AN5440

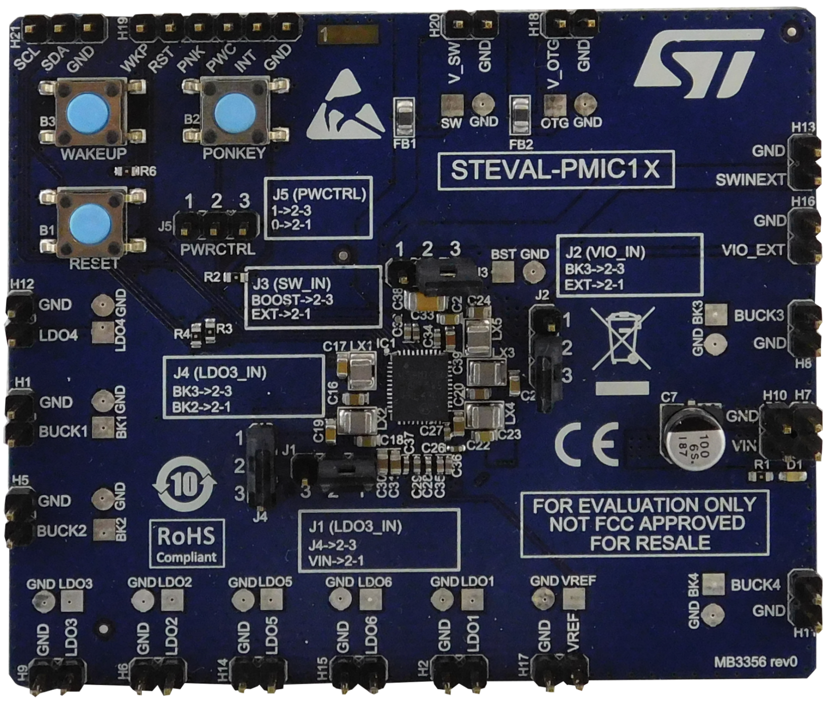
Application note

The STPMIC1 I²C programming guide

Introduction

This application note provides a detailed description of the programming function of the STPMIC1. It also describes the minimal system requirements and instructions to program the NVM for a selected power-up configuration. All examples assume that the customer uses any silicon review and has a base- knowledge of BUS I^{2}C programming.

Figure 1. STEVAL-PMIC1x



1 General considerations

This section provides some detailed information about the STPMIC1 (re)program stand-alone function. It outlines the system requirements and the instructions needed to (re)program the NVM content for a selected power-up configuration.

The STPMIC1 has a non-volatile memory (NVM) that stores essential settings, which are applied to each startup of the device. Programmability of the NVM allows the STPMIC1 to be fit to different use cases. User accessible NVM content is:

- Default output voltages
- Power-up/down sequence
- Protection behavior
- I^{2}C slave device address
- Auto turn-on and other various features (see the STPMIC1 datasheet for further details)

Three standard NVM configurations are available from the manufacturing company: the STPMIC1A, STPMIC1B and STPMIC1C. The STPMIC1C is turned ON after applying the input voltage, but all output voltages are not turned ON with Rank0 by default. To facilitate mass production, each of them can be re-programmed by I^{2}C directly into the target application. The procedure is being described below.

To access the I^{2}C interface of the STPMIC1, a simple standard I^{2}C, a USB adapter connected to the PC and a terminal emulation software (Hyper Terminal, PuTTY, RealTerm, etc…) are enough.

The communication is still present even if the device is in stand-by mode or OFF mode and VIO voltage is provided.

In this document, some examples of this procedure are described, using our USBSTICK I^{2}C to a USB adapter and the dedicated STPMIC1 GUI, directly addressing the registers and changing their content (using the general tab or a dedicated tab of the GUI program).

Figure 2. **USBSTICK – I^{2}C to USB interface**

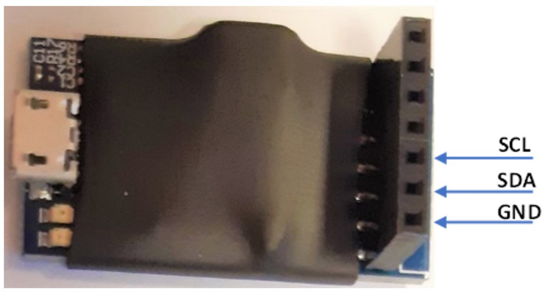
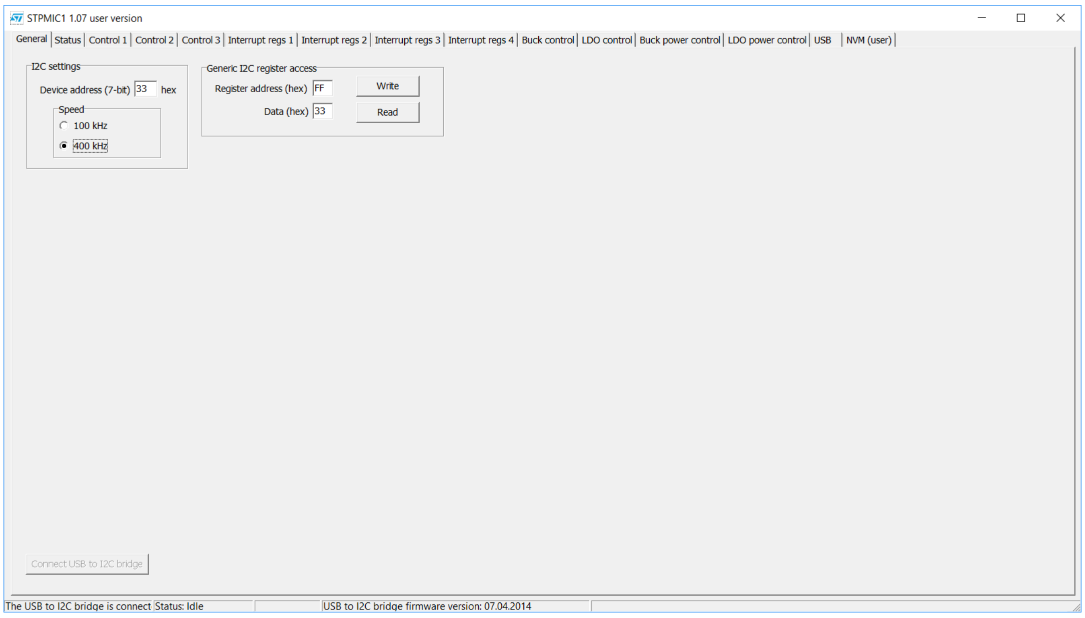


Figure 3. ST GUI interface


STPMIC1 1.07 user version
General | Status | Control 1 | Control 2 | Control 3 | Interrupt regs 1 | Interrupt regs 2 | Interrupt regs 3 | Interrupt regs 4 | Buck control | LDO control | Buck power control | LDO power control | USB | NVM (user)
Write
Read
Connect USB to I2C bridge

I2C settings
Device address (7-bit) 33 hex
Speed
○ 100 kHz
● 400 kHz

Generic I2C register access
Register address (hex) FF
Data (hex) 33

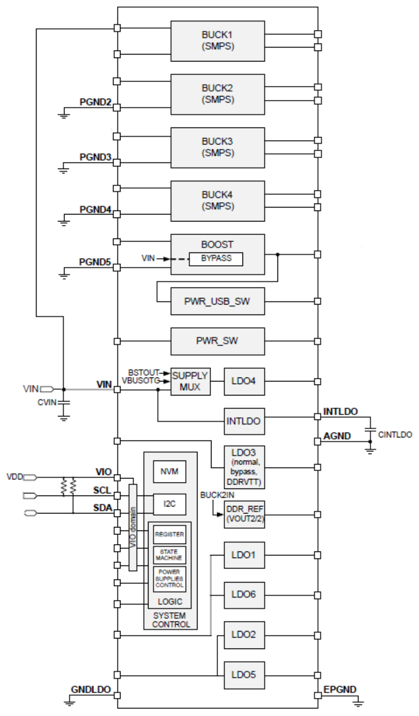
The USB to I2C bridge is connect | Status: Idle | USB to I2C bridge firmware version: 07.04.2014

1.1 Hardware considerations

The minimum system requirements to program the NVM content are:

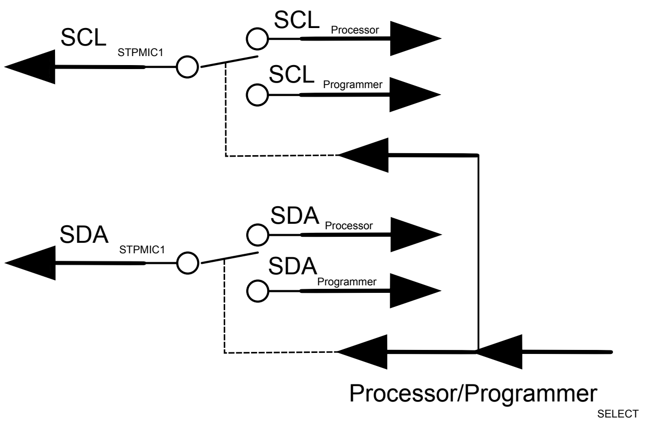
- An I^{2}C communication interface to communicate with the STPMIC1
- A power supply providing 3.8 V for VIN and 3.3 V for VIO pins
- All GND pins connected to ground

Figure 4. Minimum system requirements



In the final customer application, VIN, VIO and I^{2}C BUS communication must be isolated by using solder shorts or 0 Ω resistors to switch between external programmer or the onboard micro-processor. However, minor rework of the board is required once the NVM content is (re)programmed. Better, but more expensive if compared with the solder shorts or 0 Ω resistor, is the use of a switch to isolate the I^{2}C BUS.

Figure 5. **I^{2}C BUS**



In the STPMIC1x demo board, the VIN, I^{2}C BUS and VIO_EXT connectors are available, but you need to set the J2 to use the header H16 to supply 3.3 VIO externally.

1.2 Programming steps

The following steps are valid for the all system requirements already described:

- Turn ON the power supply at 3.8 VIN 100 mA and 3.3 V 100 mA for VIO
- Generic I^{2}C to USB converter (VCP) or the USBSTICK using ST GUI
- Check the I^{2}C communication by read the default device address at 0xFF = 0x33 (I2C_ADDR[6:0]: ‘0110011’)
- Change the NVM shadow register values according to the customer application needs, see the STPMIC1 datasheet for more details
- Transfer the new NVM shadow register values to the NVM content by starting the “NVM write operation” using the address 0xB9 and write the value 0x01 (NVM_CMD[1:0] = ‘01’)
- Done, new configuration is available in the next power-up cycle

1.3 Example 1: how to change I^{2}C device address

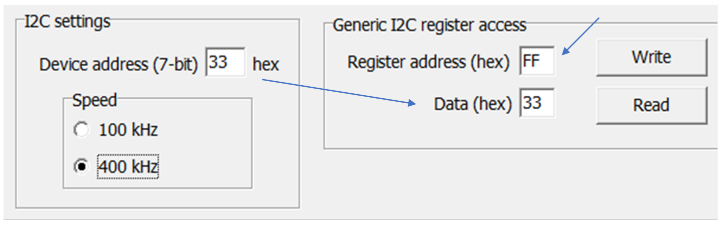
The default device address, of the STPMIC1, is 0x33. If the application needs a different value, it can be changed as described below.

It is very important to consider that the device address is changed instantly after writing procedure and a change of the settings of terminal/software is needed to properly communicate with the STPMIC1 (this programming behavior is valid for the I^{2}C device address only).

1.3.1 GUI use

For this example, the "General" of the GUI is used.

Figure 6. General tab


I2C settings
Device address (7-bit) 33 hex
Speed
100 kHz [✓ 400 kHz]
Generic I2C register access
Register address (hex) FF
Data (hex) 33

The figure above shows on the left side I^{2}C settings, 0x33 (7bit) device address and the communication speed. By clicking the "Read" button with the register 0xFF selected (you can omit 0x) and put FF on register address window, its content is shown on "Data" field, 33 (0x33) the default device address. A new value: 34 (0x34) is now entered in the "Data" field, then click "Write" button. The new address is applied immediately, so to further communicate by the GUI, the I^{2}C settings need to be updated with the new address value 34 otherwise an I^{2}C communication error is sent. See figures below.

Figure 7. I^{2}C settings

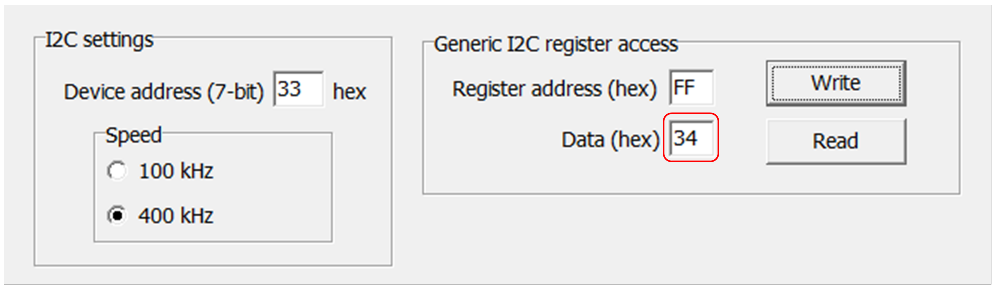
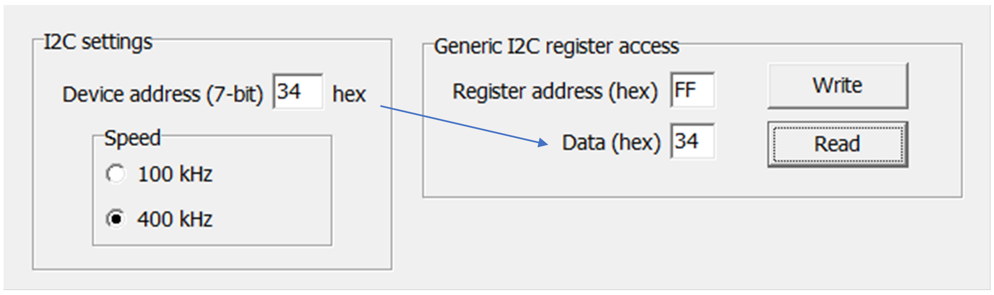

I2C settings
Device address (7-bit) 33 hex
Speed
100 kHz [✓ 400 kHz]
Generic I2C register access
Register address (hex) FF
Data (hex) 34

Figure 8. I^{2}C settings with new address value 34


I2C settings
Device address (7-bit) 34 hex
Speed
100 kHz [✓ 400 kHz]
Generic I2C register access
Register address (hex) FF
Data (hex) 34

1.3.2 Command line use

For a standard terminal emulator (like PuTTY, RealTerm, etc…) the following read/write command syntax has to be used after setting the hardware interface (Com port, Baud speed, Parity and Data Bits):

i2cget -y 0 <DEVICE ADDRESS> <ADDRESS> (read)

-y	skip the prompt for confirmation from the i2cget command
0	the I^2C bus to use, denoted as bus 0
<DEVICE ADDRESS>	the address of the slave device, [0x33]
<REGISTER>	the register on the slave to read from, [0xFF]

i2cset -y 0 <DEVICE ADDRESS> <ADDRESS> <VALUE> (write)

-y	skip the prompt for confirmation from the i2cget command
0	the I^2C bus to use, denoted as bus 0
<DEVICE ADDRESS>	the address of the slave device, [0x33]
<REGISTER>	the register on the slave to write to, [0xFF]
<VALUE>	the value to write, [0x34]

Below, step by step, an example on how to change the device address from 0x33 to 0x34

- i2cget -y 0 0x33 0xFF to 0x33 (read address value)
- i2cset -y 0 0x33 0xFF 0x34 (write new address value)
- i2cget -y 0 0x34 0xFF to 0x34 (read new address value)
- i2cset -y 0 0x34 0xB9 0x01 (NVM write operation)

Please note that changes, made on the NVM shadow registers of the device, make permanent any change, an "NVM write operation" needs to be performed by accessing the NVM control register. If this operation is not performed, previously programmed, I^2C address is loaded by NVM during next power-up sequence.

The NVM control register is located at address 0xB9 in which, the NVM write operation is performed by writing the data 0x01.

1.4 Example 2: how to change NVM content from the STPMIC1A to the STPMIC1B

The STPMIC1A version can be (re)programmed to the STPMIC1B version; the main differences between the two version are:

- LDO2 from 1.8 V to 2.9 V
- LDO2 from RANK0 to RANK2
- BUCK3 from 3.3 V to 1.8 V
- VINOK_Rise from 3.5 V to 3.3 V

The addresses to make the changes are:

- **LDO2_VOUT** 0xFD bit [3:2]
- **LDO2_RANK** 0xFA bit [3:2]
- **BUCK3_VOUT** 0xFC bit [5:4]
- **VINOK_THRES** 0xF8 bit [5:4]

1.4.1 GUI use

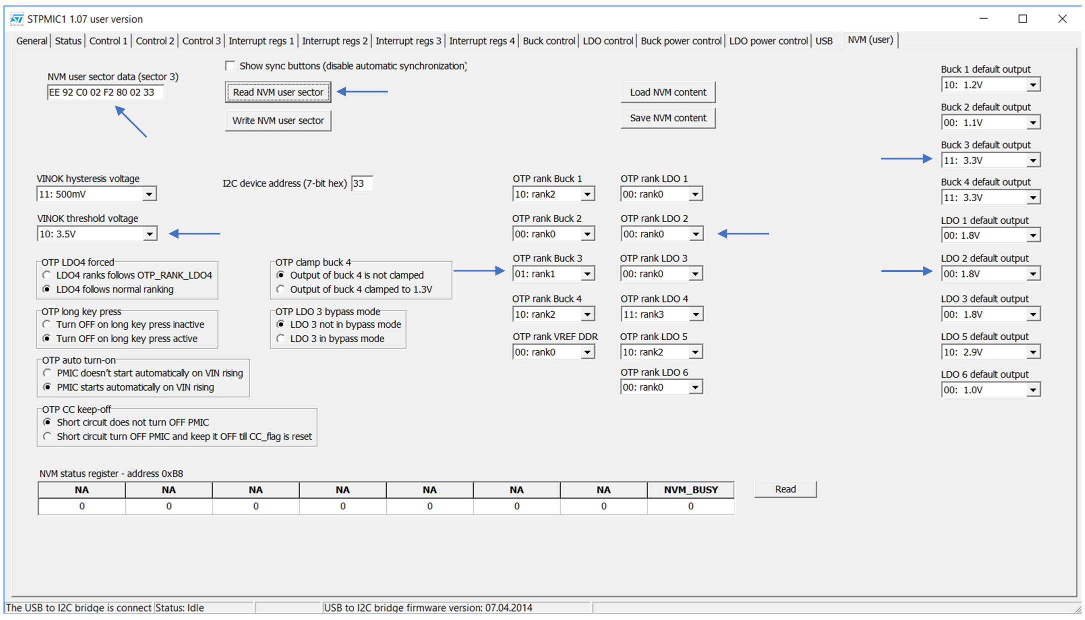
The GUI software has dedicated tabs to read/write by clicking a button, below you can see the page for NVM content (as a configurator). By clicking “Read NVM user sector” you can read the STPMIC1 version, the window “NVM user sector data” shows the NVM content and for version **A** the result is:

EE 92 C0 02 F2 80 02 33

The 4 bits highlighted must be changed to obtain the **B** version starting from the **A** version

DE 92 C8 02 D2 88 02 33

Figure 9. NVM user version A


STPMIC1 1.07 user version
General | Status | Control 1 | Control 2 | Control 3 | Interrupt regs 1 | Interrupt regs 2 | Interrupt regs 3 | Interrupt regs 4 | Buck control | LDO control | Buck power control | LDO power control | USB | NVM (user)
Read NVM user sector
Write NVM user sector
Load NVM content
Save NVM content
Read

[ ] Show sync buttons (disable automatic synchronization)

NVM user sector data (sector 3): EE 92 C0 02 F2 80 02 33

VINOK hysteresis voltage: 11: 500mV

VINOK threshold voltage: 10: 3.5V

I2C device address (7-bit hex): 33

OTP LDO4 forced: LDO4 ranks follows OTP_RANK_LDO4 [✓ LDO4 follows normal ranking]

OTP long key press: Turn OFF on long key press inactive [✓ Turn OFF on long key press active]

OTP auto turn-on: PMIC doesn’t start automatically on VIN rising [✓ PMIC starts automatically on VIN rising]

OTP CC keep-off: [✓ Short circuit does not turn OFF PMIC] Short circuit turn OFF PMIC and keep it OFF till CC_flag is reset

OTP clamp buck 4: [✓ Output of buck 4 is not clamped] Output of buck 4 clamped to 1.3V

OTP LDO 3 bypass mode: [✓ LDO 3 not in bypass mode] LDO 3 in bypass mode

OTP rank Buck 1	10: rank2
OTP rank Buck 2	00: rank0
OTP rank Buck 3	01: rank1
OTP rank Buck 4	10: rank2
OTP rank VREF DDR	00: rank0
OTP rank LDO 1	00: rank0
OTP rank LDO 2	00: rank0
OTP rank LDO 3	00: rank0
OTP rank LDO 4	11: rank3
OTP rank LDO 5	10: rank2
OTP rank LDO 6	00: rank0
Buck 1 default output	10: 1.2V
Buck 2 default output	00: 1.1V
Buck 3 default output	11: 3.3V
Buck 4 default output	11: 3.3V
LDO 1 default output	00: 1.8V
LDO 2 default output	00: 1.8V
LDO 3 default output	00: 1.8V
LDO 5 default output	10: 2.9V
LDO 6 default output	00: 1.0V

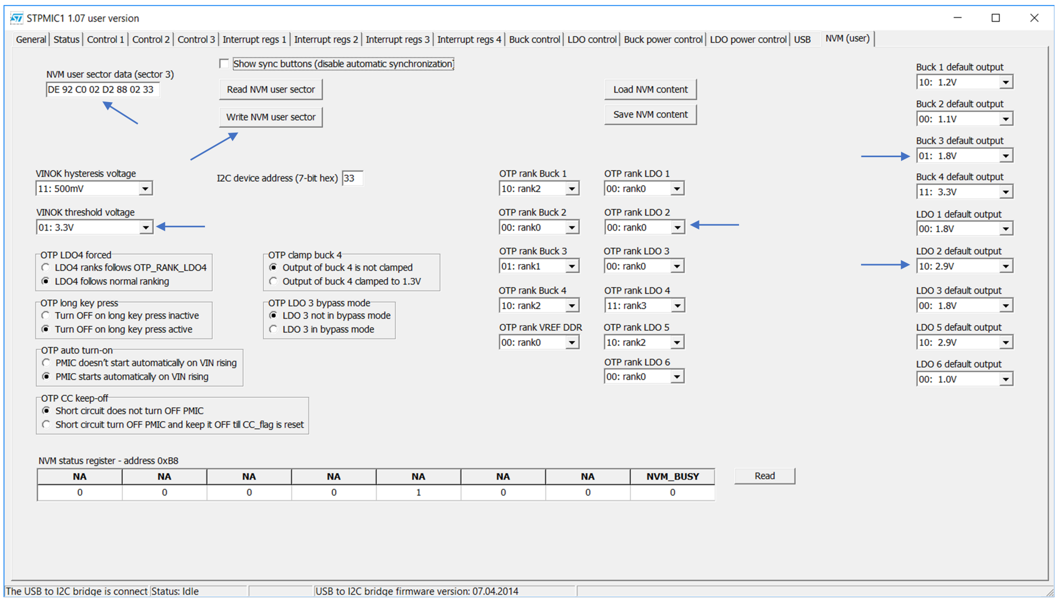
NVM status register - address 0xB8

NA	NA	NA	NA	NA	NA	NA	NVM_BUSY
0	0	0	0	0	0	0	0

The USB to I2C bridge is connect | Status: Idle | USB to I2C bridge firmware version: 07.04.2014

The figure below shows the right settings to convert the STPMIC1 to version **B**. The “NVM write operation” starts automatically by clicking the “Write NVM user sector” button and all contents are transferred from shadow registers to the NVM contents, the next power ON cycle makes effective the changes.

Figure 10. NVM user version B


STPMIC1 1.07 user version
General | Status | Control 1 | Control 2 | Control 3 | Interrupt regs 1 | Interrupt regs 2 | Interrupt regs 3 | Interrupt regs 4 | Buck control | LDO control | Buck power control | LDO power control | USB | NVM (user)
Show sync buttons (disable automatic synchronization)
Read NVM user sector
Write NVM user sector
Load NVM content
Save NVM content
Read

NVM user sector data (sector 3): DE 92 C0 02 D2 88 02 33

I2C device address (7-bit hex): 33

VINOK hysteresis voltage: 11: 500mV

VINOK threshold voltage: 01: 3.3V

OTP LDO4 forced
- ◯ LDO4 ranks follows OTP_RANK_LDO4
- ● LDO4 follows normal ranking

OTP clamp buck 4
- ● Output of buck 4 is not clamped
- ◯ Output of buck 4 clamped to 1.3V

OTP long key press
- ◯ Turn OFF on long key press inactive
- ● Turn OFF on long key press active

OTP LDO 3 bypass mode
- ● LDO 3 not in bypass mode
- ◯ LDO 3 in bypass mode

OTP auto turn-on
- ◯ PMIC doesn’t start automatically on VIN rising
- ● PMIC starts automatically on VIN rising

OTP CC keep-off
- ● Short circuit does not turn OFF PMIC
- ◯ Short circuit turn OFF PMIC and keep it OFF till CC_flag is reset

OTP rank Buck 1: 10: rank2
OTP rank Buck 2: 00: rank0
OTP rank Buck 3: 01: rank1
OTP rank Buck 4: 10: rank2
OTP rank VREF DDR: 00: rank0
OTP rank LDO 1: 00: rank0
OTP rank LDO 2: 00: rank0
OTP rank LDO 3: 00: rank0
OTP rank LDO 4: 11: rank3
OTP rank LDO 5: 10: rank2
OTP rank LDO 6: 00: rank0

Buck 1 default output: 10: 1.2V
Buck 2 default output: 00: 1.1V
Buck 3 default output: 01: 1.8V
Buck 4 default output: 11: 3.3V
LDO 1 default output: 00: 1.8V
LDO 2 default output: 10: 2.9V
LDO 3 default output: 00: 1.8V
LDO 5 default output: 10: 2.9V
LDO 6 default output: 00: 1.0V

NVM status register - address 0xB8

NA	NA	NA	NA	NA	NA	NA	NVM_BUSY
0	0	0	0	1	0	0	0

The USB to I2C bridge is connect | Status: Idle | USB to I2C bridge firmware version: 07.04.2014

1.4.2 Command line use

First, read the default values for **A** version using the commands below:

```
- i2cget -y 0 0x33 0xF8-> 0xEE
- i2cget -y 0 0x33 0xFA -> 0xC0
- i2cget -y 0 0x33 0xFC -> 0xF2
- i2cget -y 0 0x33 0xFD -> 0x80
```

Then, write changes to convert **A to B** version using the commands below:

```
- i2cset -y 0 0x33 0xF8 0xDE
- i2cset -y 0 0x33 0xFA 0xC8
- i2cset -y 0 0x33 0xFC 0xD2
- i2cset -y 0 0x33 0xFD 0x88
- i2cset -y 0 0x33 0xB9 0x01 (NVM write operation)
- Done, new configuration is available the next power-up cycle
```

1.5 Example 3: how to change NVM content from the STPMIC1C to STPMIC1A

The STPMIC1C version can be (re)programmed to the STPMIC1A version; the main differences between the two version are:

- All LDOs output voltages
- All LDOs Rank#
- All buck converters output voltages
- All buck converters Rank#
- All optional settings by default

1.5.1 GUI use

The GUI software has dedicated tabs to read/write by clicking a button. Below the page for NVM content (as a configurator) is visible.

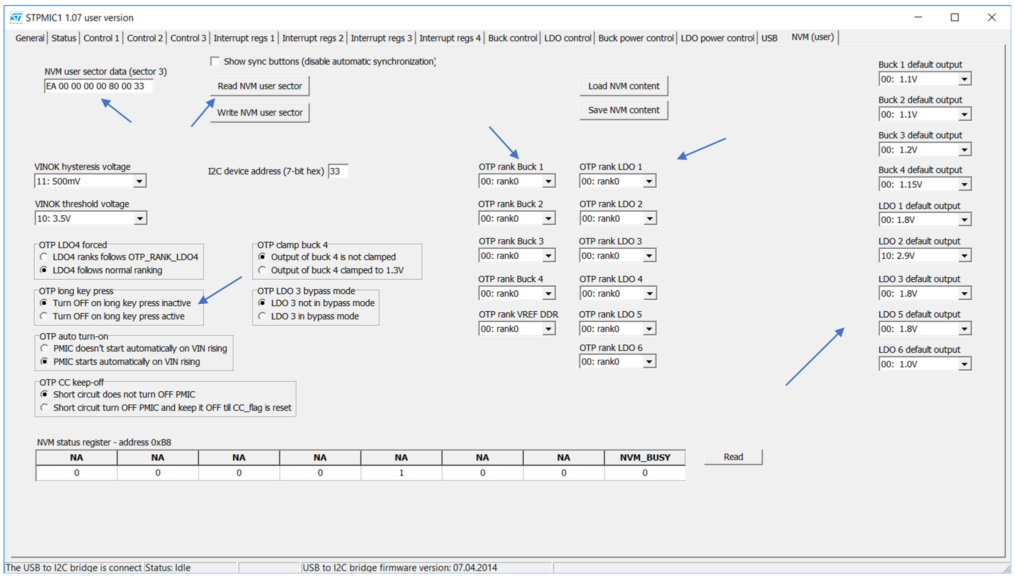
By clicking “Read NVM user sector”, the STPMIC1 version can be read and the window “NVM user sector data” shows the NVM content; concerning version **C**, the result is:

EA 00 00 00 00 80 00 33

The values below must be written to convert the STPMIC1 version from **C** to **A**

EE 92 C0 02 F2 80 02 33

Figure 11. **NVM user version C**



☐ Show sync buttons (disable automatic synchronization)

NVM user sector data (sector 3): EA 00 00 00 00 80 00 33

I2C device address (7-bit hex): 33

VINOK hysteresis voltage: 11: 500mV

VINOK threshold voltage: 10: 3.5V

OTP LDO4 forced: LDO4 ranks follows OTP_RANK_LDO4 [✓ LDO4 follows normal ranking]

OTP clamp buck 4: [✓ Output of buck 4 is not clamped] Output of buck 4 clamped to 1.3V

OTP long key press: [✓ Turn OFF on long key press inactive] Turn OFF on long key press active

OTP LDO 3 bypass mode: [✓ LDO 3 not in bypass mode] LDO 3 in bypass mode

OTP auto turn-on: PMIC doesn't start automatically on VIN rising [✓ PMIC starts automatically on VIN rising]

OTP CC keep-off: [✓ Short circuit does not turn OFF PMIC] Short circuit turn OFF PMIC and keep it OFF till CC_flag is reset

OTP rank Buck 1	00: rank0
OTP rank Buck 2	00: rank0
OTP rank Buck 3	00: rank0
OTP rank Buck 4	00: rank0
OTP rank VREF DDR	00: rank0
OTP rank LDO 1	00: rank0
OTP rank LDO 2	00: rank0
OTP rank LDO 3	00: rank0
OTP rank LDO 4	00: rank0
OTP rank LDO 5	00: rank0
OTP rank LDO 6	00: rank0
Buck 1 default output	00: 1.1V
Buck 2 default output	00: 1.1V
Buck 3 default output	00: 1.2V
Buck 4 default output	00: 1.15V
LDO 1 default output	00: 1.8V
LDO 2 default output	10: 2.9V
LDO 3 default output	00: 1.8V
LDO 5 default output	00: 1.8V
LDO 6 default output	00: 1.0V

NVM status register - address 0xB8

NA	NA	NA	NA	NA	NA	NA	NVM_BUSY
0	0	0	0	1	0	0	0

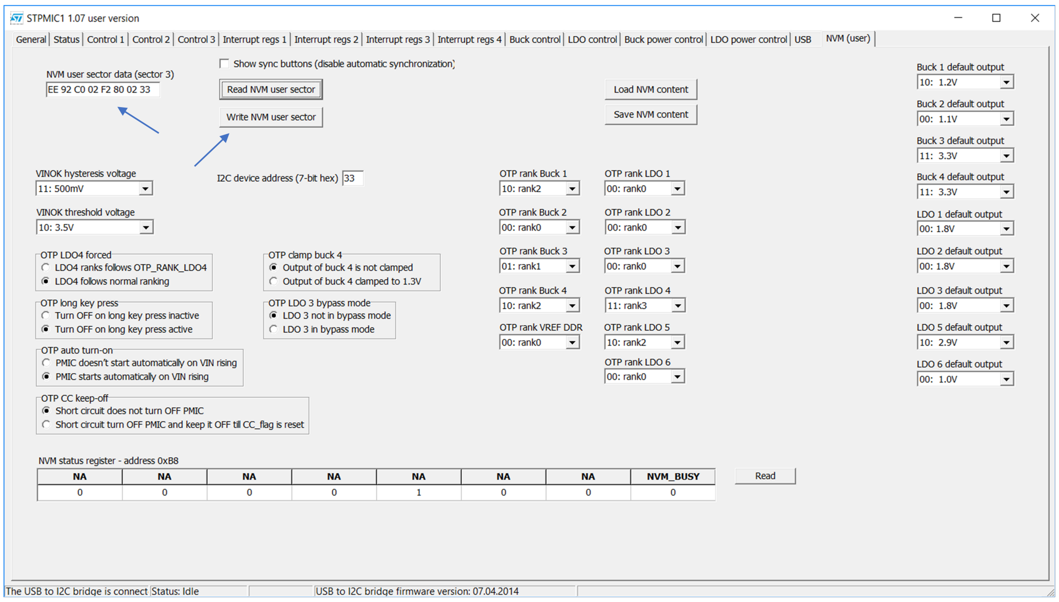
The USB to I2C bridge is connect Status: Idle USB to I2C bridge firmware version: 07.04.2014

The following steps should be performed:

- Set all output voltages the application needs
- Set all Rank numbers for all the output voltages
- Set long press key function

The figure below shows the right settings to program the STPMIC1 version **A**. The “NVM write operation” starts automatically by clicking the “Write NVM user sector” button and all contents are transferred from shadow registers to the NVM contents, the next power ON cycle makes effective the changes.

Figure 12. **NVM user version A**


STPMIC1 1.07 user version
General | Status | Control 1 | Control 2 | Control 3 | Interrupt regs 1 | Interrupt regs 2 | Interrupt regs 3 | Interrupt regs 4 | Buck control | LDO control | Buck power control | LDO power control | USB | NVM (user)
Read NVM user sector
Write NVM user sector
Load NVM content
Save NVM content
Read

Show sync buttons (disable automatic synchronization)

NVM user sector data (sector 3): EE 92 C0 02 F2 80 02 33

I2C device address (7-bit hex): 33

VINOK hysteresis voltage: 11: 500mV

VINOK threshold voltage: 10: 3.5V

OTP LDO4 forced
- ○ LDO4 ranks follows OTP_RANK_LDO4
- ● LDO4 follows normal ranking

OTP clamp buck 4
- ● Output of buck 4 is not clamped
- ○ Output of buck 4 clamped to 1.3V

OTP long key press
- ○ Turn OFF on long key press inactive
- ● Turn OFF on long key press active

OTP LDO 3 bypass mode
- ● LDO 3 not in bypass mode
- ○ LDO 3 in bypass mode

OTP auto turn-on
- ○ PMIC doesn’t start automatically on VIN rising
- ● PMIC starts automatically on VIN rising

OTP CC keep-off
- ● Short circuit does not turn OFF PMIC
- ○ Short circuit turn OFF PMIC and keep it OFF till CC_flag is reset

OTP rank Buck 1: 10: rank2
OTP rank Buck 2: 00: rank0
OTP rank Buck 3: 01: rank1
OTP rank Buck 4: 10: rank2
OTP rank VREF DDR: 00: rank0

OTP rank LDO 1: 00: rank0
OTP rank LDO 2: 00: rank0
OTP rank LDO 3: 00: rank0
OTP rank LDO 4: 11: rank3
OTP rank LDO 5: 10: rank2
OTP rank LDO 6: 00: rank0

Buck 1 default output: 10: 1.2V
Buck 2 default output: 00: 1.1V
Buck 3 default output: 11: 3.3V
Buck 4 default output: 11: 3.3V
LDO 1 default output: 00: 1.8V
LDO 2 default output: 00: 1.8V
LDO 3 default output: 00: 1.8V
LDO 5 default output: 10: 2.9V
LDO 6 default output: 00: 1.0V

NVM status register - address 0xB8

NA	NA	NA	NA	NA	NA	NA	NVM_BUSY
0	0	0	0	1	0	0	0

The USB to I2C bridge is connect Status: Idle USB to I2C bridge firmware version: 07.04.2014

1.5.2 Command line use

First step, read the default values for **C** version using the commands below:

```
- i2cget -y 0 0x33 0xF8 -> 0xEA
- i2cget -y 0 0x33 0xF9 -> 0x00
- i2cget -y 0 0x33 0xFA -> 0x00
- i2cget -y 0 0x33 0xFB -> 0x00
- i2cget -y 0 0x33 0xFC -> 0x00
- i2cget -y 0 0x33 0xFE -> 0x00
```

Then, write changes to convert **C to A** version using the commands and values below:

```
- i2cset -y 0 0x33 0xF8 0xEE
- i2cset -y 0 0x33 0xF9 0x92
- i2cset -y 0 0x33 0xFA 0xC0
- i2cset -y 0 0x33 0xFB 0x02
- i2cset -y 0 0x33 0xFC 0xF2
- i2cset -y 0 0x33 0xFE 0x02
- i2cset -y 0 0x33 0xB9 0x01 (NVM write operation)
- Done, new configuration is available next power-up cycle
```

Revision history

Table 1. **Document revision history**

Date	Version	Changes
04-Feb-2020	1	Initial release.

Contents

List of tables

List of figures