

SpiWriter User Guide

V1.00.001

Publication Release Date: May. 2018

The information in this document is subject to change without notice.

The Nuvoton Technology Corp. shall not be liable for technical or editorial errors or omissions contained herein; nor for incidental or consequential damages resulting from the furnishing, performance, or use of this material.

This documentation may not, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form without prior consent, in writing, from the Nuvoton Technology Corp.

Nuvoton Technology Corp. All rights reserved.

Table of Contents

1. Introduction	4
1.1. SpiWriter Introduction	4
2. Operation	5
2.1. SD Card 0	5
2.2. INI File	7
Raw Data Mode File Name	8
SpiLoader File Name	9
Logo File Name.....	9
Execute File Name	9
Data Verify.....	10
Chip Erase	10
4-Byte Address Mode Enable	10
Sound Play	11
Sound Play Volume.....	Error! Bookmark not defined.
Flow Status Sound File Name	12
GPIO Flow Status	12
User Image File name	13
2.3. TurboWriter.ini	13
2.4. Operation	14
2.5. Sample.....	15
3. Revision History	16

1. Introduction

1.1. SpiWriter Introduction

N9H20 series have two boot flows – one is Normal mode; the other is Recovery mode.

For N9H20, the boot flows are as below:

Normal mode boot flow is SD 0 boot -> NAND boot -> SPI boot -> SD 1 boot -> SD 2 boot -> USB boot

Recovery mode boot flow is USB boot

SpiWriter utilizes the character of Normal mode to load code of **SpiWriter.bin** from SD card 0. When SpiWriter.bin program executes, it will read the SpiWriter.ini file from SD card 0 then program spi flash according the INI file setting. This document will guide you how to prepare the SD card 0 and modify INI file.

2. Operation

2.1. SD Card 0

The SD card 0 must reserve some space to store the SDLoader and SpiWriter before usage. The procedure is as below step (N9H20 for example):

1. Launch TurboWriter in recovery mode and set the system Reserved Area Size if this SD card does not do it before
2. Burn the SDLoader binary file – N9H20K5_SpiLoader_GPM1006D0.bin as system image
3. Burn the SpiWriter binary file – N9H20_SpiWriter_QVGA.bin as execute image with “Image execute address” 0

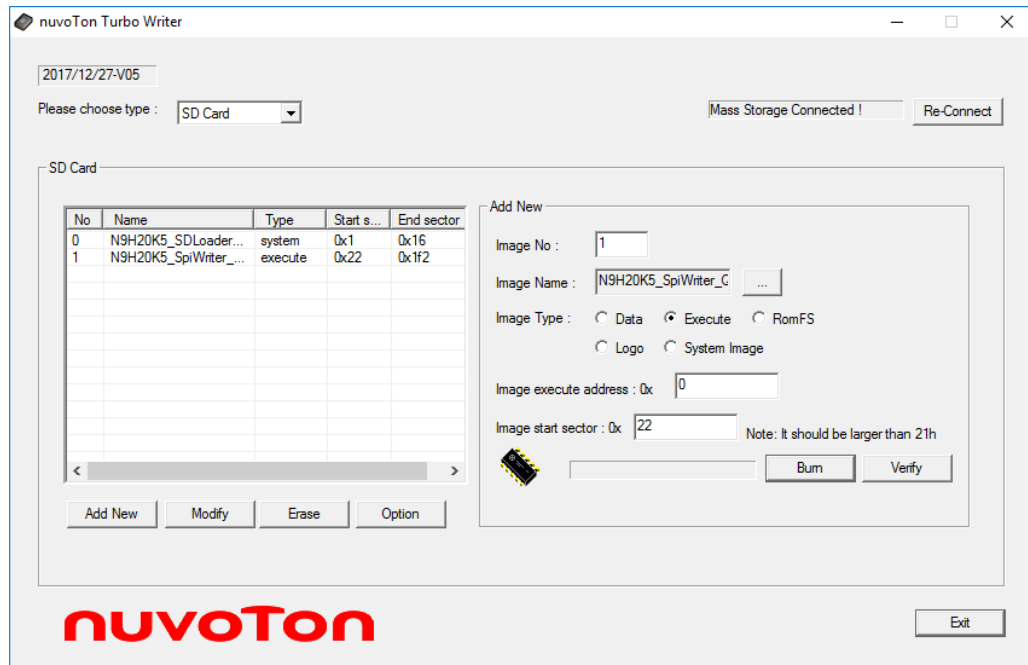


Figure 1 Burn SpiWriter

4. Reserve System area 4MB and Enable SD format.

These two files are burned in System reserved area and unable to read from card reader.

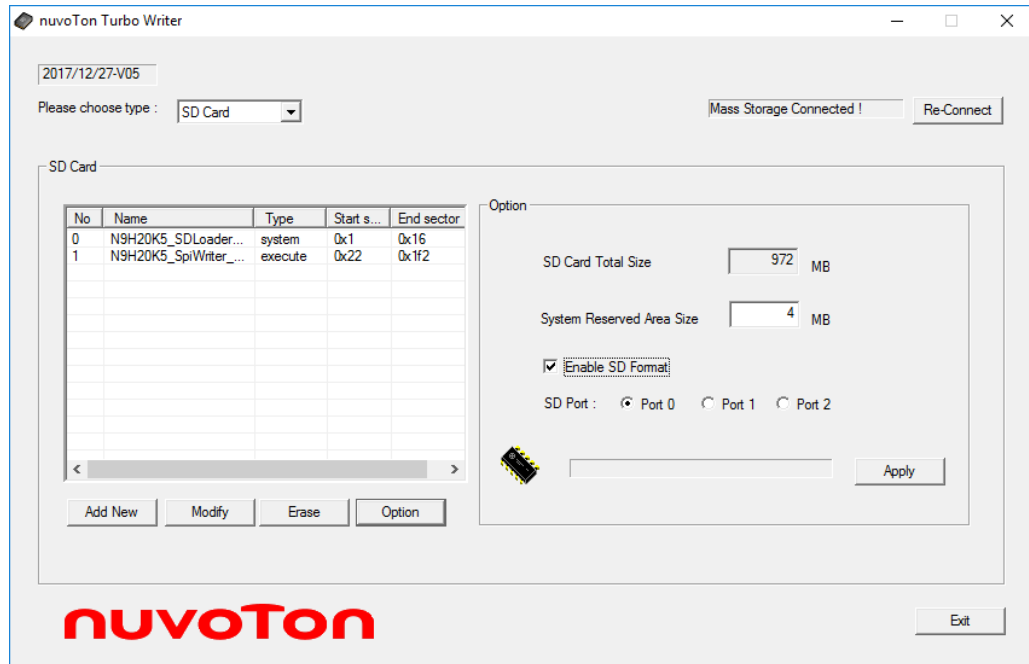


Figure 2 Set System Reserved Area Size and Format SD Card

Put this SD card to another card reader and copy SpiWriter.ini and related files that want to burn to spi flash to this SD card.

This SD card content structure is as below figure. The root directory contains the SpiWriter.ini (must), SpiLoader binary file/Raw data Image file (must), Logo binary file (option), execute binary file (option). It also provides some option in SpiWriter.ini for user. Please check the INI File section.

SD Root

— Spi Writer.ini	(Must)
— SpiLoader / Raw Data Image	(Must)
— Logo	(Option)
— Execute	(Option)
— PCM Data for Start	(Option)
— PCM Data for Pass	(Option)
— PCM Data for Fail	(Option)
— User Image 0	(Option)
— User Image 1	(Option)
— User Image 17	(Option)

2.2. INI File

The INI file means **SpiWriter.ini** file that provides the user a flexible way to do a restricted modification without modifying the source code of SpiWriter.bin.

The SpiWriter.ini file provides some sections as below (spi flash is for N9H20):

```
[Raw Data Mode File Name]
//Format:File name
//If SpiWriter gets the file name, it ignores other files and only write the file.
//RawDataImage.bin

[Spiloader File Name]
//Format:File name, execute address
N9H20K5_Spiloader_GPM1006D0, 0x900000

[Logo File Name]
// Format:File name, Start Block (0 is decided by writer), execute address
Logo.dat, 0x0, 0x500000

[Execute File Name]
// Format:File name, Start Block (0 is decided by writer), execute address
conprog.bin, 0x9, 0x00000000

[Data Verify]
// 1 to verify data, 0 to not to verify.
1

[Chip Erase]
// 1 to erase whole spi flash, 0 to erase the sector which will be used.
0

[Sound Play]
// 1 to enable sound play, 0 to disable sound play.
0

[Sound Play Volume]
// Maximum is 63
32

[Write Root]
//Index is from 1 to 4, 0 to disable Write Root Key
0

[Flow Status Sound File Name - Start]
//StartBurn.pcm

[Flow Status Sound File Name - Pass]
//BurnComplete.pcm

[Flow Status Sound File Name - Fail]
//BurnFail.pcm

// Format:GPIO Pin, Active Level, Example:GPA6, 1
[GPIO Flow Status - Start]

[GPIO Flow Status - Pass]
```

```
[GPIO Flow Status - Fail]

// Format for User Image:File name, Start Block (0 is decided by writer)
// Ex.Image0.bin, 0x46 (Start block is 70)
// Maximum User Image Files is 18.
[User Image File name]
Image0.bin, 0x46

[User Image File name]
Image1.bin, 0x4B

[User Image File name]
Image2.bin, 0x4D

[User Image File name]
Image3.bin, 0x52
```

Due to its limited parsing ability of SpiWriter.bin, there are some constraints in SpiWriter.ini as below:

- No space is allowed to precede the option for each line.
- Only “//” comment is allowed at the beginning of each line
- String in “[]” is not allowed to be changed.
- Only “[SpiLoader File Name]” is necessary. The other sections are optional.

If the logo file is not necessary for the SpiWriter, below two methods are all to skip burning Logo.dat into the target SD card.

```
[Logo File Name]
//Logo.dat, 0x0, 0x500000
```

or

```
[Logo File Name]
```

It also allows changing the file name for burning. Below sample changes the file name from N9H20K5_SpiLoader_GPM1006D0.bin to Nuvoton.bin for “[SpiLoader File Name]” section.

```
[SpiLoader File Name]
Nuvoton.bin, 0x900000
```

Raw Data Mode File Name

The format for this setting is:

```
[SpiLoader File Name]
File name
```

User can modify the file name by this setting.

Ex:


```
[Raw Data Mode File Name]
//Format:File name
//If SpiWriter gets the file name, it ignores other files and only write the file.
//RawDataImage.bin
```

If SpiWriter gets the file name, it will only write this file into Spi flash from block 0.

SpiLoader File Name

The format for this setting is:

```
[SpiLoader File Name]
File name, execute address
```

User can modify loader file name and execute address by this setting.

Ex: For N9H20, loader execute address is 0x900000

```
[SpiLoader File Name]
//Format:File name, execute address
N9H20K5_SpiLoader_GPM1006D0.bin, 0x900000
```

Logo File Name

The format for this setting is:

```
[Logo File Name]
File name, start block, execute address
```

User can modify logo file name, spi flash start block, and execute address by this setting. If user want to burn logo binary file right after the previous image, user needs to set the start block to 0.

Ex: For N9H20, logo execute address is 0x500000

```
[Logo File Name]
// Format:File name, Start Block (0 is decided by writer), execute address
Logo.dat, 0x0, 0x500000
```

Ex: For N9H20, logo execute address is 0x500000 (start block is 2)

```
[Logo File Name]
// Format:File name, Start Block (0 is decided by writer), execute address
Logo.dat, 0x2, 0x500000
```

Execute File Name

The format for this setting is:

```
[Execute File Name]
File name, start block, execute address
```

User can modify logo file name, spi flash start block, and execute address by this setting. If user want to burn logo binary file right after the previous image, user needs to set the start block to 0.

Ex: Burn execute image – demo.bin right after the previous image and execute address is 0x100000

```
[Execute File Name]
// Format:File name, Start Block (0 is decided by writer), execute address
demo.bin, 0x0, 0x1000000
```

Ex: Burn execute image – conprog.bin from block 9 and execute image is 0x00000000

```
[Execute File Name]
// Format:File name, Start Block (0 is decided by writer), execute address
conprog.bin, 0x9, 0x00000000
```

Data Verify

The format for this setting is:

```
[Data Verify]
0/1
```

User can enable or disable Data verify operation by setting. Disable Data verify operation can speed up the spi writing operation. (If SpiWriter can't find this setting, the default setting is disabled)

Ex: Disable Data verify operation.

```
[Data Verify]
// 1 to verify data, 0 to not to verify.
0
```

Chip Erase

The format for this setting is:

```
[Chip Erase]
0/1
```

User can enable or disable Chip Erase operation by setting. If Chip Erase Operation is disabled, SpiWriter will do sector erase to erase the spi sectors that will be wrote. (If SpiWriter can't find this setting, the default setting is disabled)

Ex: Disable Chip Erase operation.

```
[Chip Erase]
// 1 to erase whole spi flash, 0 to erase the sector which will be used.
0
```

4-Byte Address Mode Enable

The format for this setting is:

```
[4-Byte Address Mode Enable]
```

0/1

User can enable or 4 Byte-address mode by setting. It is used when SpiWriter can't get the actual size of spiflash.

Ex: Disable 4-Byte Address Mode.

```
[4-Byte Address Mode Enable]
// 1 to force to 4-Byte address mode, 0 to use default setting.
0
```

Sound Play

The format for this setting is:

[Sound Play]
0/1

SpiWriter can play PCM data when Burn Start/Burn Pass/Bun Fail and it has three built-in PCM data. User can enable or disable Sound Play operation by setting. (If SpiWriter can't find this setting, the default setting is disabled)

Ex: Enable Sound Play operation.

```
[Sound Play]
// 1 to enable sound play, 0 to disable sound play.
1
```

Sound Play Volume

The format for this setting is:

[Sound Play Volume]
0/1

SpiWriter can control Sound Play operation Volume by setting. The minimum value is 0 and maximum is 63. (If SpiWriter can't find this setting, the default setting is 31)

Ex: Set Sound Play operation Volume to maximum Volume

```
[Sound Play Volume]
// Maximum is 63
63
```

Write Root

The format for this setting is:

[Write Root]
0/1/2/3

SpiWriter can write W7M SPI Flash root key.

Ex: Write W74M SPI Flash Root Key 0

```
[Write Root]
//Index is from 1 to 4, 0 to disable Write Root Key
0
```

Flow Status Sound File Name

The format for this setting is:

[Flow Status Sound File Name - Start]
File name

[Flow Status Sound File Name - Pass]
File name

[Flow Status Sound File Name - Fail]
File name

SpiWriter allows user to use their own PCM data when Burn Start/Burn Pass/Bun Fail. User set the PCM file name for Burn Start/Burn Pass/Bun Fail stage by setting. (If SpiWriter can't find this setting, it will play default PCM data when Sound Play is enabled)

Ex: Enable Sound Play operation.

```
[Flow Status Sound File Name - Start]
StartBurn.pcm

[Flow Status Sound File Name - Pass]
BurnComplete.pcm

[Flow Status Sound File Name - Fail]
BurnFail.pcm
```

GPIO Flow Status

The format for this setting is:

[GPIO Flow Status - Start]
GPIO pin, Active Level

[GPIO Flow Status - Pass]
GPIO pin, Active Level

[GPIO Flow Status - Fail]
GPIO pin, Active Level

SpiWriter allow user to control GPIO pin for status check when Burn Start/Burn Pass/Bun Fail. SpiWriter will set the pin function setting for the setting and set the GPIO output level to inactive level after getting the setting. User can select GPIO pin and active level by this setting. (If SpiWriter can't find this setting, the default setting is disabled)

Ex: Set GPB11/GPA6/GPA7 for Status Check

```
// Format:GPIO Pin, Active Level, Example:GPA6, 1
[GPIO Flow Status - Start]
GPB11, 0

[GPIO Flow Status - Pass]
GPA6, 1

[GPIO Flow Status - Fail]
GPA7, 1
```

User Image File name

The format for this setting is:

```
[User Image File name]
File name, start block
```

User can set User image file name and spi flash start block by this setting. SpiLoader allow user to set 18 User image files. If user want to burn binary file right after the previous image, user needs to set the start block to 0.

Ex: Burn 4 User Image files, and set the start block to 70, 75, 77, and 82 for them.

```
// Format for User Image:File name, Start Block (0 is decided by writer)
// Ex.Image0.bin, 0x46 (Start block is 70)
// Maximum User Image Files is 18.
[User Image File name]
Image0.bin, 0x46

[User Image File name]
Image1.bin, 0x4B

[User Image File name]
Image2.bin, 0x4D

[User Image File name]
Image3.bin, 0x52
```

2.3. TurboWriter.ini

SpiWriter supports the same function as Turbowriter to provide writing some user configuration setting in the boot code header to adjust the setting of N9H20. Please use the same **TurboWriter.ini** file that used by Turbowriter for SpiWriter.

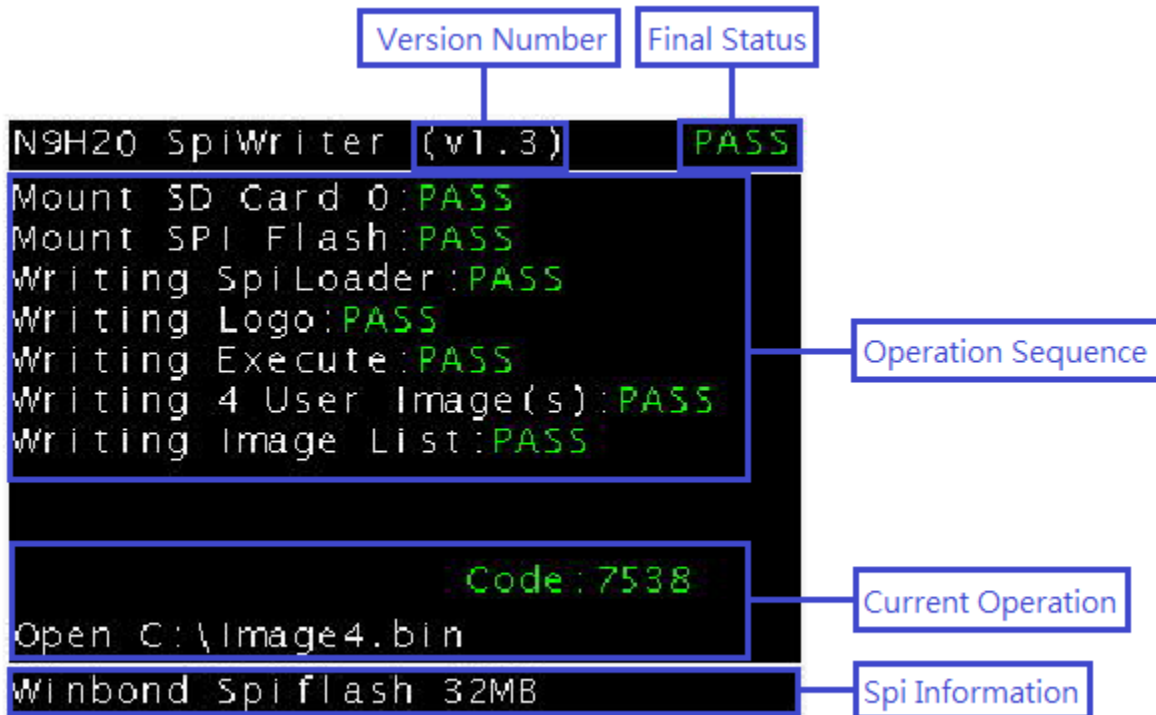
The TurboWriter.ini file provides some sections as below:

```
[ADDRESS]
ADDRESS = 00900000
[CLOCK_SKEW]
DQS0DS = 00001010
CKDQSDS = 00888800

[N9H20 USER_DEFINE]
```

2.4. Operation

When the SD card 0 is prepared successfully and booting from Normal mode, it will show the target SD card burning status on the panel as below:



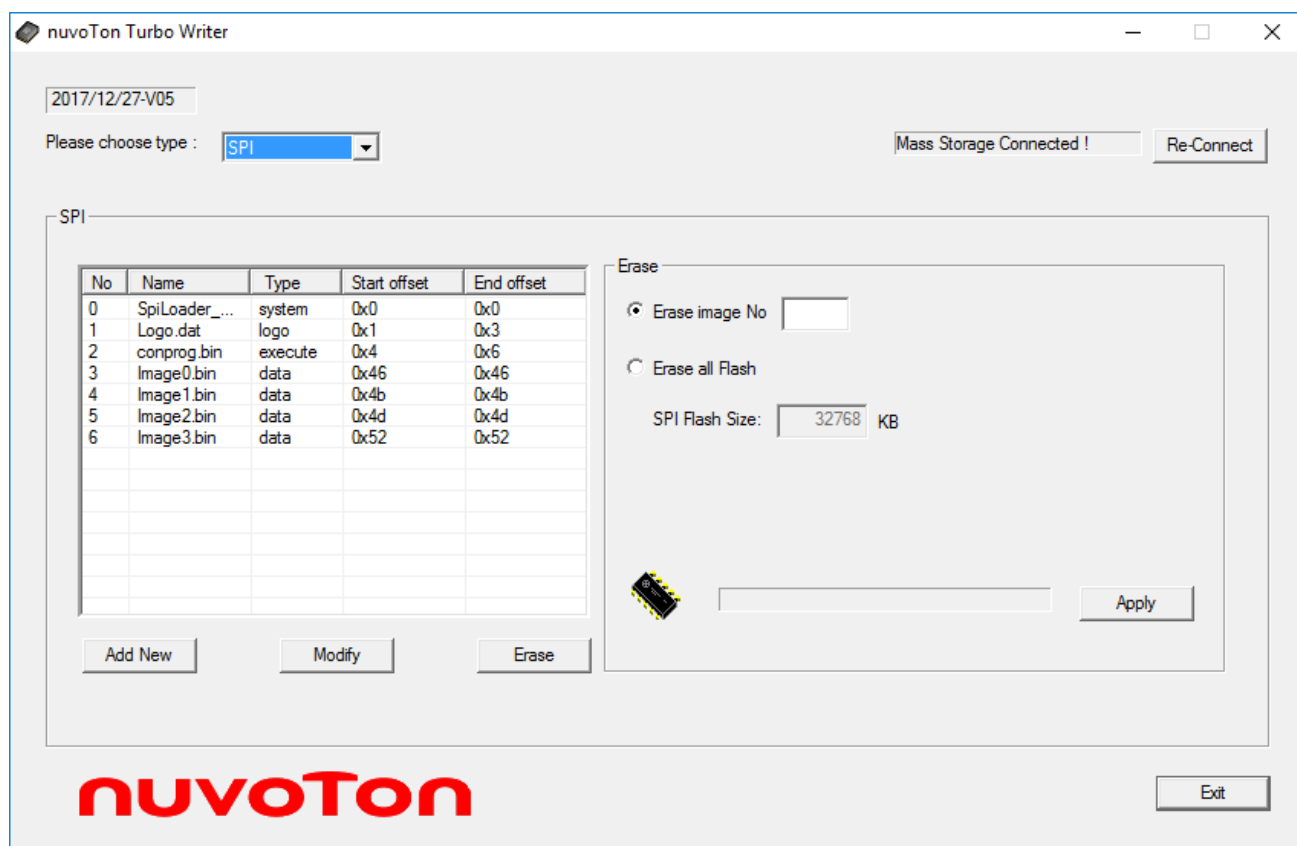
It can divide into several parts:

- Version Number: show this version number.
- Final Status: show the final operation status. If there are any fail items in the operation sequence, the final Status will be "FAIL".
- Operation Sequence: show the current operation progress.
- Current Operation: show more detail information for current operation. For example, it fails for some function, the code will show the return code for this.
- The spi flash and Fail operation Information: shows spi flash size and Fail operation status.

```
N9H20 SpiWriter (v1.3)
Mount SD Card 0:PASS
Mount SPI Flash:PASS
Writing SpiLoader:PASS
Writing Logo:PASS
Writing Execute:PASS
Writing 4 User Image(s):FAIL
Code:ffff8220
Open C:\Image0.bin
Open Image 0 File Fail
```

2.5. Sample

Here is the result for section 2.2.



3. Revision History

Version	Date	Description
V1.00.001	May, 2018	• Created

Important Notice

Nuvoton products are not designed, intended, authorized or warranted for use as components in equipment or systems intended for surgical implantation, atomic energy control instruments, aircraft or spacecraft instruments, transportation instruments, traffic signal instruments, combustion control instruments, or for any other applications intended to support or sustain life. Furthermore, Nuvoton products are not intended for applications whereby failure could result or lead to personal injury, death or severe property or environmental damage.

Nuvoton customers using or selling these products for such applications do so at their own risk and agree to fully indemnify Nuvoton for any damages resulting from their improper use or sales.