FA93
SPI Loader
Reference Guide

V1.0

Publication Release Date: Mar. 2011

Support Chips:
W55FA Series

Support Platforms:
Non-OS

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.

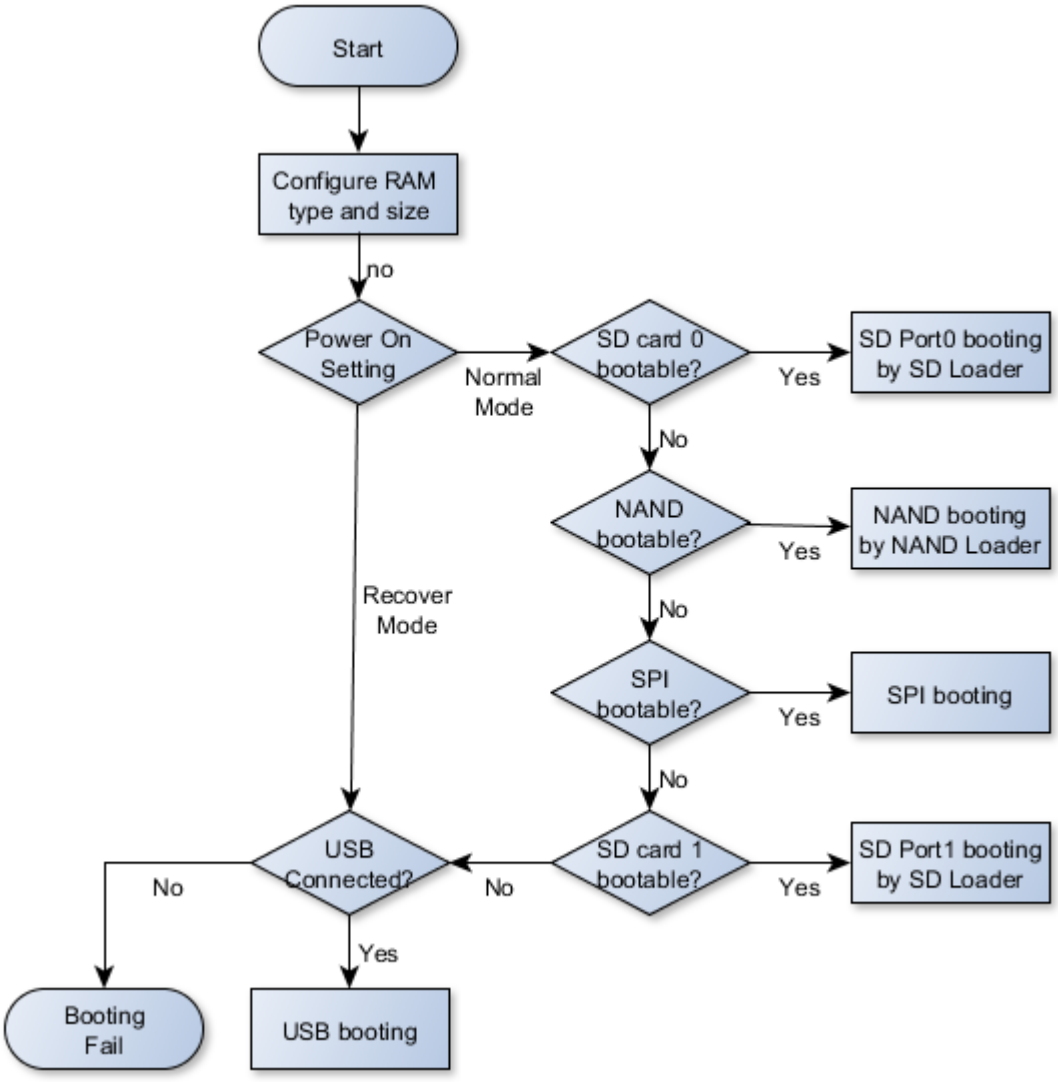
Table of Contents

1. General Description

FA93/VA93 Non-OS library consists of a set of libraries. These libraries are built to access those on-chip functions such as VPOST, APU, SIC, USBH, USBD, GPIO, I2C, SPI and UART, as well as File System (NVTFAT), USB MassStorage devices (UMAS) and NAND Flash devices (GNAND). This document describes the basic function of SPI Loader. With this introduction, user can quickly understand the SPI Loader on FA93/VA93 micro processor.

2. SPI Loader Overview

FA93/VA93 built-in 16K bytes IBR (Internal Booting ROM) where stored the boot loader to initial chip basically when power on, and then try to find out the next stage boot loader from different type of storage. It could be SD card, NAND, SPI Flash, or USB storage. The search sequence by IBR as below



The boot loader in IBR will hand over the chip controlling to SPI Loader if SD card 0 and NAND flash are not for booting. SPI Loader is a firmware stored at SPI Flash address 0x00000000.

2.1. SPI Loader Introduction

Because IBR SPI Booting Read operation takes more time than other booting, we hope the code size of SPI loader is as small as possible. The SPI Loader has two modes for application and they have the following features:

Normal-SpiLoader

- Check, load, and display Logo image if it existed at SPI flash
- Check and load next firmware if it existed at SPI Flash
- Hand over chip controlling to next firmware.

Please select target " NormalLoader" of SpiLoader project to build the Normal-SpiLoader.
The structure of SPI flash for SPI Loader is the same as NAND Loader and the basic unit of SPI Loader is one block (64KB). Here is an example for Normal-SpiLoader.

	SPI Loader	**Logo Image**	**Execute Image**
Image No.	0	1	2
Image Name	*File name for SPI Loader on host*	*File name for Logo image on host*	*File name for Execute Image on host*
Image Type	System Image	Logo	Execute
Image execute address	***0x700000***	**0x500000**	**0x800000**
Image start block	*Default value (0)*	*Behind Spi Loader*	*Behind Logo Image*

Execute-SpiLoader

- Check and load certain firmware if it existed at SPI Flash
- Hand over chip controlling to the firmware.

Please select target " ExecutelLoader" of SpiLoader project to build the Executel SpiLoader.
The mode is used to make sure NAND booting can always recovery from crash state. Execute-SpiLoader only loads one image and hand over chip controlling to it.

2.2. Execute-SpiLoader

Framework

The main Execute-SpiLoader framework can be divided into three parts.

1. Execute-SpiLoader
   SPI loader is loaded by IBR and in charge of the booting flow. User can modify the SPI loader to change the booting flow by himself. And in order to reduce the time for booting, SPI loader should be as small as possible.
2. Mass storage for NAND file-management and recovery

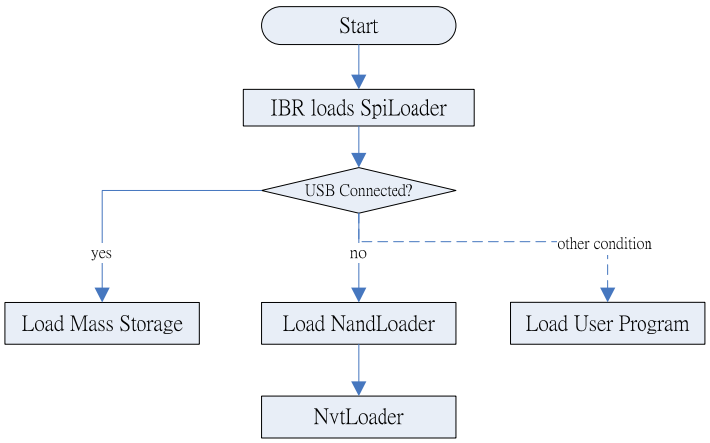
The mass storage can read/write file from/to the NAND flash and recovery from NAND crash by PC tool.

3. Main boot loader (It's NandLoader in the example) or user program

Example

The following image is an example of SPI Booting solution and it is used to introduce the solution in the following sections.

- Booting Flow



2-1 Booting flow example

- Flash Map

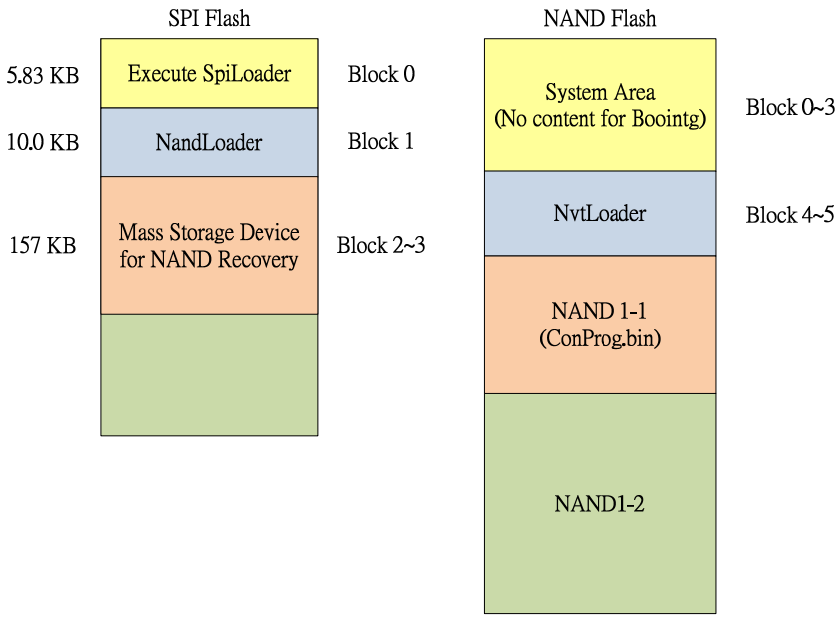


Figure 2-2 Flash Map

- Control of image loading

Execute-SpiLoader uses the image type field to find the code to be loaded and executed. User can modify Execute-SpiLoader to modify the Booting mode and use image type to add more code for other purposes, except System type (It's SpiLoader).

Image Type	Value
IMAGE_TYPE_DATA	0
IMAGE_TYPE_EXE	1
IMAGE_TYPE_ROMFS	2
IMAGE_TYPE_SYSTEM	3
IMAGE_TYPE_LOGO	4

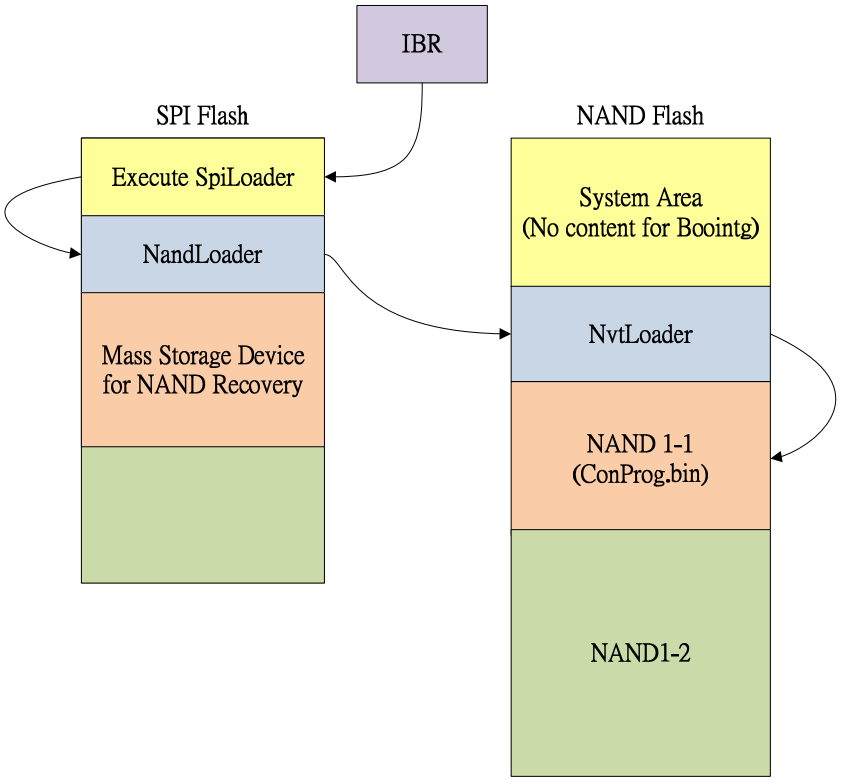
Current SpiLoader regards Mass Storage image as Data image and Main Loader image as Execute image. The source code is as follows.

```
if(inp32(PHY_CTL) & BIT31)
        image_type = IMAGE_TYPE_DATA;      /* Data Image is the code for execute */
else
        image_type = IMAGE_TYPE_EXE;       /* Execute Image is the code for execute */
```

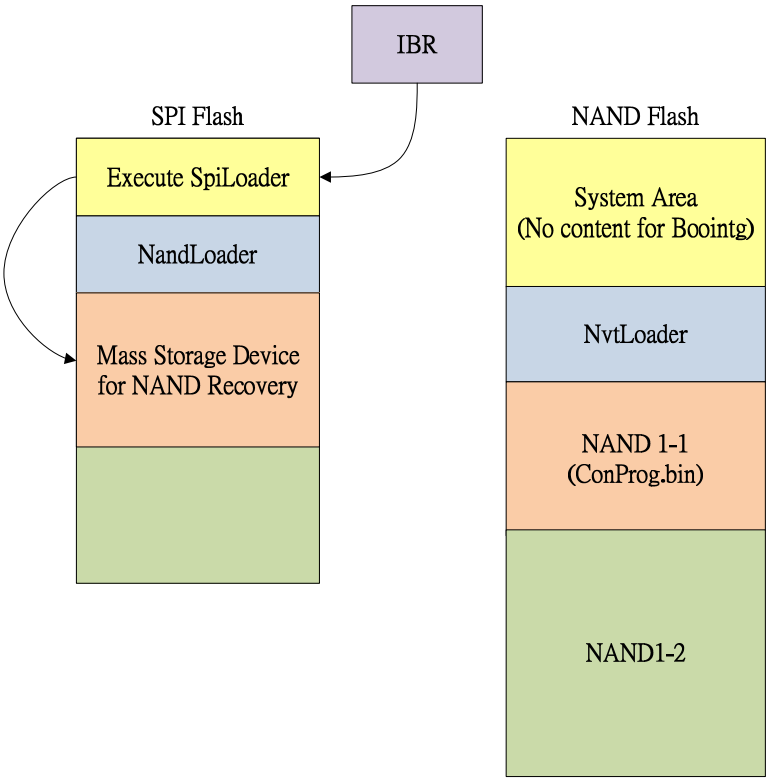
- Booting Flow

There are two modes for SPI booting example: Normal mode and USB mode.

- SPI Booting Normal mode
  IBR→SpiLoader→NAND Loader→NvtLoader→Conprog.bin



■ SPI Booting USB mode
IBR→SpiLoader→Mass storage for NAND Recovery



Time for SPI Booting Normal and USB mode:

	SpiLoader	Mass Storage	NAND loader	Total time
SPI Booting Normal mode	230 ms	N/A	48 ms	278 ms
SPI Booting USB mode		760 ms	N/A	990 ms
NAND Booting	N/A		220 ms	220 ms

The structure of SPI flash for SPI Loader is the same as NAND Loader and the basic unit of SPI Loader is one block (64KB). Here is an example for Execute-SpiLoader.

	SPI Loader	**NandLoader**	**MSC for Recovery**
Image No.	0	1	2
Image Name	*File name for SPI Loader on host*		*File name for next firmware on host*
Image Type	System Image	Data	Execute
Image execute address	***0x700000***	**0x900000**	**0x900000**
Image start block	*Default value (0)*	*Behind Spi Loader*	*Behind Logo Image*

■ Environment

- Turbowriter

  Turbo Writer provides the header setting of boot loader image for FA93. Before running the tool, user **MUST** set the file TurboWriter.ini within the same folder. The file information is as follows.

  [Address]
  Address = **00700000**
  [CLOCK_SKEW]
  DQSODS = 00001010
  CKDQSDS = 00AAAA00

  The value of Address is the execution address of SpiLoader. (0x700000)

- SPI Flash

  ExecuteSpiLoader.bin –
  - ◆ Choose the type "SPI"
  - ◆ Set Image type "System Image"
  - ◆ Browse the file " ExecuteSpiLoader.bin"
  - ◆ Press the button "Burn"

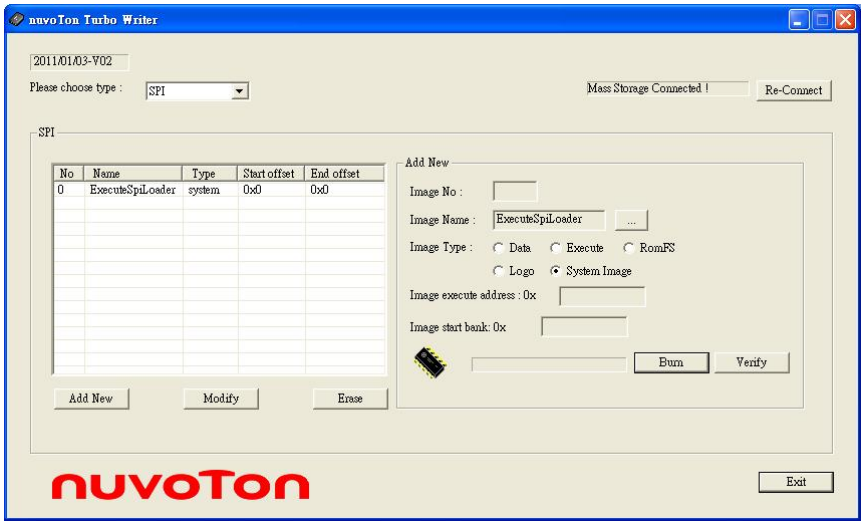


Figure 2-3 ExecuteSpiLoader.bin

NandLoader.bin –
- ◆ Set Image type "Execute"
- ◆ Image number "1"
- ◆ Browse the file "NandLoader.bin"
- ◆ Set the execute address: **0x900000**
- ◆ Set the start block number: **0x1**
- ◆ Press the button "Burn".

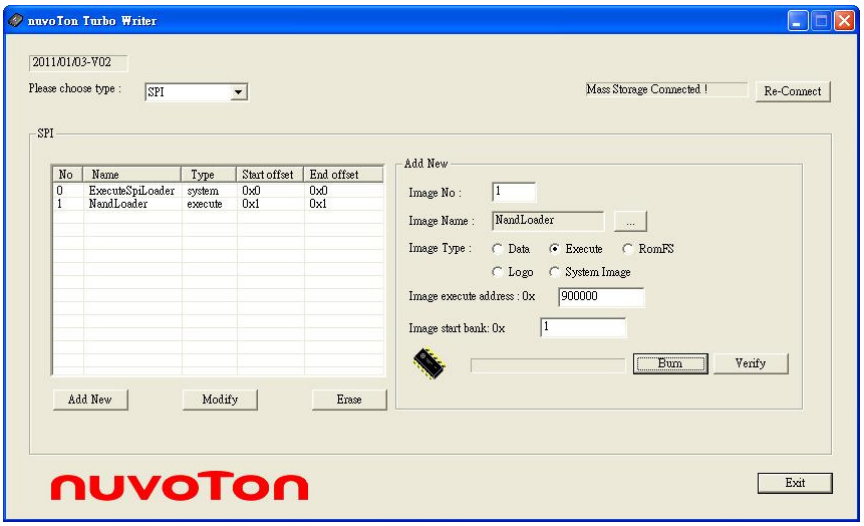


Figure 2-4 NandLoader.bin

Mass.bin for NAND Recovery–

- ◆ Image number “2”
- ◆ Set Image type “Data”
- ◆ Browse the file “Mass.bin”
- ◆ Set the executed address: **0x900000**
- ◆ Set the start block number: **0x2**
- ◆ Press the button “Burn”

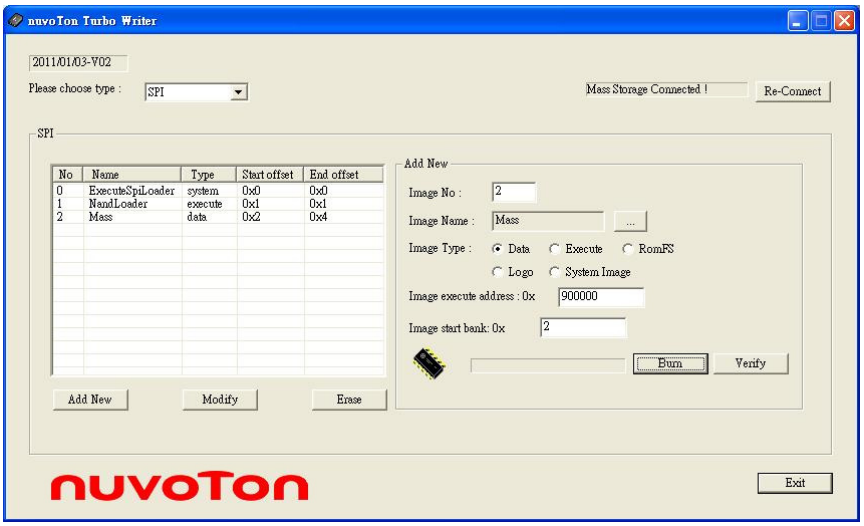


Figure 2-5 Mass.bin

- ■ NAND Flash

  Please erase the system image in the NAND Flash for SPI booting. The only necessary image in NAND flash is NvtLoader (it is in charge of Linux kernel image loading).

NvtLoader.bin –

- ◆ Set Image type “Execute”
- ◆ Image number “1”
- ◆ Browse the NvtLoader file
- ◆ Set the image execute address: **0x800000**
- ◆ Set the start block number: **0x4.** Because the original NAND Loader occupies block 0~3, so we could select block 4 to burn the NvtLoader file.
- ◆ Press the button “Burn”.

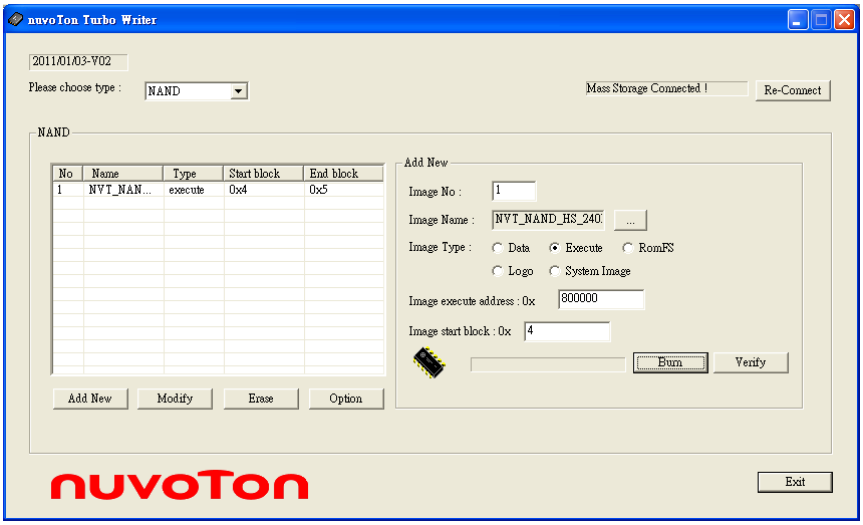


Figure 2-6 NvtLoader.bin

3. Revision History

Version	Date	Description
V1.0	Mar. 3, 2011	● 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.