



Using MachXO3D ESB to Implement ECDSA Generation and Verification

Reference Design

FPGA-RD-02053-1.1

April 2024

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Acronyms in This Document	4
1. Introduction	5
2. Reference Design Overview	6
2.1. Block Diagram	6
2.2. Overview.....	7
3. Functional Description	8
3.1. Input/Output of the Design	8
4. Design Description	10
4.1. ESB Registers for ECDSA Generation/Verification	10
5. HDL Simulation and Verification	14
6. Implementation	15
References.....	16
Technical Support Assistance	17
Revision History	18

Figures

Figure 2.1. Top-Level Block Diagram for ECDSA Generation	6
Figure 2.2. Top-Level Block Diagram for ECDSA Verification	6
Figure 3.1. I/O Diagram of ECDSA Generation/Verification Reference Design	8
Figure 4.1. ECDSA Generation Algorithm	12
Figure 4.2. ECDSA Verification Algorithm	13
Figure 5.1. ECDSA Generation: Data Input and Mode Setting.....	14
Figure 5.2. ECDSA Generation: Calculation Done and Data Output	14
Figure 5.3. ECDSA Verification: Data Input and Mode Setting	14
Figure 5.4. ECDSA Verification: Calculation Done and Data Output	14

Tables

Table 3.1. Port Definitions for ECDSA Generation and Verification	9
Table 4.1. Register Descriptions	10

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
DSA	Digital Signature Algorithm
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
ESB	Embedded Security Block

1. Introduction

In cryptography, the Elliptic Curve Digital Signature Algorithm (ECDSA) offers a variant of the Digital Signature Algorithm (DSA) which uses elliptic curve cryptography (ECC).

In ECC, the private key can be used to create a digital signature for any piece of data using a digital signature algorithm. This typically involves taking a cryptographic hash of the data and operating on it mathematically using the private key. Anyone with the public key can check if this signature is created using the private key.

Security is the new key feature for Lattice MachXO3D™ device family. With the Embedded Security Block (ESB) module in MachXO3D chips, ECDSA can be used in the generation and verification of bitstreams, or any user messages. In this reference design, the generation and verification functions are provided as separate reference designs.

2. Reference Design Overview

2.1. Block Diagram

Figure 2.1 and Figure 2.2 show the block diagrams of the ECDSA generation and the ECDSA verification reference designs.

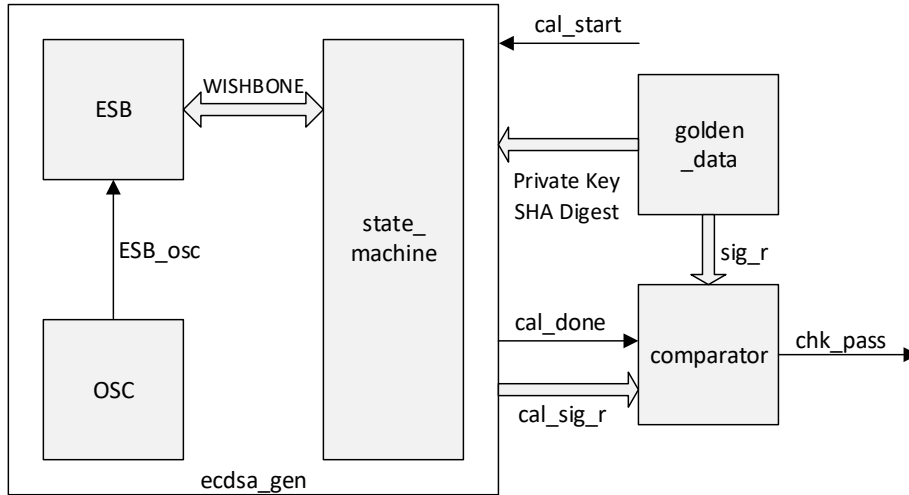


Figure 2.1. Top-Level Block Diagram for ECDSA Generation

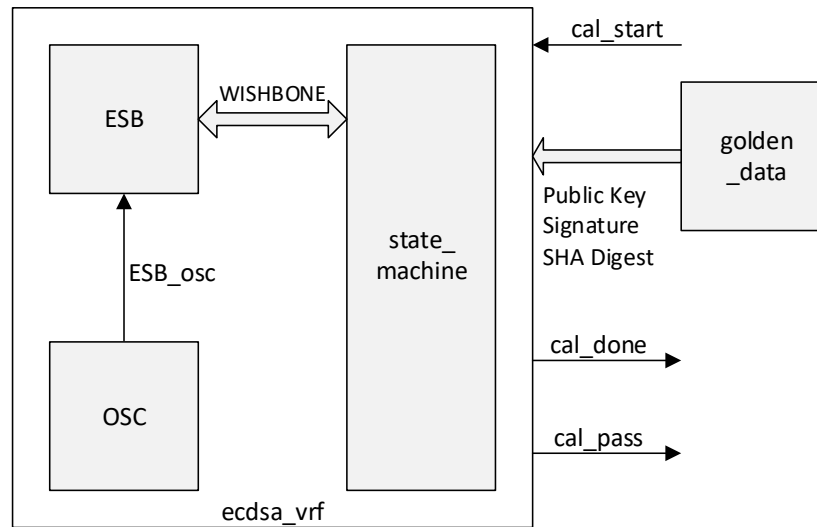


Figure 2.2. Top-Level Block Diagram for ECDSA Verification

2.2. Overview

The ECDSA generation design is used to create a digital signature for SHA digest using a DSA, while the ECDSA verification design can be used to verify the digital signature. The features include support for:

- Programmable 32-byte private key for ECDSA generation
- 32-byte SHA digest input for ECDSA generation
- Programmable 64-byte public key(X, Y) for ECDSA verification
- 64-byte digital signature(R, S) input for ECDSA verification

3. Functional Description

In the ECDSA generation reference design, with a 32-byte private key and a 32-byte digest input, after the calculation is done, the digital signature pair (R, S) is created.

In the ECDSA verification reference design, with a 64-byte public key pair (X, Y), a 64-byte digital signature pair (R, S), and a 32-byte digest input, after the calculation is done, *cal_pass* shows if the verification passes.

The definitions of I/O ports for these two designs are listed in [Table 3.1](#).

3.1. Input/Output of the Design

[Figure 3.1](#) shows the I/O diagram of the ECDSA generation and verification reference design.

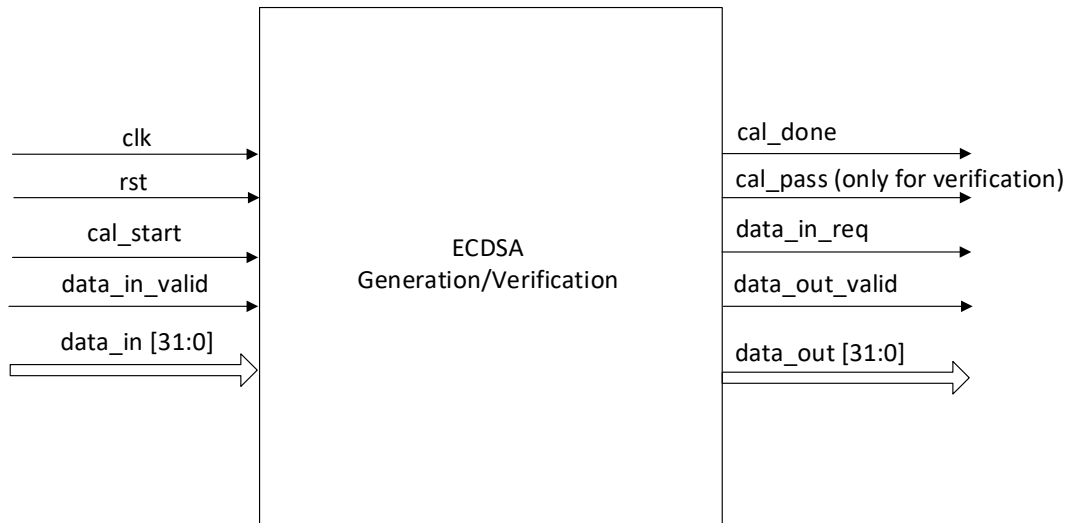


Figure 3.1. I/O Diagram of ECDSA Generation/Verification Reference Design

Table 3.1. Port Definitions for ECDSA Generation and Verification

Ports	Width	I/O	Description
Clock and Reset			
clk	1	I	System clock.
rst	1	I	Asynchronous reset. Active High.
Control and Status			
cal_start	1	I	ECDSA calculation starts. The rising edge triggers the calculation process.
cal_done	1	O	ECDSA calculation done. Active High.
cal_pass (only for verification)	1	O	ECDSA verification pass. Active High.
Input Data Interface			
data_in_req	1	O	Request flag for input data. Active High.
data_in_valid	1	I	Input data valid. Active High.
data_in	32	I	Input data : For generation, 32-byte private_key and 32-byte digest; For verification, 32-byte public_X_key, 32-byte public_Y_key, 32-byte signature_r, 32-byte signature_s, 32-byte SHA digest.
Output Data Interface			
data_out_valid	1	O	Output data valid. Active High.
data_out	32	O	Output data: For generation, 32-byte calculated signature_r and 32- byte calculated signature_s; For verification, 32-byte calculated signature_r.

4. Design Description

4.1. ESB Registers for ECDSA Generation/Verification

Table 4.1. Register Descriptions

Register Type	Register Name	Address	Read/Write	Description
Control Register	r0_gp0	18'h2_0020	Read	Check for busy status of ESB 0xB0: READY to get a new command 0xB2: ESB operation done
	ri_ctrl1	18'h2_000c	Write	ESB function: 0x00: Clear the ESB function request 0x0C: ECDSA generation 0x0D: ECDSA verification
Memory for Generation	Prv_key_0	18'h1_F800	Write	Prv_Key [31:0]
	Prv_key_1	18'h1_F804	Write	Prv_Key [63:32]
	Prv_key_2	18'h1_F808	Write	Prv_Key [95:64]
	Prv_key_3	18'h1_F80C	Write	Prv_Key [127:96]
	Prv_key_4	18'h1_F810	Write	Prv_Key [159:128]
	Prv_key_5	18'h1_F814	Write	Prv_Key [191:160]
	Prv_key_6	18'h1_F818	Write	Prv_Key [223:192]
	Prv_key_7	18'h1_F81C	Write	Prv_Key [255:224]
	Sha_res_0	18'h1_F860	Write	Sha_Msg [31:0]
	Sha_res_1	18'h1_F864	Write	Sha_Msg [63:32]
	Sha_res_2	18'h1_F868	Write	Sha_Msg [95:64]
	Sha_res_3	18'h1_F86C	Write	Sha_Msg [127:96]
	Sha_res_4	18'h1_F870	Write	Sha_Msg [159:128]
	Sha_res_5	18'h1_F874	Write	Sha_Msg [191:160]
	Sha_res_6	18'h1_F878	Write	Sha_Msg [223:192]
	Sha_res_7	18'h1_F87C	Write	Sha_Msg [255:224]
	C_sig_r_0	18'h1_F820	Read	Cal_Sig_R [31:0]
	C_sig_r_1	18'h1_F824	Read	Cal_Sig_R [63:32]
	C_sig_r_2	18'h1_F828	Read	Cal_Sig_R [95:64]
	C_sig_r_3	18'h1_F82C	Read	Cal_Sig_R [127:96]
	C_sig_r_4	18'h1_F830	Read	Cal_Sig_R [159:128]
	C_sig_r_5	18'h1_F834	Read	Cal_Sig_R [191:160]
	C_sig_r_6	18'h1_F838	Read	Cal_Sig_R [223:192]
	C_sig_r_7	18'h1_F83C	Read	Cal_Sig_R [255:224]
	C_sig_s_0	18'h1_F840	Read	Cal_Sig_S [31:0]
	C_sig_s_1	18'h1_F844	Read	Cal_Sig_S [63:32]
	C_sig_s_2	18'h1_F848	Read	Cal_Sig_S [95:64]
	C_sig_s_3	18'h1_F84C	Read	Cal_Sig_S [127:96]
	C_sig_s_4	18'h1_F850	Read	Cal_Sig_S [159:128]
	C_sig_s_5	18'h1_F854	Read	Cal_Sig_S [191:160]
	C_sig_s_6	18'h1_F858	Read	Cal_Sig_S [223:192]
	C_sig_s_7	18'h1_F85C	Read	Cal_Sig_S [255:224]
	Pub_key_Qx_0	18'h1_F800	Write	Public_Key_X [31:0]
	Pub_key_Qx_1	18'h1_F804	Write	Public_Key_X [63:32]
	Pub_key_Qx_2	18'h1_F808	Write	Public_Key_X [95:64]
	Pub_key_Qx_3	18'h1_F80C	Write	Public_Key_X [127:96]

Register Type	Register Name	Address	Read/Write	Description
Memory for Verification	Pub_key_Qx_4	18'h1_F810	Write	Public_Key_X [159:128]
	Pub_key_Qx_5	18'h1_F814	Write	Public_Key_X [191:160]
	Pub_key_Qx_6	18'h1_F818	Write	Public_Key_X [223:192]
	Pub_key_Qx_7	18'h1_F81C	Write	Public_Key_X [255:224]
	Pub_key_Qy_0	18'h1_F820	Write	Public_Key_Y [31:0]
	Pub_key_Qy_1	18'h1_F824	Write	Public_Key_Y [63:32]
	Pub_key_Qy_2	18'h1_F828	Write	Public_Key_Y [95:64]
	Pub_key_Qy_3	18'h1_F82C	Write	Public_Key_Y [127:96]
	Pub_key_Qy_4	18'h1_F830	Write	Public_Key_Y [159:128]
	Pub_key_Qy_5	18'h1_F834	Write	Public_Key_Y [191:160]
	Pub_key_Qy_6	18'h1_F838	Write	Public_Key_Y [223:192]
	Pub_key_Qy_7	18'h1_F83C	Write	Public_Key_Y [255:224]
	Sig_r_0	18'h1_F840	Write	Sig_R [31:0]
	Sig_r_1	18'h1_F844	Write	Sig_R [63:32]
	Sig_r_2	18'h1_F848	Write	Sig_R [95:64]
	Sig_r_3	18'h1_F84C	Write	Sig_R [127:96]
	Sig_r_4	18'h1_F850	Write	Sig_R [159:128]
	Sig_r_5	18'h1_F854	Write	Sig_R [191:160]
	Sig_r_6	18'h1_F858	Write	Sig_R [223:192]
	Sig_r_7	18'h1_F85C	Write	Sig_R [255:224]
	Sig_s_0	18'h1_F860	Write	Sig_S [31:0]
	Sig_s_1	18'h1_F864	Write	Sig_S [63:32]
	Sig_s_2	18'h1_F868	Write	Sig_S [95:64]
	Sig_s_3	18'h1_F86C	Write	Sig_S [127:96]
	Sig_s_4	18'h1_F870	Write	Sig_S [159:128]
	Sig_s_5	18'h1_F874	Write	Sig_S [191:160]
	Sig_s_6	18'h1_F878	Write	Sig_S [223:192]
	Sig_s_7	18'h1_F87C	Write	Sig_S [255:224]
	Sha_res_0	18'h1_F880	Write	Sha_Msg [31:0]
	Sha_res_1	18'h1_F884	Write	Sha_Msg [63:32]
	Sha_res_2	18'h1_F888	Write	Sha_Msg [95:64]
	Sha_res_3	18'h1_F88C	Write	Sha_Msg [127:96]
	Sha_res_4	18'h1_F890	Write	Sha_Msg [159:128]
	Sha_res_5	18'h1_F894	Write	Sha_Msg [191:160]
	Sha_res_6	18'h1_F898	Write	Sha_Msg [223:192]
	Sha_res_7	18'h1_F89C	Write	Sha_Msg [255:224]
	C_sig_r_0	18'h1_F880	Read	Cal_Sig_R [31:0]
	C_sig_r_1	18'h1_F884	Read	Cal_Sig_R [63:32]
	C_sig_r_2	18'h1_F888	Read	Cal_Sig_R [95:64]
	C_sig_r_3	18'h1_F88C	Read	Cal_Sig_R [127:96]
	C_sig_r_4	18'h1_F890	Read	Cal_Sig_R [159:128]
	C_sig_r_5	18'h1_F894	Read	Cal_Sig_R [191:160]
C_sig_r_6	18'h1_F898	Read	Cal_Sig_R [223:192]	
C_sig_r_7	18'h1_F89C	Read	Cal_Sig_R [255:224]	

Before starting the ECDSA generation, check the status of the ESB module, and make sure that it is ready to get a new command. Then transfer the 32-byte private key and the 32-byte SHA digest to the ESB module, and enter the ECDSA generation mode. After the calculation, *Done* indicates in the ESB status. Now, the encrypted signature (r, s) is available.

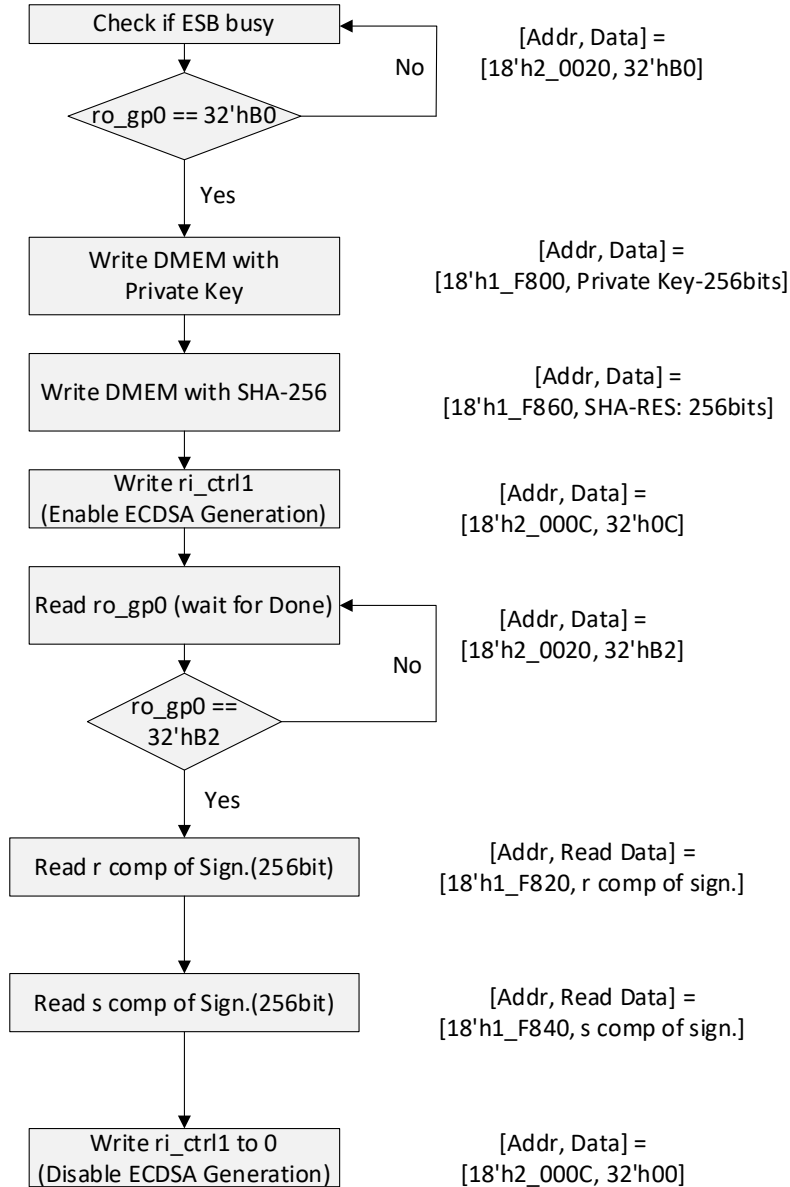


Figure 4.1. ECDSA Generation Algorithm

To verify if one bitstream is from the legitimate source, we can transfer the 64-byte public key, the signature, and the SHA digest to the ESB module. Then enter the ECDSA verification mode. When the calculation is done, check the calculated signature *r*. The verification passes only when the calculated *r* matches the input *r*. Otherwise, the verification fails.

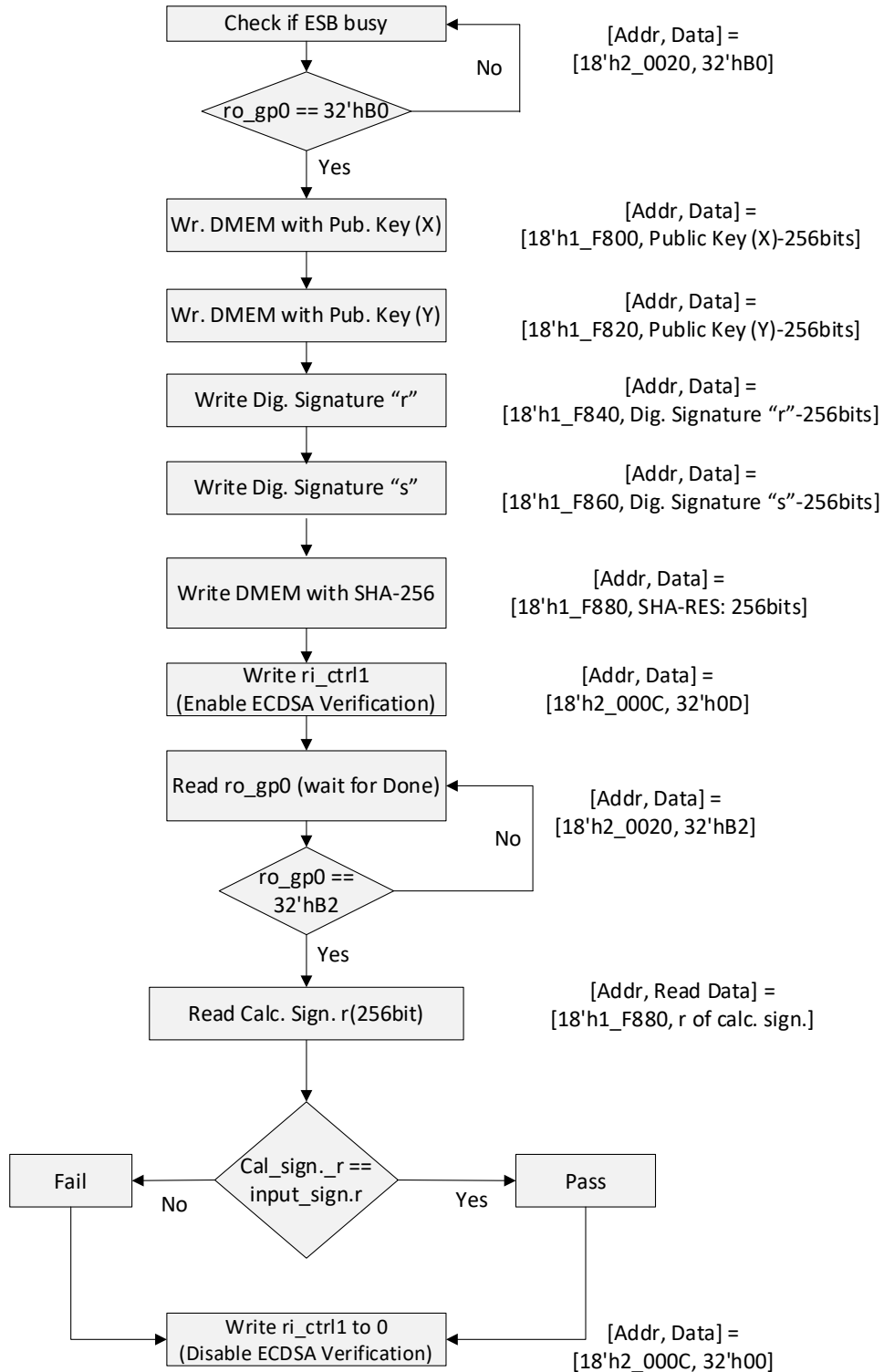


Figure 4.2. ECDSA Verification Algorithm

5. HDL Simulation and Verification

Active-HDL is used for the function simulation of the ECDSA generation/verification designs.

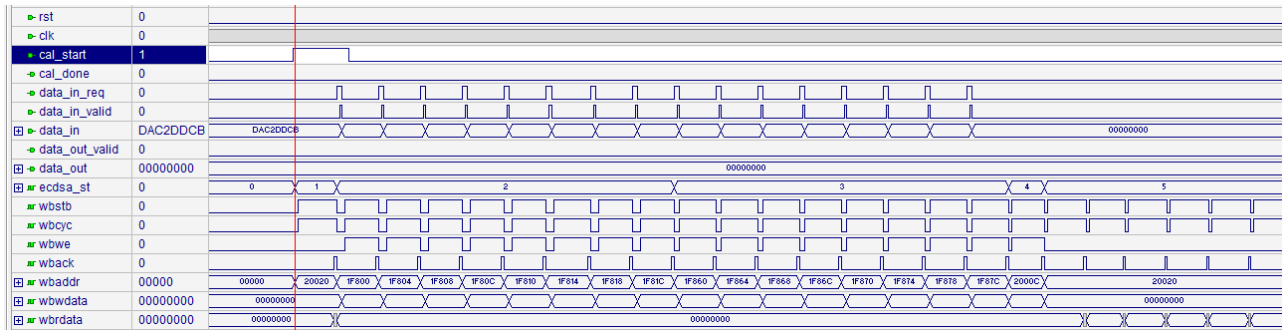


Figure 5.1. ECDSA Generation: Data Input and Mode Setting

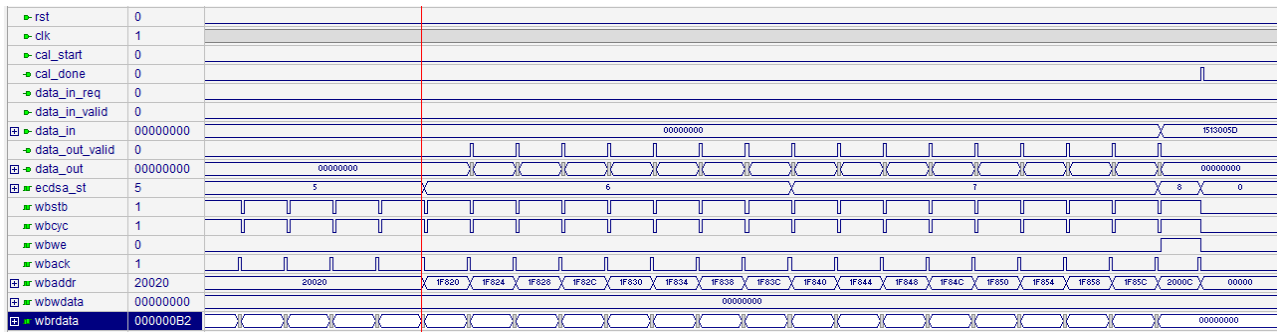


Figure 5.2. ECDSA Generation: Calculation Done and Data Output

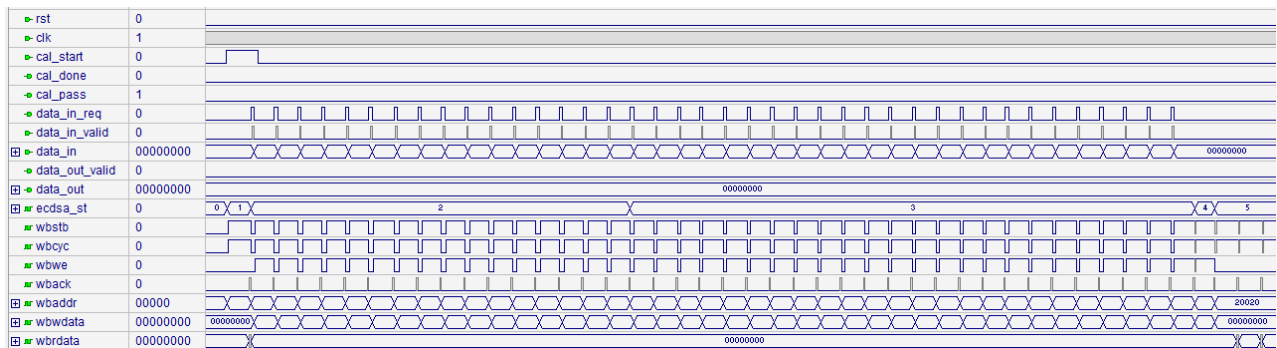


Figure 5.3. ECDSA Verification: Data Input and Mode Setting

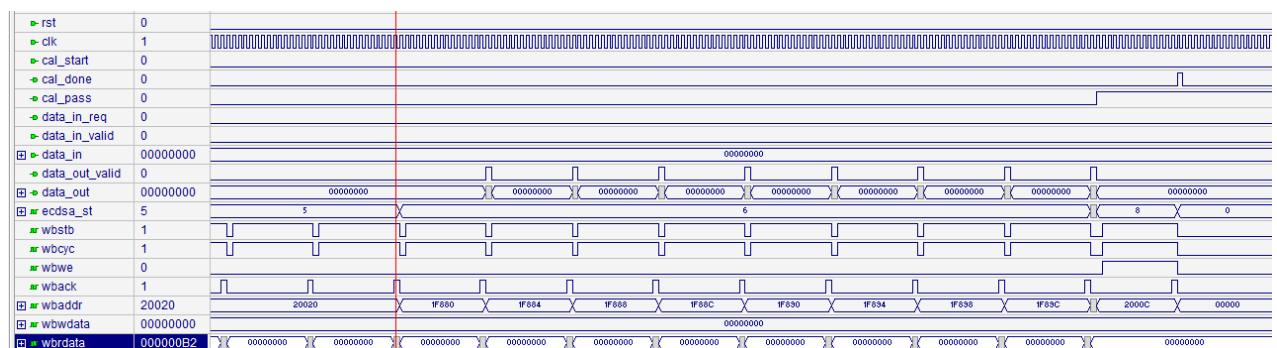


Figure 5.4. ECDSA Verification: Calculation Done and Data Output

6. Implementation

This reference design is implemented in Verilog HDL using Lattice Diamond® software. The synthesis tool is set to be Synplify Pro®. When using these two designs in a different device, density, speed, or grade, performance and utilization may vary.

Table 6.1. Performance and Resource Utilization for ECDSA Generation Design

Device Family	Language	Utilization	ESB Primitive	OSC Primitive	Fmax	Number of I/O
LCMXO3D-9400HC	Verilog HDL	146 LUTs	Yes	Yes	> 50 MHz	71

Table 6.2. Performance and Resource Utilization for ECDSA Verification Design

Device Family	Language	Resources	ESB Primitive	OSC Primitive	Fmax	Number of I/O
LCMXO3D-9400HC	Verilog HDL	807 LUTs	Yes	Yes	> 50 MHz	72

Note: Performance and utilization characteristics are generated for the LCMXO3D-9400HC device using the Lattice Diamond software 3.11.

References

- MachXO3D Embedded Security Block (FPGA-TN-02091)
- [MachXO3D](#) web page
- [Lattice Diamond Software](#) web page
- [Lattice Insights](#) for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.1, April 2024

Section	Change Summary
Disclaimers	Updated disclaimers.
Design Description	Updated the following diagrams: <ul style="list-style-type: none">• Figure 4.1. ECDSA Generation Algorithm• Figure 4.2. ECDSA Verification Algorithm
References	Updated references.
Technical Support Assistance	Added link to the Lattice Answer Database.

Revision 1.0, August 2021

Section	Change Summary
All	Production release.



www.latticesemi.com