Spansion Flash Memory Test Cases List
S28HS512T

Table of Contents

1 References....5
2 Introduction....5
3 Test Sequences....6
3.1 Power Up....6
3.1.1 Series 1. Power Up Negative Test....6
a. POWERUP_1_a....6
3.1.2 Series 2. Power Up Positive Test....6
a. POWERUP_2_a....6
3.2 Write Enable/Write Disable....7
3.2.1 Series 3. Positive Test....7
a. WREN_3_a....7
b. WRDI_3_b....7
3.3 Initial configuration (TBPARM)....7
3.3.1 Series 4. Positive Test....7
a. TBPARM_4_a....7
b. TBPARM_4_b....8
3.4 Block Protection Tests....8
3.4.1 Series 5. Positive Test....8
a. PROT_5_a....8
3.5 Initial configuration (TBPROT)....10
3.5.1 Series 6. TBPROT Tests....10
a. TBPROT_6_a....10
3.6 Read....10
3.6.1 Series 7. Positive Tests....10
a. READ_7_a....10
d. READ_7_b....10
3.7 Read Registers....11
3.7.1 Series 8. Positive Tests....11
a. READ_SR1_8_a....11
3.8 Read OTP....13
3.8.1 Series 9. Positive Tests....13
a. READ_OTP_9_a....13
3.9 Read ID....14
3.9.1 Series 10. Positive Tests....14
a. READ_JID_10_a....14
b. READ_SFDP_10_b....14
3.10 Software Reset....15
3.10.1 Series 11. Software Reset Tests....15
a. RST_11_a....15
b. RST_11_b....15
3.11 OTP Program - Positive Tests....15
3.11.1 Series 12. Positive Tests....15
a. OTPPROG_12_a....15
b. OTPPROG_12_b....16
3.12 OTP Program - Negative Tests....16
3.12.1 Series 13. Negative Tests....16
a. OTPPROG_13_a....16
b. OTPPROG_13_b....17
3.13 Page Programming (3 bytes address)....17
3.13.1 Series 14. Positive Tests....17
a. PGPROG_14_a....17
c. PGPROG_14_c....18

3.14 Program Suspend.....19
3.14.1 Series 15. Positive Test.....19
a. PGSUSP_15_a.....19
3.15 PPB Bit Programming.....19
3.15.1 Series 16. PPB Bits Tests.....19
a. PGPPB_16_a.....19
3.16 PPB Erase.....19
3.16.1 Series 17. PPB Bits Tests.....19
a. PPBERS_17_a.....19
3.17 DYB Bit Programming.....20
3.17.1 Series 18. DYB Bit Programming.....20
a. DYBPG_18_a.....20
3.18 Sector Erase.....20
3.18.1 Series 19. Positive Tests.....20
a. SECERS_19_a.....20
3.19 Erase Suspend.....21
3.19.1 Series 20. Positive Tests.....21
a. ERSSUSP_20_a.....21
3.20 Program During Erase Suspend.....21
3.20.1 Series 21. Positive Tests.....21
a. ERSSUSPPG_21_a.....21
3.21 Software Reset.....22
3.21.1 Series 22 Positive Tests.....22
a. RESET_22_a.....22
b. RESET_22_b.....22
c. RESET_22_c.....22
3.22 Bulk Erase.....23
3.22.1 Series 23. Negative Tests.....23
a. BULKERS_23_a.....23
3.22.2 Series 23. Positive Tests.....23
a. BULKERS_23_a.....23
3.23DPD.....23
3.23.1 Series 24. Positive Tests.....23
a. DPD_24_a.....23
b. DPD_24_b.....23
c. DPD_24_c.....24
3.24CRC ASO.....24
3.24.1 Series 25. Positive Tests.....24
a. CRCASO_25_a.....24
b. CRCASO_25_b.....24
c. CRCASO_25_c.....24
d. CRCASO_25_d.....25
e. CRCASO_25_e.....25
3.26 Persistent Protection Mode.....25
3.26.1 Series 26. Positive Tests.....25
a. PSTPROT_26_a.....25
3.27 Password Protection Mode.....26
3.27.1 Series 27. Positive Tests.....26
a. PSWPROT_27_a.....26
b. PSWPROT_27_b.....26
c. PSWPROT_27_c.....26
3.28 Series 28. Password ProgramSPI.....27
3.28.1 Series 28. Positive Tests.....27
a. PSWPG_28_a.....27

b. PSWPG_28_b.....27
3.29 Series 29. Password ProgramOPI.....27
3.29.1 Series 29. Positive Tests.....27
a. PSWPG_OPI_29_a.....27
b. PSWPG_OPI_29_b.....27
c. PSWPG_OPI_29_c.....28
3.30.2 Series 30. Negative Tests.....28
a. PSWPROT_30_a.....28
b. PSWPROT_30_b.....28
3.31 SectorEraseCounter.....28
a. SECERASE_CNT_31_a.....29
3.32 Jedec Reset.....29
a. JEDEC_RST _32_a.....29

1 References

[1] 002-18216_0F_V.pdf Revised March 26, 2018
[2] spansion_flash_testbench_struct.pdf

2 Introduction

This document defines the test cases that will be used to verify the functional behavior and timing of the Flash Memory S28HS512TVITAL model.
Functional behavior and timing is described in the appropriate S28HS512T data sheets.
Test cases are grouped in several sequences:

- Power-up
- Write Enable/Write Disable
- Initial Configuration (TBPARM)
- Block Protection/ Sector Protection
- Initial Configuration (TBPROT)
- Read Flash Memory array (Standard and Continuous)
- DDR Read Mode (Standard and Continuous)
- Read Registers
- Read OTP
- Read ID
- OTP Programming
- Page Programming
- xSPI (Octal) Mode operations
- Program Suspend
- Program Registers operations
- Sector Erase
- Erase suspend
- Software Reset
- Hardware Reset
- Bulk Erase
- Configuration Register bits for data protection
- Persistent Protection mode
- Password protection mode

Each sequence consists of series of Positive and Negative tests
Each test case has the code name complying with following rule:
XXX_YYY_ZZZ
where:
XXX – Test Sequence name.
YYY – Test series number
ZZZ – Test case (a, b, c, ...)

3 Test Sequences

3.1 Power Up

3.1.1 Series 1. Power Up Negative Test

a. POWERUP_1_a

Test case symbolic name: POWERUP_1_a
Description: Test if the device ignores commands written during power up.
What is being verified:
The device ignores all instructions until a time of t_{PU} has elapsed.
Method of verification:
At simulation time 0 fs drive signals CS# low. Initiate read instruction and read SO signal to verify that read has not been accepted on the rising edge of CS#. After power up time output SO should be tristated. Warning is generated.

3.1.2 Series 2. Power Up Positive Test

a. POWERUP_2_a

Test case symbolic name: POWERUP_2_a
Description: Test if the device is properly delivered with all bits set to 1 after power up and that Status and Configuration Registers contain proper values.
What is being verified:
After power-up SO signal should be Hi-Z and model should be ready to accept any instruction.
Method of verification:
After time of t_{PU} has elapsed drive signal CS# low and initiate read instruction to verify memory preloaded data. Initiate read status register instruction to verify status register bits. Initiate read configuration register instruction to verify configuration register bits.

3.2 WriteEnable/WriteDisable

3.2.1 Series3. PositiveTest

a. WREN_3_a

Testcasesymbolicname: WREN_3_a
Description: Testwritesto WriteEnableLatch(WEL)bit.
Whatis beingverified:
WriteEnableinstruction.
Methodof verification:
DriveCS#signallow,initiateWRENinstructionandthendriveCS#signalhigh.Initiatereadstatusregisterinstructionto verify thatWRENinstructionsetstheWELbit.

b. WRDI_3_b

Testcasesymbolicname: WRDI_3_b
Description: Testwritesto WriteEnableLatch(WEL)bit.
Whatis beingverified:
WriteDisableinstruction.
Methodof verification:
DriveCS#signallow,initiateWRDIinstructionandthendriveCS#signalhigh.Initiatereadstatusregisterinstructionto verify thatWRDIinstructionresetstheWELbit.

3.3 Initialconfiguration(TBPARM)

3.3.1 Series4. PositiveTest

a. TBPARM_4_a

Testcasesymbolicname: TBPARM_4_a
Description: TBPARMbit positivetest
Whatis beingverified:
The desired state of TBPARM must be selected during the initial configuration of the device during system manufacture;beforethefirstprogramoreraseoperationonthemainFlasharray.TBPARMmustnotbeprogrammed afterprogrammingoreraseisdoneinthemainFlasharray.
Methodof verification:
First, initiate WREN instruction to set WEL bit. Then, initiate WRR command with sixteen input cycles to change ConfigurationRegisterTBPARMbit.InitiateRDSR1instructiontoverifythatwritecycleisinprogress(WIP='1').After t_w verifythatwritecycleisdonebycheckingStatusRegisterWIPbit(WIP='0').
Afterwritecycleisover,initiateRDCRcommandtoverifythatTBPARMhascorrectvalue.

b. TBPARM_4_b

Testcasesymbolicname: TBPARM_4_b
Description: TBPARMbit negativetest
Whatis beingverified:
If TBPARMis alreadyprogrammedto 1, an attempt to changeit backto zero will fail and set the Program Error bit (P_ERRin SR1[6]).
Methodof verification:
First, initiate WRENinstructionto set WELbit. Then, initiate WRRcommandwith sixteeninput cyclesto change Configuration RegisterTBPARMbit. Initiate RDSR1 instructionto verify that operationis not accepted(WIP='0'). Verify that P_ERRbit is set. Initiate read RDCRinstructionto verify that TBPARMbit did not changeits value (TBPARM='1')

3.4 BlockProtectionTests

3.4.1 Series5. PositiveTest

a. PROT_5_a

Testcasesymbolicname: PROT_5_a
Description: Test Block Protectionfunctionalityfor Hybrid Architecture (TBPROT= 0)
Whatis beingverified:
The Block Protection (BP2-BP0) bits in the status registercan be usedto protectan addressrangeof the main Flash arrayfrom programand eraseoperations. The sizeof the range is determinedby the valueof the BP bits and the upper startingpoint of the rangeis selectedby the TBPROTbit of the configurationregister.
Methodof verification:
For eachcombinationof BP2-BP0 bits in the status registerinitiate WRENinstructionto set Status Register1 WELbit. Thenwith WRRcommandset (BP2-BP0) bits value.
Initiate again WRENinstructionto set WELbit. Then, initiate PP instructionto programto flash memory. Programlocationshouldbe in protectedsector. Verifythat programoperationfailedby readingstatusregister(WIP= '0', P_ERR='1').

b. PROT_5_b

Testcasesymbolicname: PROT_5_b
Description: Test Block Protectionfunctionality for Hybrid Architecture (TBPROT= 1)
Whatis beingverified:
The Block Protection (BP2-BP0) bits in the status registercan be usedto protectan addressrangeof the main Flash arrayfrom programand eraseoperations. The sizeof the range is determinedby the valueof the BP bits and the lower startingpoint of the rangeis selected by the TBPROTbit of the configurationregister.
Methodof verification:
For eachcombinationof BP2-BP0 bits in the status registerinitiate WRENinstructionto set WEL bit. Thenwith WRRcommandset (BP2-BP0) bits.
Initiate again WRENinstructionto set WELbit. Then, initiate PP instructionto programto flash memory. Programlocationshouldbe in protectedsector. Verifythat programoperationfailedby readingstatusregister(WIP= '0', P_ERR='1').

c. PROT_5_c

Testcasesymbolicname: PROT_5_c
Description: Description: Test Block Protection functionality for Uniform Architecture (TBPROT= 0)
What is being verified:
The Block Protection (BP2-BP0) bits in the status register can be used to protect an address range of the main Flash array from program and erase operations. The size of the range is determined by the value of the BP bits and the upper starting point of the range is selected by the TBPROT bit of the configuration register.
Method of verification:
For each combination of BP2-BP0 bits in the status register initiate WREN instruction to set Status Register 1 WEL bit. Then with WRR command set (BP2-BP0) bits value.
Initiate again WREN instruction to set WEL bit. Then, initiate PP instruction to program to flash memory. Program location should be in protected sector. Verify that program operation failed by reading status register (WIP = '0', P_ERR='1').

d. PROT_5_d

Testcasesymbolicname: PROT_5_d
Description: Test Block Protection functionality for Uniform Architecture (TBPROT= 1)
What is being verified:
The Block Protection (BP2-BP0) bits in the status register can be used to protect an address range of the main Flash array from program and erase operations. The size of the range is determined by the value of the BP bits and the lower starting point of the range is selected by the TBPROT bit of the configuration register.
Method of verification:
For each combination of BP2-BP0 bits in the status register initiate WREN instruction to set WEL bit. Then with WRR command set (BP2-BP0) bits.
Initiate again WREN instruction to set WEL bit. Then, initiate PP instruction to program to flash memory. Program location should be in protected sector. Verify that program operation failed by reading status register (WIP = '0', P_ERR='1').

3.5 Initial configuration(TBPROT)

3.5.1 Series6. TBPROT Tests

a. TBPROT_6_a

Test case symbolic name: TBPROT_5_a
Description: TBPROT bit negative test
What is being verified:
If TBPROT is programmed to 1, an attempt to change it back to zero will fail and set the Program Error bit (P_ERR in SR1[6]).
Method of verification:
First, initiate WREN instruction to set WEL bit. Then, initiate WRR command with sixteen input cycles to change Configuration Register TBPROT bit. Verify that program operation failed by reading status register (WIP = '0', P_ERR='1').
Initiate RDCR command to verify that TBPROT bit still has previous value.

3.6 Read

3.6.1 Series7. Positive Tests

a. READ_7_a

Test case symbolic name: READ_7_a
Description: Test read (4 Bytes address).
What is being verified:
The device is first selected by driving the CS# chip select input pin to the logic low state.
The instruction opcode is followed by a 4-byte address with each bit being latched in during the rising edge of the serial SCK signal. Then the memory contents at given address is shifted out on the serial output SO pin. Each bit is shifted out at the SCK frequency during the falling edge of the serial clock SCK signal. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.
Method of verification:
Initiate Read Normal mode. The specified address must be in the lower 128Mbit. Verify that correct data has been read.
Initiate Read Normal mode with clock_num aux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). The specified address must be in the lower 128Mbit. Verify that correct data has been read.
Initiate Fast Read mode. The specified address must be in the lower 128Mbit. Verify that correct data has been read.
Initiate Fast Read mode with clock_num aux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). The specified address must be in the lower 128Mbit. Verify that correct data has been read.

Repeat described reading section with Wrap Enable Mode activated.

d. READ_7_b

Test case symbolic name: READ_7_b

Description: Test read in xSPI (OCTAL) Mode.

What is being verified:

The device is first selected by driving the CS# chip select input pin to the logic low state.

The instruction opcode is followed by a 4-byte address with each bit being latched in during the rising edge of the serial SCK signal. Then the memory contents at given address is shifted out on the serial output IO0-IO7 pins. Each bit is shifted out at the SCK frequency during the falling edge of the serial clock SCK signal. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.

Method of verification:

Initiate Octal I/O High Performance Read Mode. Verify that correct data has been read.

Initiate Octal I/O High Performance Read Mode with clock_num aux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). Verify that correct data has been read.

Initiate Octal I/O DDR Read Mode. The specified address must be in the lower 128Mbit. Verify that correct data has been read.

Initiate Octal I/O DDR Read Mode with clock_num aux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). The specified address must be in the lower 128Mbit. Verify that correct data has been read.

3.7 Read Registers

3.7.1 Series 8. Positive Tests

a. READ_SR1_8_a

Test case symbolic name: READ_SR1_8_a

Description: Test read Status Register 1.

What is being verified:

The Read Status Register 1 (RDSR1) instruction opcode allows the Status Register 1 contents to be read out of the SO serial output pin. It is possible to read the Status Register 1 continuously by providing multiples of eight clock cycles. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.

Method of verification:

Initiate Read Status Register Read (RDSR1) instruction. Verify that correct data has been read.

Initiate Read Status Register Read (RDSR1) with clock_num aux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). Verify that correct data has been read.

b. READ_SR2_8_b

Test case symbolic name: READ_SR2_8_b

Description: Test read Status Register 2.

What is being verified:

The Read Status Register 2 (RDSR2) instruction opcode allows the Status Register 2 contents to be read out of the SO serial output pin. It is possible to read the Status Register 2 continuously by providing multiples of eight clock cycles. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.

Method of verification:

Initiate Read Status Register Read (RDSR2) instruction. Verify that correct data has been read.

Initiate Read Status Register Read (RDSR2) with clock_num aux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). Verify that correct data has been read.

c. READ_CR_8_c

Testcasesymbolicname READ_CR1_8_c
Description Test read Configuration Register.
What is being verified:
The Configuration Register (RDCR1) instruction opcode allows the Configuration Register contents to be read out of the SO serial output pin. It is possible to read the Configuration Register continuously by providing multiples of eight clock cycles. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.
Methodof verification:
Initiate Configuration Register Read (RDCR1) instruction. Verify that correct data has been read.
Initiate Configuration Register Read (RDCR1) with clock_numaux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). Verify that correct data has been read.

d. READ_PPBLOCK_8_d

Testcasesymbolicname READ_PPBLOCK_8_d
Description Test read PPB Lock Register.
What is being verified:
The PPB Lock Register (PLBRD) instruction opcode allows the PPB Lock Register contents to be read out of the SO serial output pin. It is possible to read the PPB Lock Register continuously by providing multiples of eight clock cycles. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.
Methodof verification:
Initiate PPB Lock Register Read (PLBRD) instruction. Verify that correct data has been read.
Initiate PPB Lock Register Read (PLBRD) with clock_numaux_parameter (bus_data_read cycle drives CS# low for number of clock cycles that isn't multiple of eight.). Verify that correct data has been read.

e. READ_ACCREG_8_e

Testcasesymbolicname READ_ACCREG_8_e
Description Test read PPB and DYB Access Registers.
What is being verified:
The PPB Access Register (PPBRD4) instruction opcode allows the PPB Access Register contents to be read out of the SO serial output pin. The device is first selected by driving the CS# chip select input pin to the logic low state. Then the instruction code is latched into the SI pin during the rising edge of the serial clock SCK signal. Then the 31-Bit address is shifted in. After address, the 8-bit PPB access register contents is shifted out on the serial output SO pin. Each bit is shifted out at the SCK frequency during the falling edge of the serial clock signal. It is possible to read the PPB Access Register continuously by providing multiples of eight clock cycles. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.
The DYB Access Register (DYBRD4) instruction opcode allows the DYB Access Register contents to be read out of the SO serial output pin. Procedure for reading this register is the same as for PPB Access Register.
Methodof verification:
Initiate PPB Access Register Read (PPBRD4) instruction. Verify that correct data has been read.
Initiate DYB Access Register Read (DYBRD4) instruction. Verify that correct data has been read.

f. READ_ECCREG_8_f

Testcasesymbolicname READ_ECCREG_8_f
Description Test read ECC Register.
What is being verified:
To read the ECC Status Register, the command is followed by the ECC unit address, the four least significant bits (LSB) of address must be set to zero. This is followed by the number of dummy cycles selected by the read latency value in CR2V[3:0]. Then the 8-bit contents of the ECC Register, for the ECC unit selected, are shifted out on SO 16 times, once for each byte in the ECC Unit. If CS# remains low the next ECC unit status is sent thru SO 16 times, once for each byte

in the ECC Unit. The maximum operating clock frequency for the ECC READ command is 166 MHz.

Method of verification:

Initiate ECC Register Read ECCRD instruction. Verify that correct data has been read.

g. READ_RDAR_8_g

Test case symbolic name: READ_RDAR_8_g

Description: Test read RDAR Register.

What is being verified:

The Read Any Register (RDAR) command provides a way to read all device registers non-volatile and volatile. The instruction is followed by a 4 Byte address, followed by a number of latency (dummy) cycles set by CR2V[3:0] or CR3V[7:6]. Then the selected register

Reading undefined locations provides undefined data.

Method of verification:

Initiate RDAR instruction for all addresses in defined address range. Verify that correct data has been read.

h. READ_RDAR_8_h

Test case symbolic name: READ_RDAR_8_h

Description: Test read RDAR Register in Octal Mode.

What is being verified:

The Read Any Register (RDAR) command provides a way to read all device registers non-volatile and volatile. The instruction is followed by a 4 Byte address, followed by a number of latency (dummy) cycles set by CR2V[3:0] or CR3V[7:6]. Then the selected register

Reading undefined locations provides undefined data.

Method of verification:

Initiate RDAR instruction for all addresses in defined address range. Verify that correct data has been read.

3.8 Read OTP

3.8.1 Series 9. Positive Tests

a. READ_OTP_9_a

Test case symbolic name: READ_OTP_9_a

Description: Test read OTP region.

What is being verified:

The Read OTP Data bytes commands reads data from the OTP region. The protocol of the Read OTP Data Bytes commands is the same as the Fast Read Data Bytes command except that it will not wrap to the starting address after the OTP address is at its maximum; instead, the data beyond the maximum OTP address will be undefined. CS# signal can be driven high after any bit of the data-out sequence is being shifted out to terminate the transaction.

Method of verification:

Initiate OTPR instruction from the OTP region 1. Verify that correct data has been read.

Initiate OTPR instruction from the OTP region 31. Verify that correct data has been read.

Initiate OTPR instruction from the address beyond the maximum OTP address. Verify that data is undefined.

3.9 ReadID

3.9.1 Series10. PositiveTests

a. READ_JID_10_a

Testcasesymbolicname: READ_JID_10_a
Description: Test ReadJEDECStandardID instruction.
Whatis beingverified:
The device is first selected by driving the CS# chip select input to the logic low state. After this, the RDID 8-bit instruction opcodeis shiftedin ontothe SI serial input. Afterthe last bit of the RDIDinstruction opcodeis shiftedinto the device, a byte of manufacturer identification, two bytes of device identification, extended device identification and CFI information will be shifted sequentially out on the SO serial output. The Read Identification (RDID) instruction sequence is terminatedby drivingthe CS# chip selectinputto the logichigh state anytimeduringdata output.
Methodof verification:
InitiateJEDECRDIDinstruction. Verifythat manufactureridentification, deviceidentification, extendeddeviceidentificationandCFI informationare correct.
InitiateJEDECRDIDwithclock_numaux_parameter(bus_data_readcycle drivesCS#lowfor numberof clockcyclesthatisn't multipleof eight.). Verifythatcorrectdatahasbeenread.

b. READ_SFDP_10_b

Testcasesymbolicname: READ_SFDP_10_b
Description: Test ReadSerial FlashDiscoverableParameters
Whatis beingverified:
Thecommandis initiatedby shiftingon SI the instructioncode "5Ah", followedby a 24-bit addressof 000000h, followed by 8 dummycycles. The SFDP bytes are then shifted out on SO starting at the falling edge of SCK after the dummy cycles. The SFDP bytes are always shifted out with the MSB first. If the 24-bit address is set to any other value, the selected location in the SFDP space is the starting point of the data read. This enables random access to any parameterin the SFDPspace.
Methodof verification:
Initiate RSFDPinstruction. Verifythatcorrectdatahasbeenread.

c. READ_SFDP_OPI_10_c

Testcasesymbolicname: READ_SFDP_OPI_10_c
Description: Test ReadSerial FlashDiscoverableParametersandJedecID in OPI
Whatis beingverified:
Enter Octal SDR mode. The commandis initiatedby shiftingthe instructioncode "5Ah", followedby a 24-bit addressof 000000h, followedby 8 dummycycles. The SFDP bytes are then shifted out starting at the falling edge of SCK after the dummycycles. The SFDP bytes are always shifted out with the MSB first. If the 24-bit address is set to any other value, the selected location in the SFDP space is the starting point of the data read. This enables random access to any parameterin the SFDPspace.
Enter Octal DDR mode. The commandis initiatedby shiftingthe instructioncode "5Ah", followedby a 24-bit addressof 000000h, followedby 8 dummycycles. The SFDP bytes are then shifted out starting at the falling edge of SCK after the dummycycles. The SFDP bytes are always shifted out with the MSB first. If the 24-bit address is set to any other value, the selected location in the SFDP space is the starting point of the data read. This enables random access to any parameterin the SFDPspace.
Methodof verification:
Initiate RSFDP, RDIDNandRDUIDinstructions.. Verifythatcorrectdatahasbeenread.

3.10 Software Reset

3.10.1 Series 11. Software Reset Tests

a. RST_11_a

Test case symbolic name: RST_11_a
Description: RSTEN+ RST command set – negative test
What is being verified:
The Reset Enable Command is required immediately before a Reset command (RST). Any command other than RST following the RSTEN command will clear the reset enable condition and prevent a later RST command from being recognized.
Method of verification:
Initiate RSTEN instruction. Then initiate WREN command to clear reset enable condition. Initiate RST command and verify that command is ignored.

b. RST_11_b

Test case symbolic name: RST_11_b
Description: RSTEN+ RST command set – positive test
What is being verified:
The RST command immediately following a RSTEN command initiates the software reset process.
Method of verification:
Initiate RSTEN instruction. Then initiate RST command to start Software Reset procedure. A reset command is executed when CS# is brought high at the end of instruction and requires tRPH time to execute. After tRPH time is elapsed verify that the non-volatile bits in the configuration register CR1NV (TBPROT_NV, TBPARM_NV) retain their previous state. Verify that the Block Protection bits in the status register SR1V are reseted to their default state. Verify that after Reset VDLR value becomes NVDLR value.

3.11 OTP Program – Positive Tests

3.11.1 Series 12. Positive Tests

a. OTPPROG_12_a

Test case symbolic name: OTPPROG_12_a
Description: Write to OTP region - Single Byte Programming.
What is being verified:
The OTP Program command programs data in the OTP region, which is in a different address space from the main array data. The protocol of the OTP Program command is the same as the Page Program command. Each region in the OTP memory space can be programmed one or more times, provided that the region is not locked. Subsequent OTP programming can be performed only on the unprogrammed bits.
Method of verification:
First, initiate WREN instruction to set WEL bit. Then, initiate OTP_PP instruction to program a single byte to OTP region. Initiate read status register instruction to verify that OTP_PP cycle is in progress. After t_{PP} verify that

programming is done by reading status register.
After programming is over verify that correct data has been programmed. Lock programmed OTP region.

b. OTPPROG_12_b

Test case symbolic name: OTPPROG_12_b
Description: Write to OTP region - OTP Page Programming.
What is being verified:
The OTP Program command programs data in the OTP region, which is in a different address space from the main array data. The protocol of the OTP Program command is the same as the Page Program command. Each region in the OTP space can be programmed one or more times, provided that the region is not locked. Subsequent OTP programming can be performed only on the unprogrammed bits.
Method of verification:
First, initiate WREN instruction to set WEL bit. Then, initiate OTP_PP instruction to program a 200 bytes to OTP region. Initiate read status register instruction to verify that OTP_PP cycle is in progress. After t_{PP} verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.

3.12 OTP Program – Negative Tests

3.12.1 Series 13. Negative Tests

a. OTPPROG_13_a

Test case symbolic name: OTPPROG_13_a
Description: Programming locked region
What is being verified:
The OTP Program command programs data in the OTP region, which is in a different address space from the main array data. The protocol of the OTP Program command is the same as the Page Program command. Each region in the OTP space can be programmed one or more times, provided that the region is not locked. Subsequent OTP programming can be performed only on the unprogrammed bits.
Method of verification:
Initiate WREN instruction to set WEL bit. Then, initiate OTP_PP instruction. Program location should be in lock region. Verify that program operation failed by reading status register (WIP = '0', P_ERR='1'). Verify that location has not been programmed.

b. OTPPROG_13_b

Testcasesymbolicname: OTPPROG_13_b
Description: Programmingoveraddresslimit
Whatis beingverified:
The OTP Program command programs data in the OTP region, which is in a different address space from the main array data. The protocol of the OTP Program command is the same as the Page Program command. Each region in the OTP space can be programmed one or more times, provided that the region is not locked. Subsequent OTP programming can be performed only on the unprogrammed bits.
Methodof verification:
Initiate WREN instruction to set WEL bit. Then, initiate OTP_PP instruction. Program location should be outside OTP regions. OTP Program operations outside the valid OTP Range is
ignored without P_ERR set to '1'.

c. OTPPROG_13_c

Testcasesymbolicname: OTPPROG_13_c
Description: programming16lowestaddressbytes
Whatis beingverified:
Programming zeros at 16 lowest address bytes will fail and set P_ERR.
Programming ones at 16 lowest address bytes is ignored without P_ERR set to "1"
Methodof verification:
Initiate WREN instruction to set WEL bit. Then, initiate OTP_PP instruction with data different that 16'hFF. Program location should be within the 16 lowest address bytes. Verify that operation failed with P_ERR= “1” and WIP = “1”. A Clear Status Register followed by Write Disable command must be sent to return device to standby state.
Initiate WREN instruction to set WEL bit. Then, initiate OTP_PP instruction with data equal to 16'hFF. Program location should be within the 16 lowest address bytes. Verify that operation is ignored and P_ERR= “0”.

3.13 Page Programming(3 bytes address)

3.13.1 Series 14. Positive Tests

a. PGPROG_14_a

Testcasesymbolicname: PGPROG_14_a
Description: Write to memory array – 3 bytes address, Page Size = 256
Whatis beingverified:
The Page Program instruction opcode allows bytes to be programmed in the memory. Before the Page Program instruction opcode can be accepted by the device internal state machine, a WREN instruction must be issued to set WEL bit. The Page Program instruction is entered by driving the CS# chip select input pin to the logic low state. The instruction opcode is followed by 3-byte address and at least one data byte on the SI serial input pin. Up to 256 data bytes can be placed into the SI serial input pin after the 3-bytes address. If the 8 least significant address bits are not all zero, all transmitted data goes beyond the end of the current page are programmed from the start address of the same page. The CS# chip select pin must be driven to the logic low state during the entire duration of the sequence. If more than 256 bytes are sent to the device, previously latched data are discarded and the last 256 data bytes are guaranteed to be programmed correctly within the same page.
Methodof verification:
First, initiate WREN instruction to set WEL bit. Then, initiate PP instruction to program single byte. Initiate read status register instruction to verify that PP cycle is in progress. After tPP verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.

Initiate WREN instruction to set WEL bit. Then, initiate PP instruction to program 3 bytes. Initiate read status register instruction to verify that PP cycle is in progress. After tPP verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.
Initiate WREN instruction to set WEL bit. Then, initiate PP instruction to program more then 256 bytes. Initiate read status register instruction to verify that PP cycle is in progress. After tPP verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.

b. PGPROG_14_b

Testcase symbolic name: PGPROG_14_b
Description: Write to memory array – 3 bytes address, Page Size = 512
What is being verified:
The Page Program instruction opcode allows bytes to be programmed in the memory. Before the Page Program instruction opcode can be accepted by the device internal state machine, a WREN instruction must be issued to set WEL bit. The Page Program instruction is entered by driving the CS# chip select input pin to the logic low state. The instruction opcode is followed by 3-byte address and at least one data byte on the SI serial input pin. Up to 512 data bytes can be placed into the SI serial input pin after the 3-bytes address. If the 8 least significant address bits are not all zero, all transmitted data goes beyond the end of the current page are programmed from the start address of the same page. The CS# chip select pin must be driven to the logic low state during the entire duration of the sequence. If more than 512 bytes are sent to the device, previously latched data are discarded and the last 512 data bytes are guaranteed to be programmed correctly within the same page.
Method of verification:
Initiate WREN instruction to set WEL bit. Then, initiate PP instruction to program more then 512 bytes. Initiate read status register instruction to verify that PP cycle is in progress. After tPP verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.

c. PGPROG_14_c

Testcase symbolic name: PGPROG_14_c
Description: Write to memory array – 4 bytes address
What is being verified:
Each region in the memory space can be programmed one or more times, provided that the sector is not locked. Subsequent programming can be performed only on the unprogrammed bits.
Method of verification:
First, initiate WREN instruction to set WEL bit. Then, initiate PP4 instruction to program single byte. Initiate read status register instruction to verify that PP4 cycle is in progress. After tPP verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.
Initiate WREN instruction to set WEL bit. Then, initiate PP4 instruction to program 4 bytes. Initiate read status register instruction to verify that PP4 cycle is in progress. After tPP verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.
Initiate WREN instruction to set WEL bit. Then, initiate PP4 instruction to program more then 512 bytes. Initiate read status register instruction to verify that PP4 cycle is in progress. After tPP verify that programming is done by reading status register.
After programming is over verify that correct data has been programmed.

3.14 Program Suspend

3.14.1 Series 15. Positive Test

a. PGSUSP_15_a

Test case symbolic name: PGSUSP_15_a
Description: Test Program suspend operation
What is being verified:
Program suspend instruction.
Method of verification:
First, initiate WREN instruction to set WEL bit. Then, initiate PP4 instruction to program to flash memory. Initiate read status register instruction to verify that PP4 cycle is in progress. Initiate ESP instruction, and verify that after tEPS command has been accepted by reading status registers.
For all read commands initiate read from the top of memory array. Verify that when the highest address is reached, the address counter will wrap around and roll back to 000000h, allowing the read sequence to be continued indefinitely.
Initiate READ4 instruction from program suspended sector. Verify that output is undefined.
Initiate command that is not allowed during Program Suspend. Verify that command is ignored.
Initiate PGRS command and verify that programing again in-progress by reading status register.
After programming is over verify that correct data has been programmed.

3.15 PPB Bit Programming

3.15.1 Series 16. PPB Bits Tests

a. PGPPB_16_a

Test case symbolic name: PGPPB_16_a
Description: Test PPB Bit program.
What is being verified:
PPB Access register write (PPBP) instruction.
Method of verification:
Initiate Hardware Reset to set value of PPB LOCK bit to '1'. After t_{RPH} initiate read PPB Lock register instruction to confirm that PPB Lock bit is set.
Initiate WREN instruction to set WEL bit. Then, initiate PPB Bit program (PPBP) command. Initiate read status register instruction to verify that program cycle is in progress. After t_{PP} verify that programming is done by reading status register. Initiate read PPB Bit instruction to verify that correct data has been programmed.

3.16 PPB Erase

3.16.1 Series 17. PPB Bits Tests

a. PPBERS_17_a

Test case symbolic name: PPBERS_17_a
Description: Test PPB Bit erase command.
What is being verified:
PPB bits can't be changed if PPB LOCK= '0';
Method of verification:

First, initiate WREN instruction to set WEL bit. Then, initiate PPB Lock register write (PLBWR) command to clear PPBLOCK. Initiate read status register instruction to verify that program cycle is in progress. After t_{PP} verify that programming is done by reading status register. Initiate read PPB Lock register instruction to verify that correct data has been programmed.
Initiate WREN instruction to set WEL bit. Then, initiate PPB Erase (PPBE) command. Initiate read status register instruction to verify that erase cycle is in progress. After t_{PPBE} verify that erasure is done by reading status register. Verify that erase operation failed by reading status register (WIP = '0', E_ERR='1').
Initiate Hardware Reset to set value of PPBLOCK bit to '1'. After t_{RPH} initiate read PPB Lock register instruction to confirm that PPB Lock bit is set.

3.17 DYB Bit Programming

3.17.1 Series 18. DYB Bit Programming

a. DYBPG_18_a

Test case symbolic name: DYBPG_18_a
Description: Test DYB Bit program
What is being verified:
DYB Access register write (DYBWR) instruction.
Method of verification:
Initiate WREN instruction to set WEL bit. Then, initiate DYB Bit program (DYBWR) command. Initiate read status register instruction to verify that program cycle is in progress. After t_{PP} verify that programming is done by reading status register. Initiate read DYB Bit instruction to verify that correct data has been programmed.

3.18 Sector Erase

3.18.1 Series 19. Positive Tests

a. SECERS_19_a

Test case symbolic name: SECERS_19_a
Description: Test sector erase instruction (4 Bytes address).
What is being verified:
Erasure of sector.
Method of verification:
Initiate WREN instruction to set WEL bit. Then, initiate SE4 instruction. Initiate read status register instruction to verify that SECTOR_ERASE cycle is in progress. After t_{SE} verify that erasing is done by reading status register.
After erasing is over, verify that selected sector was erased by reading from them. Read from any location inside erased sector should return 0xFF.

b. SECERS_19_b

Test case symbolic name: SECERS_19_b
Description: Sector Erase applied to a 256 KB range that includes 4 KB sectors
What is being verified:
Erasure of sector.
Method of verification:

Initiate WREN instruction to set WEL bit. Then, initiate SE4 instruction. Initiate read status register instruction to verify that SECTOR_ERASE cycle is in progress. After t_{SE} verify that erasing is done by reading status register.
After erasing is over, verify that selected sector was erased by reading from them. Read from any location inside erased sector should return 0xFF.

c. SECERS_19_c

Test case symbolic name: SECERS_19_c
Description: Test parameter 4kB sector erase instruction (4 Bytes address).
What is being verified:
Erasure of sector.
Method of verification:
Initiate WREN instruction to set WEL bit. Then, initiate P4E4 instruction. Initiate read status register instruction to verify that SECTOR_ERASE cycle is in progress. After t_{SE} verify that erasing is done by reading status register.
After erasing is over, verify that selected sector was erased by reading from them. Read from any location inside erased sector should return 0xFF.

3.19 Erase Suspend

3.19.1 Series 20. Positive Tests

a. ERSSUSP_20_a

Test case symbolic name: ERSSUSP_20_a
Description: Test Erase suspend operation (4 Bytes address).
What is being verified: Erase suspend instruction.
Method of verification:
First, initiate WREN instruction to set WEL bit. Then, initiate SECTOR_ERASE instruction to erase sector in flash memory. Initiate read status register instruction to verify that SECTOR_ERASE cycle is in progress. Initiate ERSP instruction, and verify that after t_{ESL} command has been accepted by reading status register.
Initiate READ4 instruction from sector not selected for programming. Verify that correct data has been read.
Initiate READ4 instruction from sector being program-suspended. Verify that output is undefined.
Initiate FAST_READ4 instruction from sector not selected for programming. Verify that correct data has been read.
Initiate FAST_READ4 instruction from sector being program-suspended. Verify that output is undefined.
Initiate READ Configuration Register instruction. Verify that correct data has been read.
Initiate READ PPB Lock Register instruction. Verify that correct data has been read.
Initiate EPSR command and verify that erasing is again in progress by reading status register.
After erasing is over verify that selected sector was erased by reading from them. Read from any location inside erased sector should return 0xFF.

3.20 Program During Erase Suspend

3.20.1 Series 21. Positive Tests

a. ERSSUSPPG_21_a

Test case symbolic name: ERSSUSPPG_21_a
Description: Test Program during Erase suspend operation.
What is being verified:
Program instruction during erase suspend.

Methodof verification:

Initiate WRENinstruction to set WELbit. Then, initiate sector erase instruction. Check WIP bit of status register to see if erase is in progress. Initiate erase suspend instruction and check if WIP bit is de-asserted. Initiate page program instruction on sector that is not suspended. Check WIP bit to see if programming has started. Initiate program suspend instruction. Initiate series of read instructions for memory array to check data. For read instructions that address the suspended sectors memory should output 'X' ('U'). Initiate write bank register instruciton and check if write was successful by reading bank register. Initiate program resume instruction and check WIP bit to see if programing is in progress. Initiate read instruction to check programmed data. Initiate erase resume program and check WIP bit to see if erase is in progress. Initiate read array instructions to check if sector is erased (all ones)

3.21 SoftwareReset

3.21.1 Series22 PositiveTests

a. RESET_22_a

Testcase symbolicname: RESET_22_a

Description: Page Programalgorithmtermination

Whatis being verified:

RESETinstruction applied during Page ProgramOperation terminate programming.

Methodof verification:

Initiate WRENinstruction to set WELbit. Initiate PP4 instruction and by reading status register verify that PP4 cycle is in progress.

During programming operation initiate RESETinstruction. Verify by reading from status register that PP cycle is terminated. Initiate Read operation from location inside selected page. Read from any location inside page should return undefined data.

b. RESET_22_b

Testcase symbolicname: RESET_22_b

Description: Sector Erase algorithmtermination

Whatis being verified:

RESETinstruction applied during Sector Erase terminate operation in progress.

Methodof verification:

Initiate WRENinstruction to set WELbit. Initiate SECTOR_ERASE_4instruction and by reading status register verify that SECTOR_ERASE_4cycle is in progress.

During erasing operation initiate RESETinstruction. Verify by reading from status register that SECTOR_ERASE_4 cycle is terminated. Initiate Read operation from location inside selected sector. Read from any location inside sector should return undefined data.

c. RESET_22_c

Testcase symbolicname: RESET_22_c

Description: ProgramSuspend algorithmtermination

Whatis being verified:

If a Reset is performed while program is suspended, the suspend operation will abort, and the contents of the page that was being programmed and subsequently suspended will be undefined.

Methodof verification:

First, initiate WRENinstruction to set WELbit. Then, initiate PP4 instruction to program to flash memory. Initiate read status register instruction to verify that PP4 cycle is in progress.

Initiate PGSPinstruction, and verify that after 10us command has been accepted by reading status register.

During program suspend initiate RESET instruction. Verify by reading from status register that PP4 cycle is terminated. Initiate Read operation from location inside selected page. Read from any location inside page should return undefined data.

3.22 Bulk Erase

3.22.1 Series 23. Negative Tests

a. BULKERS_23_a

Test case symbolic name: BULKERS_23_a
Description: Test Bulk Erase operation
What is being verified:
If at least one of sectors are software protected, Bulk Erase Operation will be rejected.
Method of verification:
Initiate WREN instruction to set WEL bit. Initiate BE instruction and by reading status register verify that BE cycle is not in progress.
Initiate READ from several different locations in memory. Verify that correct data has been read.

3.22.2 Series 23. Positive Tests

a. BULKERS_23_a

Test case symbolic name: BULKERS_23_a
Description: Test Bulk Erase operation
What is being verified:
Bulk Erase Operation.
Method of verification:
Initiate WREN instruction to set WEL bit. Initiate WRR instruction to unprotect all sectors. After t_W verify that writing is done by reading status register.
Initiate WREN instruction to set WEL bit. Initiate BE instruction and by reading status register verify that BULK_ERASE cycle is in progress. After t_{BE} verify that erasing is done by reading status register.
After erasing is over, verify that flash memory was erased by reading from them. Read from any location inside flash memory should return 0xFFFF.

3.23 DPD

3.23.1 Series 24. Positive Tests

a. DPD_24_a

Test case symbolic name: DPD_24_a
Description: *Deep power down mode positive test*
What is being verified: Deep power down mode positive test
Method of verification: *Enter Deep Power Down command and wait for 3us to enter DPD. Then initate reset enable (66h) command and Sodtware Reset (99h) to exit DPD.*

b. DPD_24_b

Testcasesymbolicname DPD_*24*_b
Description *Deeppowerdownmodepositivetest*
Whatis beingverified: Deeppowerdownmodepositivetest
Method of verification: *Write 08h to CR3NV register, then initate ENDPD_0_0 (B9h) command. Read value of CR3NV register that should be HiZ. Then initate reset enable (66h) command and Sodtware Reset (99h) to exit DPD. Read value of CR3NV register.*

c. DPD_24_c

Testcasesymbolicname DPD_*24*_c
Description *Deeppowerdownmodenegativetest*
Whatis beingverified: Deeppowerdownmodenegativetest
Method of verification: *Write 08h to CR3NV register, then initate ENDPD_0_0 (B9h) command. Generate CS# pulse less than tCSDPD. Then initate reset enable (66h) command and Sodtware Reset (99h) to exit DPD. Read value of CR3NV register.*

3.24CRCASO

3.24.1 Series25. PositiveTests

a. CRCASO_25_a

Testcasesymbolicname CRCASO_*25*_a
Description Test CRC ASO, CRC calculation and read of CRC register
Whatis beingverified:
Entering CRC ASO and starting CRC calculation over a part of memory array. Read of CRC registers to verify correct calculation.
Methodof verification:
Initiate Set CRC ASO enter instruction, set starting address for CRC calculation and set ending address for CRC calculation. Then, by reading status register verify that write cycle is in progress. After $t_{CRCSETUP}$ verify that calculation is done by reading status register. Initiate read RDAR instruction to verify that correct data has been read.

b. CRCASO_25_b

Testcasesymbolicname CRCASO_*25*_b
Description Test CRC ASO, CRC suspend and read of CRC register
Whatis beingverified:
Entering CRC ASO and starting CRC calculation over a part of memory array. Then initate CSC Suspend command.
Methodof verification:
Initiate Set CRC ASO enter instruction, set starting address for CRC calculation and set ending address for CRC calculation, then initate CRC Suspend command (B0h). When the CRC Suspend command is written during a CRC operation, the device requires a maximum of tCRCSL to suspend the program operation.

c. CRCASO_25_c

Testcasesymbolicname CRCASO_*25*_c
Description Test Interface CRC.
Whatis beingverified:

TestInterfaceCRC.

Methodof verification:

InitiateSetCRCASOenterinstruction,setstartingaddressforCRCcalculationandsetendingaddressforCRC calculation,theninitateCRCSuspendcommand(B0h).WhentheCRCSuspendcommandis writtenduringa CRCoperation,thedevice requiresa maximumoftCRCSLtosuspendtheprogramoperation.Thenresetdevice.Theninitiate InterfaceCRCcommandto verifyvalue ofcalculatedInterfaceCRCregisteraftercalculatingICRCduring RDAY2_4_0(EEh)command.

d. CRCASO_25_d

Testcasesymbolicname: CRCASO_*25*_d
Description: TestInterfaceCRC.
Whatis beingverified:
TestInterfaceCRC.
Methodof verification:
Theninitiate InterfaceCRCcommandto verifyvalue ofcalculatedInterfaceCRCregisteraftercalculatingICRCduring PRPGE_4_1(12h)command.

e. CRCASO_25_e

Testcasesymbolicname: CRCASO_*25*_e
Description: TestInterfaceCRC.
Whatis beingverified:
TestInterfaceCRC.
Methodof verification:
Restartdevice.Theninitiate InterfaceCRCcommandto verifyvalue ofcalculatedInterfaceCRCregisteraftercalculating ICRCduringRDSR1_4_0 command.Afterreading ICRVregistarvaluerestartdevice.

3.26PersistentProtectionMode

3.26.1Series26. PositiveTests

a. PSTPROT_26_a

Testcasesymbolicname: PSTPROT_*26*_a
Description: PersistentProtectionModeTest
Whatis beingverified:
Selectionof Protectionmode.VerificationofPPBLockbit (ThePersistentProtectionmethodsets thePPBLockBitto'1'duringHardwareReset,sothatPPBbitsareunprotectedbya devicereset)
Methodof verification:
InitiateWRENinstructiontosetWELbit.Then,initiateASPPinstructiontosetconditionsfor PersistentProtectionmode(ASPR[2:1]='10')andbyreadingstatusregisterverifythat programcycleisinprogress. Aftert_{PP}verifythatprogrammingisdonebyreadingstatusregister. InitiatereadASPinstructiontoverifythatcorrectdatahasbeenprogrammed.
DriveRST#signallow.AfterRST#goeshighdeviceisreadytoreaddata.
Initiatereadof sPPBLOCKregisterandverifythatPPBLOCKBitissetto'1'.

3.27 Password Protection Mode

3.27.1 Series 27. Positive Tests

a. PSWPROT_27_a

Test case symbolic name: PSWPROT_*27*_a
Description: Password Protection Mode Test
What is being verified:
If the Password Mode is chosen, the password must be programmed prior to setting the Protection Mode Lock Bits.
Method of verification:
Initiate WREN instruction to set WEL bit. Then, initiate Password register write (PASSP) command. Initiate read status register instruction to verify that program cycle is in progress.
After t_{PP} verify that programming is done by reading status register.
Initiate read Password register instruction to verify that correct password has been programmed.

b. PSWPROT_27_b

Test case symbolic name: PSWPROT_*27*_b
Description: Password Protection Mode Test
What is being verified:
Selection of Password mode. Verification of PPB Lock bit (The Password Protection method clears the PPB Lock Bit to '0' during Hardware Reset, so that PPB bits are protected by a device reset)
Method of verification:
Initiate WREN instruction to set WEL bit. Then, initiate ASP P instruction to set conditions for Password Protection mode (ASPR[2:1] = '01') and by reading status register verify that program cycle is in progress. After t_{PP} verify that programming is done by reading status register.
Initiate read ASP instruction to verify that correct data has been programmed.
Drive RST# signal low. After RST# goes high device is ready to read data. Initiate read of PPB LOCK register and verify that PPB LOCK Bit is cleared to '0'.

c. PSWPROT_27_c

Test case symbolic name: PSWPROT_*27*_c
Description: Password Protection Mode Test
What is being verified:
Password Unlock command sequence
Method of verification:
Initiate read PPB Lock register command to verify that PPB Lock bit is cleared (PPB bits are in protected state).
Initiate WREN instruction to set WEL bit. Then, initiate Password unlock (PASSU) command. Initiate read status register instruction to verify that password unlock cycle is in progress.
After t_{pp} verify that password unlock cycle is done by reading status register. Initiate read of PPB Lock register and verify that PPB Lock Bit is set to '1'.
Drive RST# signal low. After RST# goes high device is ready to read data. Initiate read of PPB Lock register and verify that PPB Lock Bit is cleared to '0'.

3.28 Series28. *PasswordProgramSPI*

3.28.1 Series28. Positive Tests

a. PSWPG_28_a

Testcasesymbolicname: PSWPG*_28_*a
Description: *Passwordprogram:programandverify*
Whatis beingverified:
Passwordprogram:programandverify
Methodof verification:
After Write Enable command, initate WRARG_4_0 command and write 01h into CFR5V registar. Then Program password with WRARG_4_0 command. After that, read Password to verify that Password is programed. Then enter Password Protection mode and reset device.

b. PSWPG_28_b

Testcasesymbolicname: PSWPG*_28_*b
Description: *PasswordUnlockcommandsequence*
Whatis beingverified:

Methodof verification:
Confirm PPB lock bit is cleared, initiate password unlock sequence. Read Status Register 2 and Read Configuration Register. Confirm PPB lock bit is set. Initiate PLBWR command to clear PPB Lock bit. Confirm PPB Lock bit is cleared.

3.29 Series29. *PasswordProgramOPI*

3.29.1 Series29. Positive Tests

a. PSWPG_OPI_29_a

Testcasesymbolicname: PSWPG*_OPI_29_*a
Description: *Passwordprogram:programandverify*
Whatis beingverified:
Passwordprogram:programandverify in Octal mode
Methodof verification:
After Write Enable command, initate WRARG_4_0 command and write 01h into CFR5V registar. Enter OCTAL DDR Mode by writing 03h into CFR5V registar via WRARG_4_0 command. Then Program password with WRARG_4_0 command. After that, read Password to verify that Password is programed. Then enter Password Protection mode and reset device.

b. PSWPG_OPI_29_b

Testcasesymbolicname: PSWPG*_OPI_29_*b
Description: *EnterPasswordProtectionmode-*
Whatis beingverified:

Enter Password Protection mode- PPB Lock is cleared after reset

Method of verification:

Initate WRARG_4_0 command and enter Password Protection mode. After restarting device enter in OPI mode after Write Enable command, initate WRARG_4_0 command and write 01h into CFR5V registar. After Write Enable command, initate WRARG_4_0 command and write 03h into CFR5V registar to enter OCTAL DDR mode and verify that PPB bit is cleared.

c. PSWPG_OPI_29_c

Test case symbolic name: PSWPG_OPI_29_c

Description: *Password Unlock command sequence.*

What is being verified:

Password Unlock command sequence.

Method of verification:

Confirm PPB lock bit is cleared, then initiate password unlock sequence. Initate WRARG_4_0 command and set PPB lock bit. Confirm that PPB lock bit is set. Initiate WRPLB_0_0 command to clear PPB Lock bit. Confirm PPB Lock bit is cleared.

3.30.2 Series 30. Negative Tests

a. PSWPROT_30_a

Test case symbolic name: PSWPROT_30_a

Description: Password Protection Mode Test

What is being verified:

Password read and program can't be done if Password protection mode is set.

Method of verification:

Initiate WREN instruction to set WEL bit. Then, initiate Password register write (PASSP) command. Initiate read status register instruction to verify that program cycle is not in progress. Initiate read Password register instruction and verify that command is ignored.

b. PSWPROT_30_b

Test case symbolic name: PSWPROT_30_b

Description: Password Protection Mode Test

What is being verified:

Password unlock: wrong password entered

Method of verification:

Initiate read PPB Lock register command to verify that PPB Lock bit is cleared (PPB bits are in protected state).

Initiate WREN instruction to set WEL bit. Then, initiate Password unlock (PASSU) command with incorrect password. Initiate read status register instruction to verify that password unlock cycle is in progress.

After t_{PP} verify that password unlock cycle is done by reading status register. Verify by reading status register that programming error occurred.

Initiate read of PPB Lock register and verify that PPB Lock Bit is still cleared to '0'.

3.31 Sector Erase Counter

3.31.1 Series 31. Positive Tests

a. SECERASE_CNT_31_a

Testcasesymbolicname: SECERASE_CNT_*30*_a
Description: *ReadSectorEraseCountpositivetests*
Whatis beingverified:
ReadSectorEraseCountpositivetests
Methodof verification:
Initiate SectorErasecommandandverifySectorerasecounterbyreadingSECVregistarwithRDARG_4_0command.

3.32 JedecReset

3.32.1 Series31. PositiveTests

a. JEDEC_RST_32_a

Testcasesymbolicname: JEDEC_RST_*30*_a
Description: *JEDECreset,positivetest*
Whatis beingverified:
JEDECreset,positivetest
Methodof verification:
Initiate jedecreset.