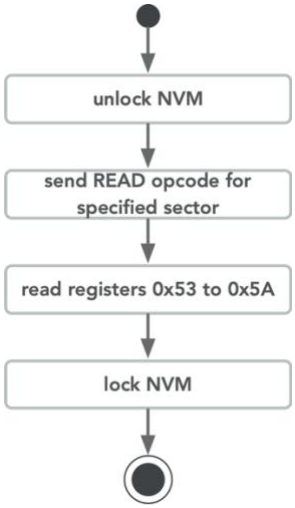
STUSB4500 Register Description

1 NVM Interface

The Non-Volatile Memory has an interface that is programmed via dedicated commands. The commands are configured in the NVM_CTRL register (NVM_CTRL_LOW@0x96 and NVM_CTRL_HIGH@0x97). The REQ bit makes the internal interface parse and execute the command. After the command has been executed it resets the REQ bit. Therefore it is important to first setup the high register of NVM_CTRL (0x97) and then write to the low register of NVM_CTRL (0x96).
The internal interface can be shut off and reset by dedicated bits in the NVM_CTRL register. While interfacing the NVM these bits have to be set to 1.

1.1 NVM Read Operation

Reading the NVM is done by sending the READ opcode along with the specified sector. Afterwards the sector data can be read in the registers 0x53 to 0x5A.



Example command for reading out sector 2:

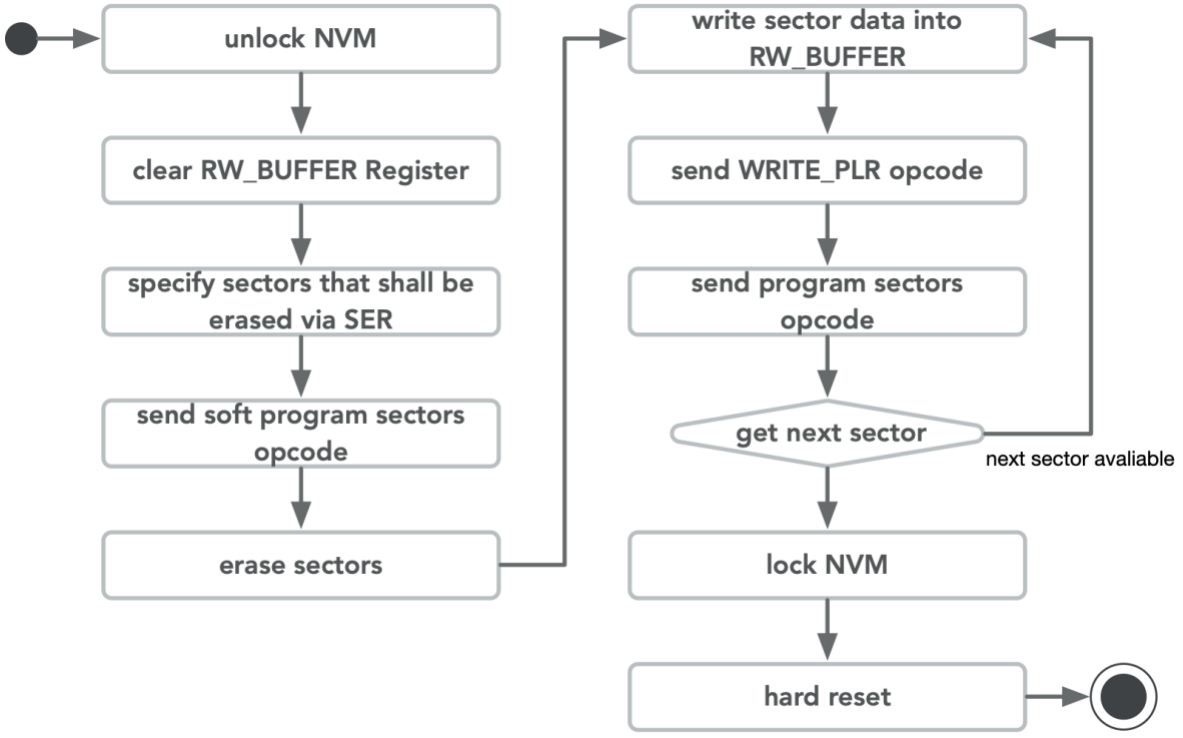
Configure the high byte of NVM_CTRL to the READ opcode:
NVM_CTRL[15:8] = READ

Configure the low byte of NVM_CTRL with powering on the internals and un-reset it as well as sector 2 which shall be read:
NVM_CTRL[15:8] = (1 << PWR) | (1 << RST_N) | (1 << REQ) | 2

Wait until REQ bit is cleared and then read out registers 0x53 to 0x5A

1.2 NVM Write Operation

Before writing to a sector the sector has to be erased in advance.



1.3 Command Description

All commands require NVM_CTRL[7:6] to be 0b11.

Command	Dependency
READ	NVM_CTRL[2:0] – SECTOR specifying the respective sector (only 1 sector per command)
WRITE_PLR	
WRITE_SER	NVM_CTRL[15:11] – SER_MASK each bit specifying 1 sector
READ_PLR	
READ_SER	
ERASE_SECTOR	-
PROG_SECTOR	NVM_CTRL[2:0] – SECTOR specifying the respective sector that will be programmed
SOFT_PROG_SECTOR	NVM_CTRL[15:11] – SER_MASK indicating which sectors will be available for reprogramming

1.4 Register Summary

Offset	Name	Bit Pos.								
0x53	NVM_RW_BUFFER	7:0	RW_BUFFER[0]							
0x54		7:0	RW_BUFFER[1]							
0x55		7:0	RW_BUFFER[2]							
0x56		7:0	RW_BUFFER[3]							
0x57		7:0	RW_BUFFER[4]							
0x58		7:0	RW_BUFFER[5]							
0x59		7:0	RW_BUFFER[6]							
0x5A		7:0	RW_BUFFER[7]							
0x95	NVM_PASSWD		PASSWORD							
0x96	NVM_CTRL	7:0	PWR	RST_N		REQ		SECTOR		
0x97		15:8	SER_MASK					OPCODE		
0xC0	NVM Sector 0:0	7:0								
0xC1	NVM Sector 0:1	7:0								
0xC2	NVM Sector 0:2	7:0								
0xC3	NVM Sector 0:3	7:0								
0xC4	NVM Sector 0:4	7:0								
0xC5	NVM Sector 0:5	7:0								
0xC6	NVM Sector 0:6	7:0								
0xC7	NVM Sector 0:7	7:0								
0xC8	NVM Sector 1:0	7:0			GPIO_CFG					
0xC9	NVM Sector 1:1	7:0			VBUS_DISCH_DISABLE					
0xCA	NVM Sector 1:2	7:0								
0xCB	NVM Sector 1:3	7:0								
0xCC	NVM Sector 1:4	7:0								
0xCD	NVM Sector 1:5	7:0								
0xCE	NVM Sector 1:6	7:0								
0xCF	NVM Sector 1:7	7:0								
0xD0	NVM Sector 2:0	7:0								
0xD1	NVM Sector 2:1	7:0								
0xD2	NVM sector 2:2	7:0								
0xD3	NVM Sector 2:3	7:0								
0xD4	NVM Sector 2:4	7:0								
0xD5	NVM Sector 2:5	7:0								
0xD6	NVM Sector 2:6	7:0								
0xD7	NVM Sector 2:7	7:0								
0xD8	NVM Sector 3:0	7:0								
0xD9	NVM Sector 3:1	7:0								
0xDA	NVM Sector 3:2	7:0	I_SNK_PDO1				SNK_UNCONS_POWER	SNK_PDO_NUMB		USB_COMM_CAPABLE
0xDB	NVM Sector 3:3	7:0	SHIFT_VBUS_HL1							
0xDC	NVM Sector 3:4	7:0	SHIFT_VBUS_LL2				I_SNK_PDO2			
0xDD	NVM Sector 3:5	7:0	I_SNK_PDO3				SHIFT_VBUS_HL2			
0xDE	NVM Sector 3:6	7:0	SHIFT_VBUS_HL3				SHIFT_VBUS_LL3			
0xDF	NVM Sector 3:7	7:0								
0xE0	NVM Sector 4:0	7:0	V_SNK_PDO2[1:0]							
0xE1	NVM Sector 4:1	7:0	V_SNK_PDO2 [9:2]							
0xE2	NVM sector 4:2	7:0	V_SNK_PDO23[7:0]							
0xE3	NVM Sector 4:3	7:0	I_SNK_PDO_FLEX[5:0]						V_SNK_PDO23[9:8]	
0xE4	NVM Sector 4:4	7:0		POWER_OK_CFG			I_SNK_PDO_FLEX[9:6]			
0xE5	NVM Sector 4:5	7:0								
0xE6	NVM Sector 4:6	7:0				REQ_SRC_CURRENT	POWER_ONLY_ABOVE_5V			
0xE7	NVM Sector 4:7	7:0		Alarm Interrupt Mask						

1.5 NVM Control Register

Name:	CTRL
Offset:	0x96
Reset:	0x0040
Property:	

15	14	13	12	11	10	9	8
SER_MASK					OPCODE		
RW					RW		
7	6	5	4	3	2	1	0
PWR	RST_N		REQ		SECTOR		
RW	RW		RW		RW		

Bits 15:11 – SER_MASK[7:3]: Sector Erase Register Mask
For the OPCODEs WRITE_SER and ERASE_SECTOR the SER_MASK bits indicate which sectors are to be erased. There are five sectors each containing eight bytes. Each bit of SER_MASK specifies one sector.
First write the sector mask SER_MASK into SER via WRITE_SER opcode and then erase them via ERASE_SECTOR.

Bits 10:8 – OPCODE: Command to be executed

OPCODE	Value	Description
READ	0	Reads out one internal sector (0xC0 – 0xE7) specified by SECTOR and provides the values in RW_BUFFER (0x53-0x5A)
WRITE_PLR	1	Write Program Load Register (internally loads a program that enables the write to the NVM)
WRITE_SER	2	Write Sector Erase Register specified by SER_MASK
READ_PLR	3	Read Program Load Register
READ_SER	4	Read Sector Erase Register
ERASE_SECTOR	5	Erase sectors specified by SER_MASK
PROG_SECTOR	6	Program sector specified by SECTOR (requires an erase in advance)
SOFT_PROG_SECTOR	7	Soft program sectors specified by SER_MASK, so after erasing the sector it can be reprogrammed

BIT 7 – PWR
Powers on internal NVM circuitry to handle commands (unverified, best guess)

Bit 6 – RST_N
Enables internal NVM circuitry (unverified, best guess)
Active low.

Bit 4: REQ – Request command
Setting this bit triggers the chip to execute the OPCODE, so make sure to configure CTRL[15:8] firstly. When the chip is done executing the bit will be cleared.

Bits 2:0 SECTOR
Used by the READ and PROG_SECTOR opcodes to read the dedicated sector. Valid values are 0 to 4 meaning Sectors 0 to 4 respective.