How to read or to write STUSB4531 NVM content

Introduction

The STUSB4531 embeds a non-volatile memory (NVM) to store key parameters used by the autorun algorithm. The NVM has four sectors of 8 bytes each.

The mapping between bits and sector bytes is performed using the STSW-STUSB020 graphical user interface.

Typically, NVM parameters are programmed during the end-product manufacturing process.

1 STUSB4531 prerequisite

1.1 Generic I²C functions

Two generic functions are defined in this document to indicate I²C read or write actions.

For both functions, the device address is omitted to avoid overloading the document. The device address can be 0x28 or 0x29, as specified in the datasheet.

I2C_Read(register_address, number_of_byte):

- Register_address: indicates address starting point to be read
- Number_of_byte: indicates if single or multiple consecutive read operations will be performed

I2C_Write(register_address, data, number_of_byte):

- Register_address: indicates address starting point to be written
- Data: is the content to be written
- Number_of_byte: indicates if single or multiple consecutive write operations will be performed

1.2 Register setup

The registers used in this document are:

- NVM_CUST_CTRL address 0x0041
- NVM_STATUS address 0x001F
- DPM_SRC_PDO1 address 0x00A0
- DPM_SRC_PDO2 address 0x00A1

DPM_SRC_PDO1 and DPM_SRC_PDO2 are used as a single 8-byte NVM buffer, starting at the DPM_SRC_PDO1 address, for read or write operations.

As per the register map, NVM_CUST_CTRL and NVM_STATUS descriptions are:

Figure 1. NVM_CUST_CTRL address 0x0041

Address: 0x0041

Reset: 0x00

[7] **NVM_MODE:**
- 0x0: (MAIN_MODE) chip in main mode.
- 0x1: (NVM_MODE) Chip in NVM mode.

[5:4] **NVM_CUST_SECT:**
- 0x0: (CUST_SECT_0) customer sector 0.
- 0x1: (CUST_SECT_1) customer sector 1.
- 0x2: (CUST_SECT_2) customer sector 2.
- 0x3: (CUST_SECT_3) customer sector 3.

[2:0] **NVM_CUST_OPCODE:**
- 0x0: (NOP) No Operation.
- 0x1: (READ_CUST) Read.
- 0x2: (WRITE_CUST) Write.
- 0x3: (WRITE_FUNC_REG) copy functional registers to NVM.
- 0x4: (WRITE_2_FUNC_REG) copy NVM to the functional registers.

Figure 2. **NVM_STATUS address 0x001F**

Address: 0x001F

Reset: 0x00

[7] **NVM_LOADED:**
- 0x0: (NO_NVM_LOAD) load not yet done.
- 0x1: (NVM_LOAD_DONE) load completed.

[6] **NVM_END_OP:**
- 0x0: (NVM_OP_ONGOING) the operation is ongoing.
- 0x1: (NVM_OP_ENDED) the operation is finished.

[5:0] **NVM_WRITE_STATUS:** Verify result after last write operation.
Each bit represents a sector (bit0: sector0, bit1: sector1...)

1.3 Check STUSB4531 NVM status

In order to be able to read or write NVM content, the STUSB4531 NVM needs to be in a stable state.

- I2C_Read (NVM_STATUS, 1)
- Expected content:
  - NVM_LOADED = NVM_LOAD_DONE
  - NVM_END_OP = NVM_OP_ENDED
  - i.e. : b11xxxxxx

2 Read NVM bank

Step 1:

The NVM needs to be set in a special mode of operation in order to access a dedicated sector n:

- NVM_CUST_CTRL data:
  - NVM_MODE = NVM_MODE
  - NVM_CUST_OP_CODE = NOP
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

Step 2:

Sector n is enabled in read mode:

- NVM_CUST_CTRL data:
  - NVM_MODE = NVM_MODE
  - NVM_CUST_OP_CODE = READ_CUST
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

Step 3:

Wait for the end of the NVM sector n read process:

- I2C_Read(NVM_STATUS, 1)
- Expected content:
  - NVM_END_OP= NVM_OP_ENDED
  - i.e.: bx1xxxxxx

Step 4:

Read sector n content located in DPM_SRC_PDO1 and DPM_SRC_PDO2:

- I2C_Read(DPM_SRC_PDO1, 8)
- The data read will be the 8 bytes of the sector n content

Step 5:

Disable access to sector n:

- NVM_CUST_CTRL data:
  - NVM_MODE = NVM_MODE
  - NVM_CUST_OP_CODE = NOP
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

Step 6:

Switch the NVM back to normal operation:

- NVM_CUST_CTRL data:
  - NVM_MODE = MAIN_MODE
  - NVM_CUST_OP_CODE = NOP
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

In order to read multiple sectors, repeat operations from step 2 to step 5 before executing step 6.

3 Write NVM bank

Step 1:

The NVM needs to be set in a special mode of operation in order to access a dedicated sector n:

- NVM_CUST_CTRL data:
  - NVM_MODE = NVM_MODE
  - NVM_CUST_OP_CODE = NOP
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

Step 2:

Write the 8 bytes of sector n content in DPM_SRC_PDO1:

- I2C_Write(DPM_SRC_PDO1, data, 8)

Step 3:

Sector n is enabled in write mode:

- NVM_CUST_CTRL data:
  - NVM_MODE = NVM_MODE
  - NVM_CUST_OP_CODE = WRITE_CUST
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

Step 4:

Wait for the end of the NVM sector n write process:

- I2C_Read(NVM_STATUS, 1)
- Expected content:
  - NVM_END_OP= NVM_OP_ENDED
  - NVM_WRITE_STATUS[5:0] = bit n set
    - i.e.: Sector 2 to be written: expect bit 2 to be set

Step 5:

Disable access to sector n:

- NVM_CUST_CTRL data:
  - NVM_MODE = NVM_MODE
  - NVM_CUST_OP_CODE = NOP
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

Step 6:

Switch the NVM back to normal operation:

- NVM_CUST_CTRL data:
  - NVM_MODE = MAIN_MODE
  - NVM_CUST_OP_CODE = NOP
  - NVM_CUST_SECT = CUST_SECT_n
- I2C_Write(NVM_CUST_CTRL, data, 1)

In order to write to multiple sectors, repeat operations from step 2 to step 5 before executing step 6.

4 References

STUSB4531 Datasheet: STUSB4531
STUSB4531 Register map:
STUSB4531 GUI: STSW-STUSB020: STSW-STUSB020

Revision history

Table 1. **Document revision history**

Date	Version	Changes
12-Nov-2025	1	Initial release.

Contents

List of tables

List of figures