	M6[15:8] = 00h	M6[7:0] = 00h
M7[31:24] = 00h	M7[23:16] = 00h	M7[15:8] = 00h	M7[7:0] = 00h
M8[31:24] = 00h	M8[23:16] = 00h	M8[15:8] = 00h	M8[7:0] = 00h
M9[31:24] = 00h	M9[23:16] = 00h	M9[15:8] = 00h	M9[7:0] = 00h
M10[31:24] = 00h	M10[23:16] = 00h	M10[15:8] = 00h	M10[7:0] = 00h
M11[31:24] = 00h	M11[23:16] = 00h	M11[15:8] = 00h	M11[7:0] = 00h
M12[31:24] = 00h	M12[23:16] = 00h	M12[15:8] = 00h	M12[7:0] = 00h
M13[31:24] = 00h	M13[23:16] = 00h	M13[15:8] = 00h	M13[7:0] = 00h
M14[31:24] = 00h	M14[23:16] = 00h	M14[15:8] = 00h	M14[7:0] = 00h
M15[31:24] = 00h	M15[23:16] = 00h	M15[15:8] = 00h	M15[7:0] = 18h

Mx = Input buffer of SHA-1 engine; 0 ≤ t ≤ 15; 32-bit words with a start address at 00h and ending address at 0Fh.

Memory-Mapped SHA-1 Coprocessor

Timing Specification

Figure 3, Figure 4, and Table 4 show delay values measured from 50% of supply to 50% of supply using an ARM TSMC CL018G (0.18µm generic process) 1.8V SAGE-X™ standard cells library, version 2004q3v1, at +25°C. The output signals are not loaded. Input signals are driven with a standard slew of 0.200ns from 10% to 90% of supply.

Table 4. Data Bus Interface Timing

PARAMETER	SYMBOL	MIN	MAX	UNITS
CLK Cycle (Note 1)	t_{CYC}	12.500		ns
Chip Select Setup Before Rising Edge of CLK (Note 1)	t_{CSS}	0.229		ns
Chip Select Hold After Rising Edge of CLK (Note 1)	t_{CSH}	0.000		ns
Address and Data Setup Before Rising Edge of CLK (Note 1)	t_{AS}	0.229		ns
Address and Data Hold After Rising Edge of CLK (Note 1)	t_{AH}	0.000		ns
Active Output Time to DATAO Valid (Notes 1, 2)	t_{AO}		0.984	ns
Deactivate DATAO[31:0] (Note 1)	t_D		0.984	ns

Note 1: These values depend upon the process used to realize the circuit. Values shown are for example purposes only and modeled using the ARM TSMC CL018G (0.18µm generic process) 1.8V SAGE-X standard cells library, version 2004q3v1. The ARM part number is A0082.

Note 2: This time is defined as the longest possible delay to valid output for the typical corners.

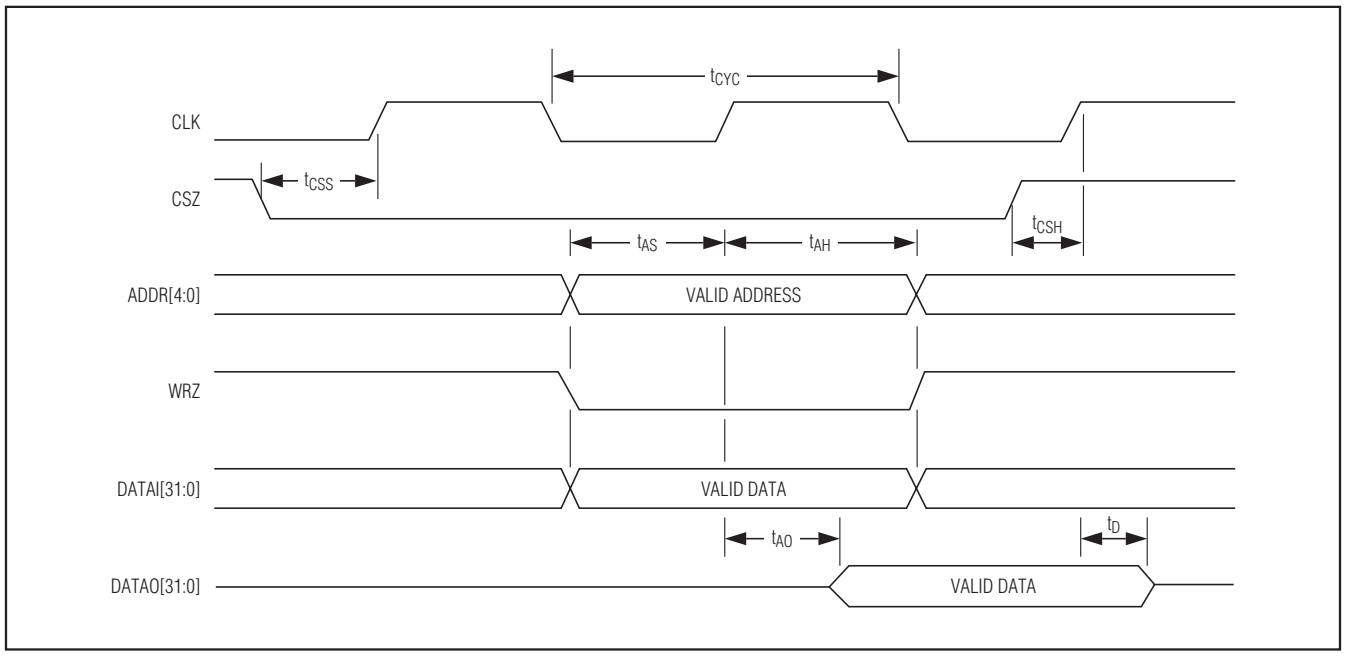


Figure 3. Write Cycle

SAGE-X is a trademark of ARM Ltd.

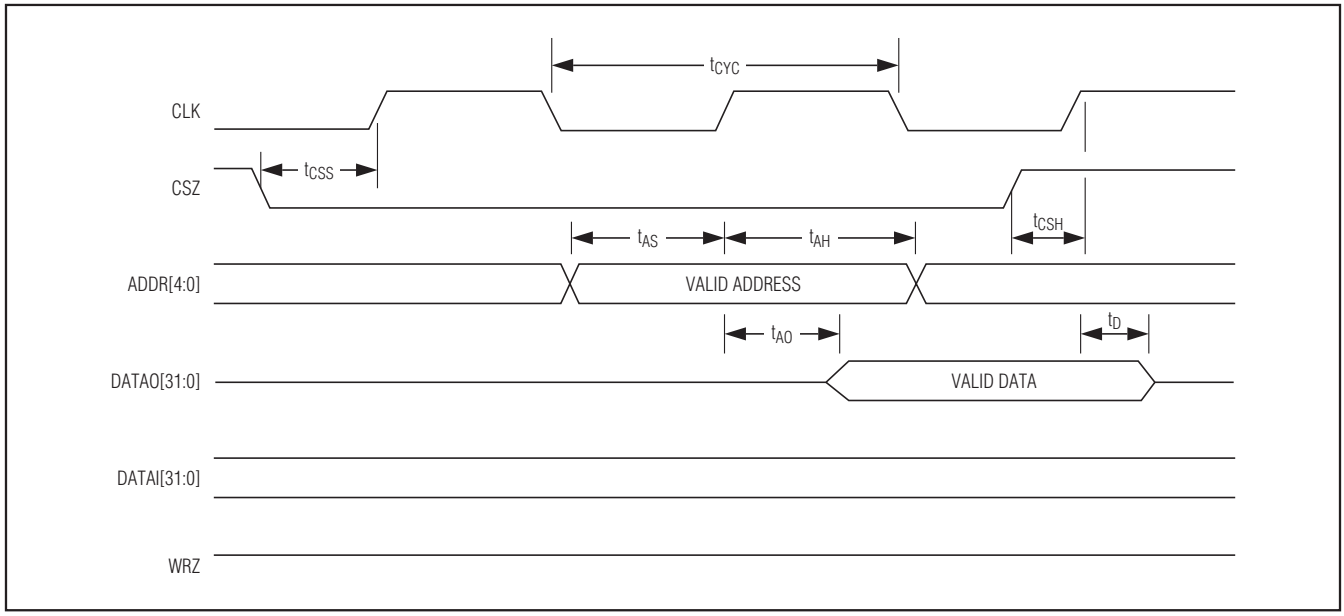


Figure 4. Read Cycle

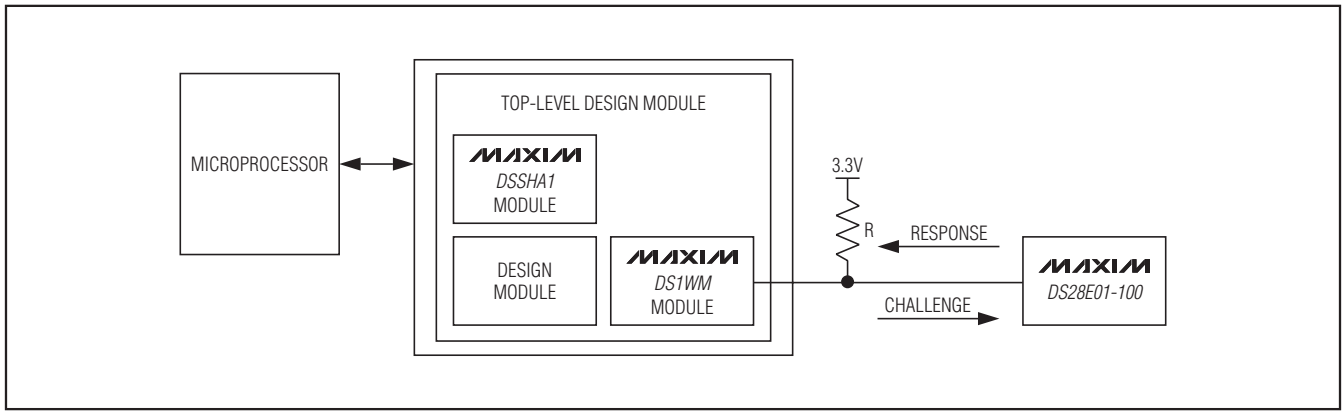


Figure 5. Typical FPGA or ASIC Application

Applications Information

FPGAs or ASICs integrate the designed DSSHA1. Using several modules, an achievable authentication method makes a design secure. In Figure 5, the design module with a microprocessor can offload the SHA-1 computation to the DSSHA1. In Figure 5, the designer first crafts a randomly generated challenge and compares the result from DSSHA1 to the response received from the DS28E01. If the result and response match, then the design has been authenticated and can enable the product's functionality. It is often desirable to make variations in the authentication process in software and hardware. This makes successful attacks less likely. Refer to Application Note 1098: *White Paper 3: Why are 1-Wire SHA-1 Devices Secure?* for more information.

Memory-Mapped SHA-1 Coprocessor

Physical Estimates

- Gate count 6,423 (NAND 2x1 used for calculation).
- Area is 85,470µm^2 without routing.
- Area is 102,256µm^2 with routing estimate.

Library used for estimate:

ARM TSMC CL018G (0.18um generic process) 1.8V SAGE-X standard cells library, version 2004q3v1. The ARM part number is A0082. This is a free, foundry-sponsored library that can be obtained at: www.arm.com/products/physicalip/productsservices.html.

Verification

The industry typically denotes the level of verification of an IP block with the following conventions:

- Gold IP has been to target silicon.
- Silver IP has been to target silicon in FPGA.
- Bronze IP has been verified in silicon models with logical timing closure.
- In-development IP has not yet been verified.

Note: The DSSHA1 has achieved silver status.

Deliverables

The DSSHA1 package comes complete with:

- Verilog HDL
- Verilog Test Bench
- Readme Information on Setup and Scripts

The free DSSHA1 IP is available by request at **https://support.maxim-ic.com/1-Wire**.

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	5/11	Initial release	—