ARM926EJ-S™ Based

32-bit Microprocessor

NUC980 SD Writer User Manual

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

Table of Contents

1 OVERVIEW

The SD Writer utility is a tool used for mass production or firmware update. When the NUC980 boots from SD, the SD Writer writes files stored in a SD card to NAND Flash, SPI NOR/NAND Flash, or eMMC according to the settings of config file stored in the root directory of SD card file system. This user manual will guide you to prepare the files in the SD card and modify the config file.

2 SD CARD CONTENT

The SD writer (*SD_Writer.bin*) is programmed in the reserved sectors after MBR and before file system. The SD card file system should contain a SD Writer configuration (config) file and the files to be programmed. The config file must reside in the root directory of SD card. The SD writer writes files according to the config file.

User has to use NuWriter to program the SD writer into a SD card, and then format the SD card. Afterwards, copy other files into the SD card by PC. This chapter provides two SD card content examples — one for Linux and the other for Non-OS.

2.1 Program SD Writer and Format SD Card

User has to use NuWriter to write SD writer into SD card and format SD card.

2.1.1 Program SD Writer

Use the NuWriter to program SD Writer to SD card with following steps:

1. Choose type: Select "**eMMC/SD**"
2. Image Name: Select *SD_Writer.bin*
3. Image Type: Select "**Loader**"
4. Image execution address: Set 0x8000
5. Click "**Program**" button

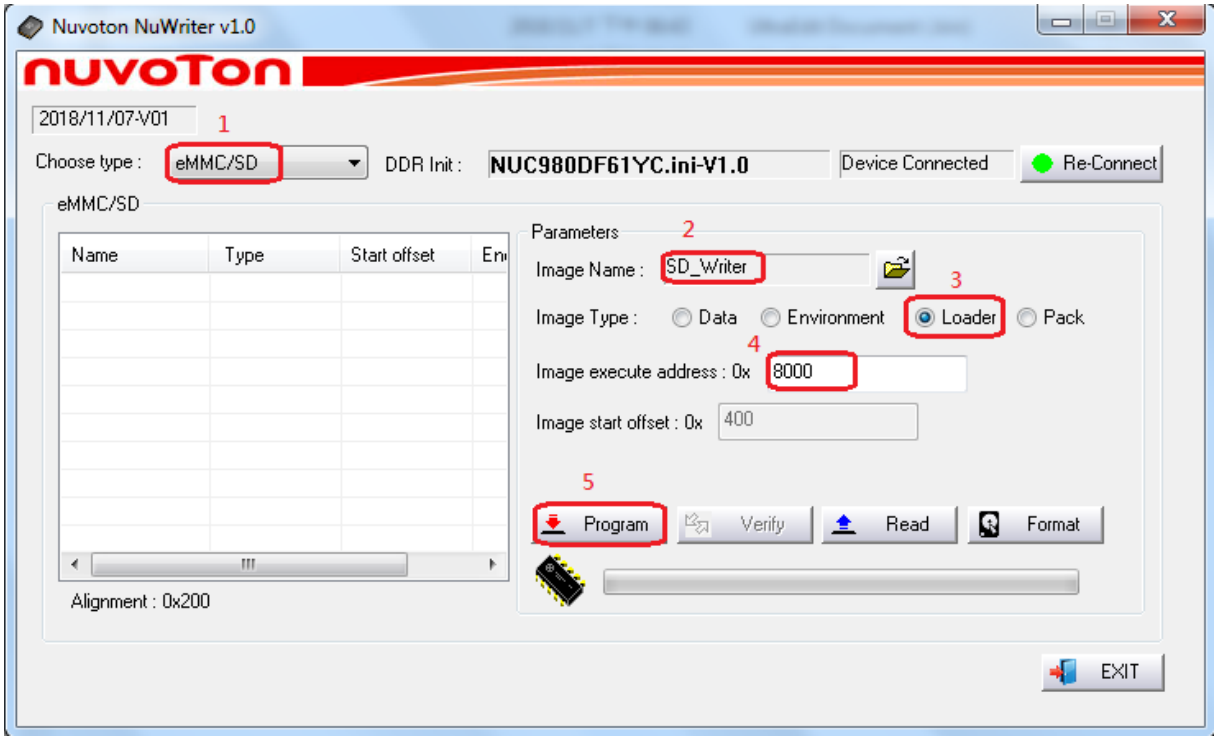

Figure 2-1 SD_Writer Programming Setting

After programming done, the NuWriter displays the End offset of SD Writer. Remember to reserve this space for SD writer.

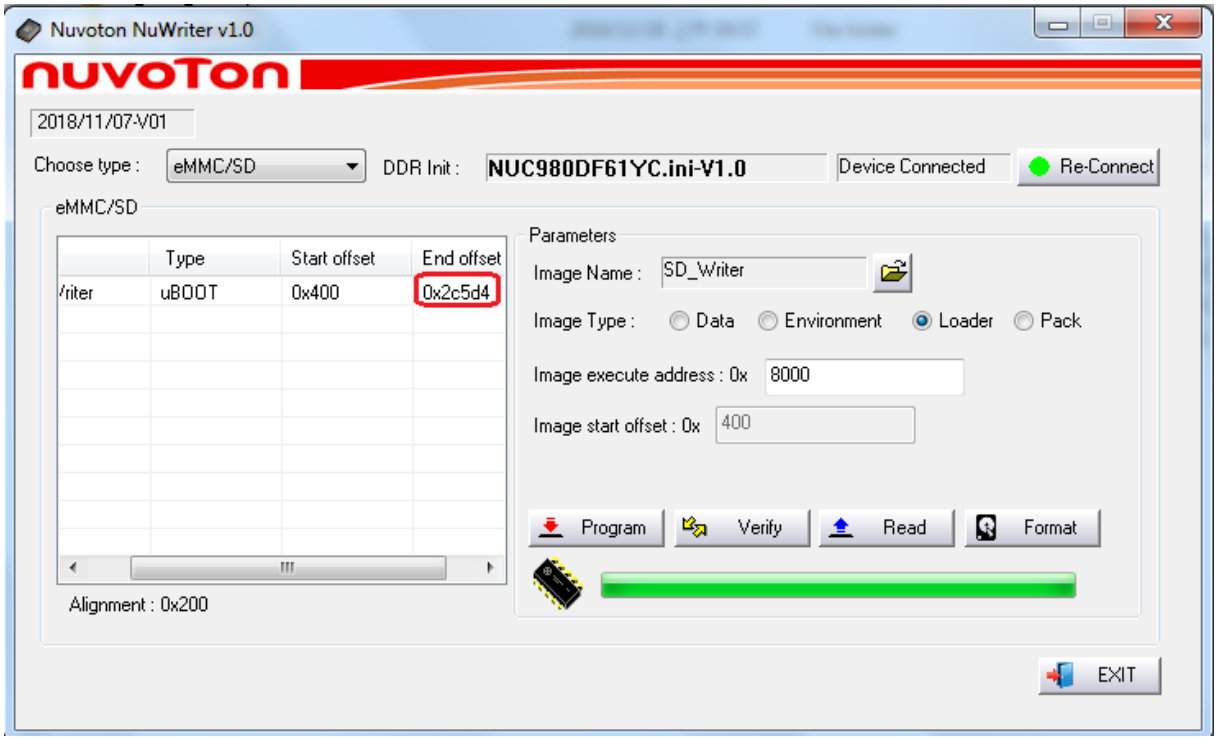

Figure 2-2 Check SD_Writer Image Size

In this case, the end offset of SD Writer is 0x2c5d4. The size of one sector is 512(0x200) bytes. That is, the end sector of SD Writer is (0x2c5d4/0x200 = 355). Hence, users have to reserve 355 sectors minimal for SD Writer while they format SD card.

2.1.2 Format SD Card

Use the NuWriter to format SD card. Remember to reserve space for the SD Writer. The space depends on the size of SD Writer. In the example, as mentioned in section 2.1, reserve 355 sectors for SD Writer.

1. Choose type: Select "**eMMC/SD**"
2. Click "**Format**" button.

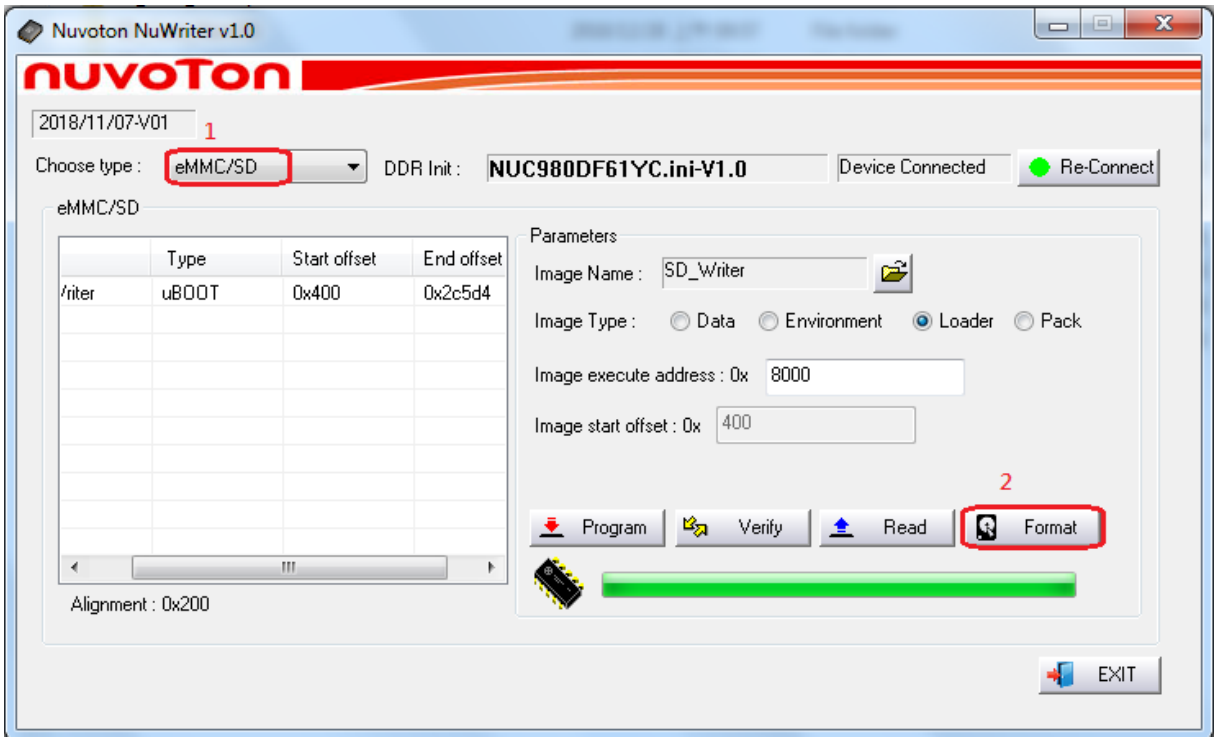

Figure 2-3 Format SD Card

Fill in 355 in Reserve space, and then Click "**OK**" button.

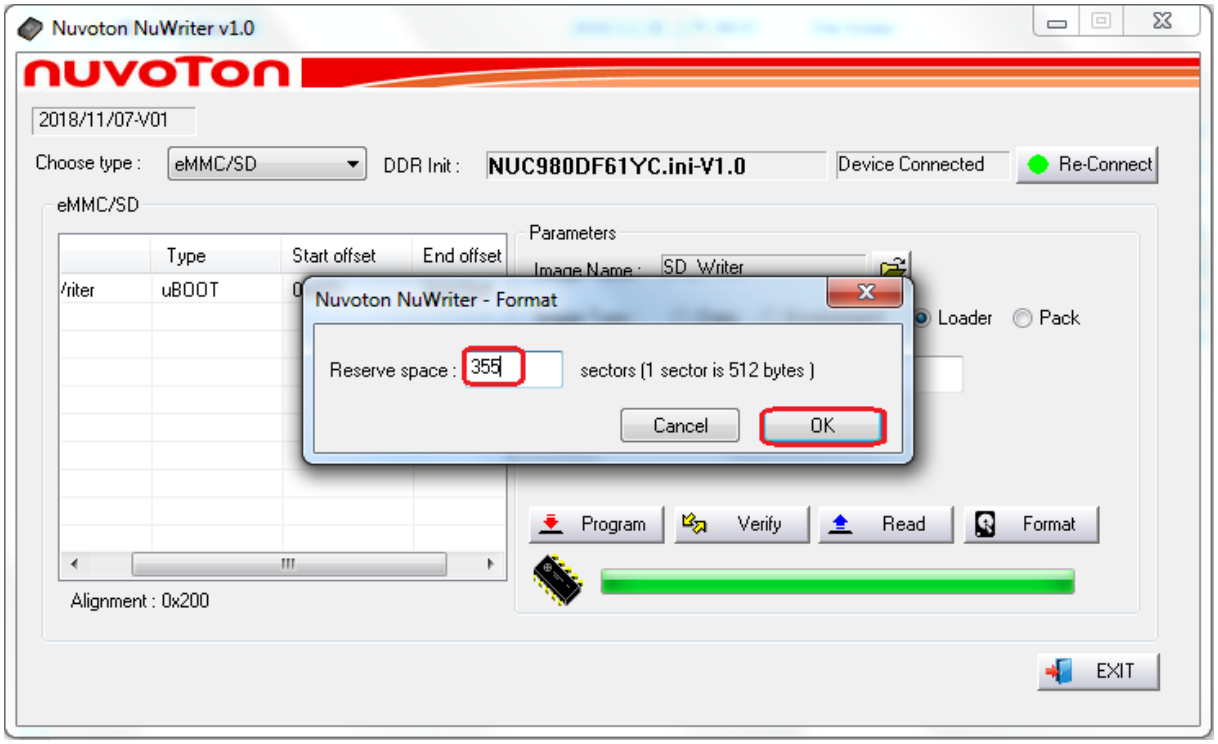

Figure 2-4 Set Reserved Space Size

2.2 SD Card Content Structure

After programming the SD Writer and formatting SD card. Users could copy necessary files into

SD card using PC. The content is different per user application. The following provides two SD card content examples – one for Linux and the other for Non-OS.

2.2.1 Linux Kernel

For a Linux system, the SD card contains the config file, DDR initialization file, main U-Boot binary file, SPL U-Boot binary file, U-Boot environment variables file, and Linux kernel image.

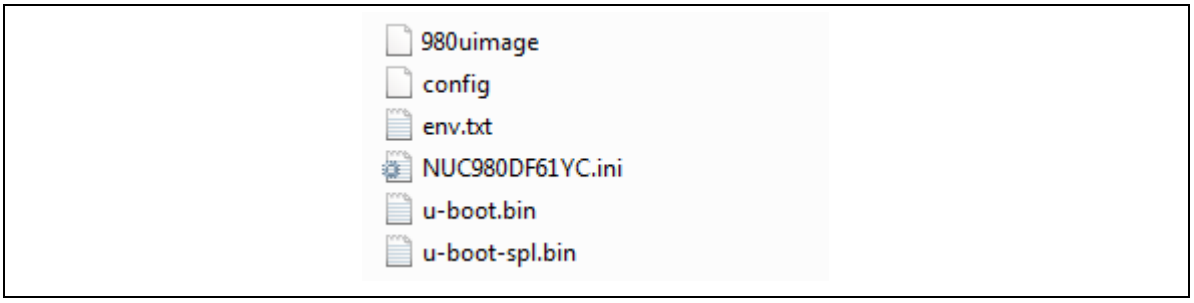

Figure 2-5 SD Card Content For Linux System

- **980uimage**: NUC980 Linux Kernel image
- **config**: Configuration file that tells SD writer how to write files
- **env.txt**: U-Boot environment variables file
- **NUC980DF61YC.ini**: DDR initialization file for NUC980DF61YC
- **u-boot.bin**: Main U-Boot binary file
- **u-boot-spl.bin**: SPL U-Boot binary file.

2.2.2 Non-OS

For Non-OS system, the SD card contains config file, DDR initialization file, Main U-Boot binary file, SPL U-Boot binary file, U-Boot environment variables file, and application program.

Below is an example of files in the root directory of SD card.

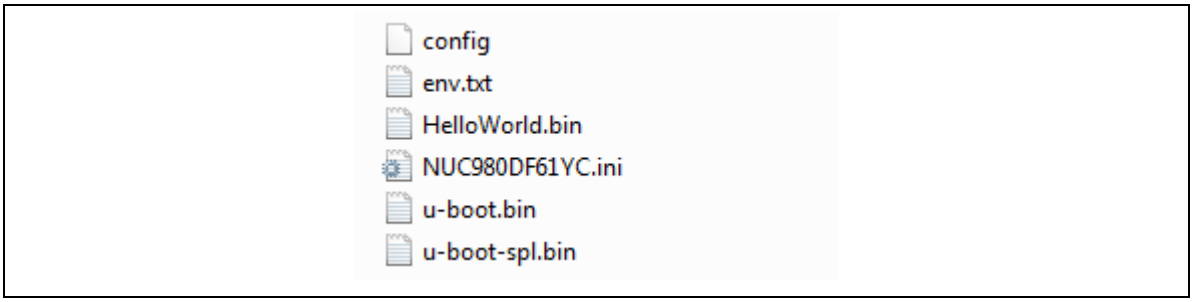

Figure 2-6 SD Card Content For Non-OS System

- **config**: Configuration file that tells SD writer how to write files
- **env.txt**: U-Boot environment variables file
- **HelloWorld.bin**: application program
- **NUC980DF61YC.ini**: DDR initialization file for NUC980DF61YC
- **u-boot.bin**: Main U-Boot binary file
- **u-boot-spl.bin**: SPL U-Boot binary file.

3 CONFIG FILE

The config file must reside in the root directory of SD card. Other files are optional per user's need. The config file guides SD writer how to program files stored in SD card. The file name of configuration file must be "config", all letters are in lowercase. It provides the user a flexible way to do a restricted modification without modifying the source code of SD writer.

This chapter has an example for programming Main U-Boot binary file, SPL U-Boot binary file, U-Boot environment variables file, and Linux kernel image.

3.1 Config Support Options

3.1.1 Write type [Type]

This option specifies the storage media. Select 1~4 according to the storage media to program.

- 1: SPI NOR Flash
- 2: SPI NAND Flash
- 3: NAND Flash
- 4: eMMC

3.1.2 DDR initialization File [DDR]

The option name of DDR initialization file is [DDR]. The format is file name of DDR initialization file. The DDR initialization file can get from the directory "sys_cfg" in NuWriter. There are five DDR initialization files currently.

- NUC980DF61YC.ini
- NUC980DF71YC.ini
- NUC980DK61Y.ini
- NUC980DK61YC.ini
- NUC980DR61Y.ini

New DDR initialization files or updated files may be added into NuWriter in the future. Users could always find latest file in "sys_cfg" directory of NuWriter tool.

3.1.3 Loader [Loader]

The option name of Loader is [Loader]. Format is file name, execution address.

For instance, the loader for NAND boot is u-boot-spl.bin and execution address is 0x200. User has to fill in:

```
u-boot-spl.bin, 0x200
```

3.1.4 Environment Variable [Env]

The section name of Environment variable is [Env]. Format is file name, offset

For instance, the file name of environment variable is env.txt and offset is 0x80000. User has to fill in:

```
env.txt, 0x80000
```

Note that the default offset of U-Boot environment variable is 0x80000. If you set different offset here, you also have to rebuild U-Boot with new setting.

3.1.5 Data [Data]

The SD writer supports up to 10 Data files. The section name of these data files are [Data0],

[Data1], [Data2], … to [Data9].

User can choose any section from those ten sections. For example, if user has 3 data files, the 3 sections can be [Data0], [Data1], [Data2], or user can also choose section [Data1], [Data3], [Data5].

The format of Data section is file name, offset. For example, user wants to write u-boot.bin to 0x100000, and write 980uimage to 0x200000. User has to fill in:

```
[Data0]
u-boot.bin, 0x100000

[Data1]
980uimage, 0x200000
```

3.1.6 User Defined [UserDefine]

The SD writer supports user definition for SPI. User can define page size and spare area size of SPI NAND. Besides, user can define SPI Quad mode read command, read status command, write status command, Quad mode enable value of SPI Flash status register, and dummy byte. Section name of user defined SPI attributes is [UserDefine].

The format of page and spare area size is PageSize=SPI NAND page size, SpareArea=SPI NAND spare area size. Note that the size is in decimal format.

For example, user defines SPI NAND page size is 2048 and spare area size is 64, he has to fill in:

```
PageSize=2048, SpareArea=64
```

The SPI NAND page size and spare area size differs from SPI NAND Flash model. This information can be found in SPI NAND's datasheet. Below is an example that SPI NAND's datasheets describes its page size and spare area size.

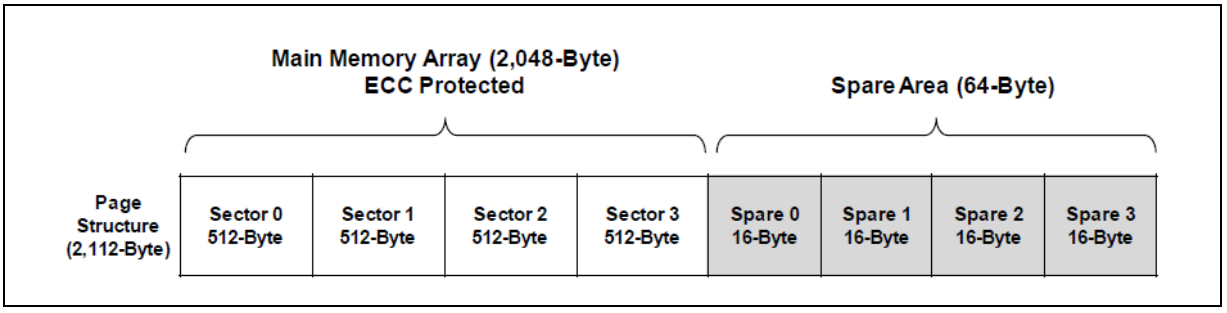

Figure 3-1 SPI NAND Flash Architecture

The format for Quad mode command is QuadReadCmd=Quad mode read command, ReadStatusCmd=Read status command, WriteStatusCmd=Write status command, StatusValue=Quad mode enable bit in SPI Flash status register, DummyByte=dummy byte length in Quad mode read command. Note that the command here is in hex format. Below is an example:

```
QuadReadCmd=0x6B, ReadStatusCmd=0x35, WriteStatusCmd=0x31, StatusValue=0x2,
DummyByte=0x1
```

SPI Quad mode read command and relative information can also be found in SPI Flash datasheet .

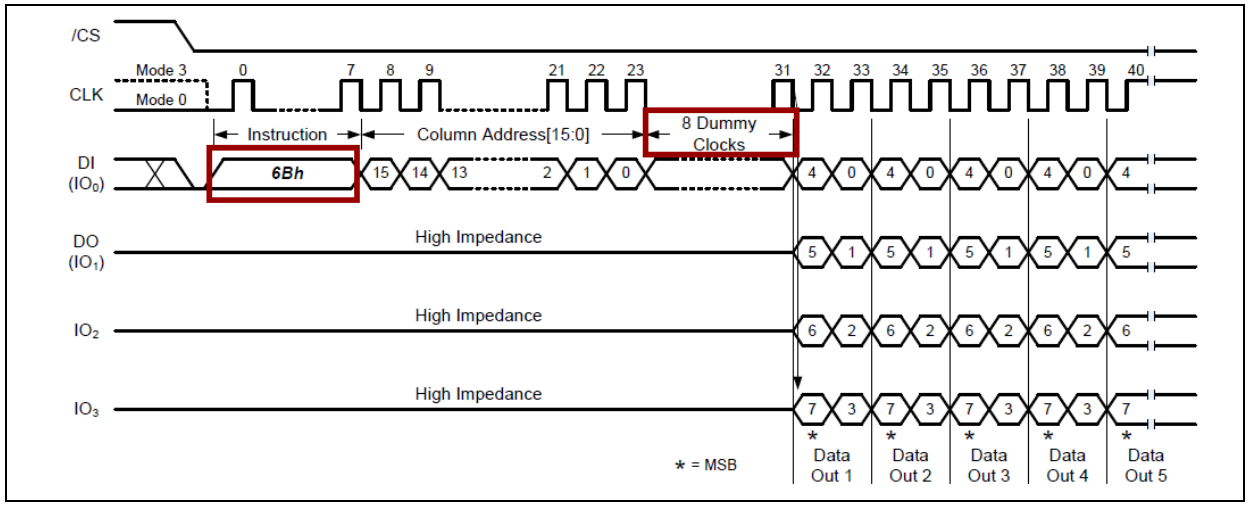

Figure 3-2 SPI NAND Flash Quad Read Command

3.2 Example of config File

The following config file is an example to write SPL U-Boot (u-boot-spl.bin), Main U-Boot (u-boot.bin), environment variable file (env.txt), and Linux kernel image (980uimage) to NAND Flash.

```
[TYPE]
//Format: write type (1~4)
//1: SPI NOR  2: SPI NAND  3: NAND  4: EMMC
3

[DDR]
//Format: file name
// NUC980DF61YC.ini
// NUC980DF71YC.ini
// NUC980DK61Y.ini
// NUC980DK61YC.ini
// NUC980DR61Y.ini
NUC980DF61YC.ini

[Loader]
//Format: file name, execution address
u-boot-spl.bin, 0x200

[ENV]
//Format: file name, offset
env.txt, 0x80000

[Data0]
//Format: file name, offset
```

```
u-boot.bin, 0x100000


[Data1]
//Format: file name, offset
980uimage, 0x200000


[UserDefine]
PageSize=2048, SpareArea=64
QuadReadCmd=0x6B, ReadStatusCmd=0x35, WriteStatusCmd=0x31, StatusValue=0x2,
DummyByte=0x1
```

There are some limitations of config file as listed 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 changed.

4 PROJECT SOURCE FILE

This chapter gives an overview of SD Writer source code. Users can modify the source code per their application.

4.1 Development Environment

Keil IDE is used as Non-OS BSP development environment. To support ARM9, MDK Plus or Professional edition shall be used.

Feature	MDK Edition			
	Professional	Plus	Essential	Lite
	All-in-one solution including Middleware	Supports all microcontroller cores and Middleware	Supports selected Cortex-M	Free with code size limit: 32 KBytes
Device Support				
Arm Cortex-M0/M0+/M3/M4/M7	✔	✔	✔	✔
Arm Cortex-M23/M33 Non-secure only	✔	✔	✔	✘
Arm Cortex-M23/M33 Secure and non-secure	✔	✔	✘	✘
Armv8-M Architecture Models including FastModel	✔	✘	✘	✘
Arm SecurCore®	✔	✔	✘	✘
Arm7™, Arm9™, Arm Cortex-R4	✔	✔	✘	✘

Figure 4-1 MDK Edition Selection

4.2 Function of Source File

After opening the SD Writer project in Keil IDE, the project window shows a project structure as shown below.

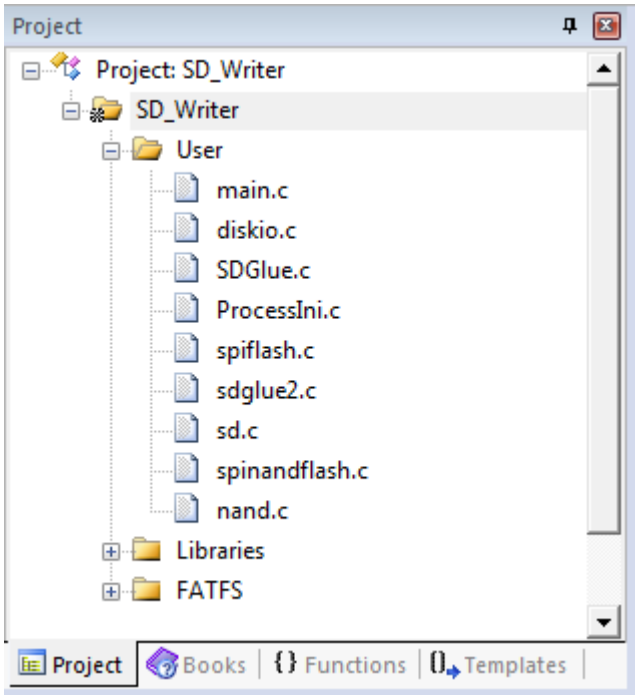

Figure 4-2 SD Writer Project Structure

The following lists the files that may need to be modified during customization and their functions.

- **main.c**: main program for SD Writer
- **diskio.c**: Low level disk I/O module skeleton for FAT file system
- **SDGlue.c** and **sdglue2.c**: SD Glue function for FAT file system
- **ProcessIni.c**: Process config file
- **spiflash.c**: SPI NOR Flash driver
- **sd.c**: SD driver
- **spinandflash.c**: SPI NAND Flash driver
- **nand.c**: NAND driver.

5 REVISION HISTORY

Date	Revision	Description
2018.12.28	1.00	1. Initially issued.
2019.04.19	1.01	1. Editorial Change

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.