ARM926EJ-S™ Based

32-bit Microprocessor

NUC980 SD Writer 使用手冊

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

www.nuvoton.com

Table of Contents

1 NUC980 SD Writer 簡介 3
2 SD Card 內容 4
2.1 燒錄 SD Writer 以及格式化 SD 卡 4
2.1.1 Write SD Writer(SD_Writer.bin) to SD card 4
2.1.2 格式化 SD 卡 5
2.2 SD card 存放內容 7
2.2.1 Linux 內核 7
2.2.2 無作業系統 7
3 config 文件 8
3.1 CONFIG 所支援的設定 8
3.1.1 燒錄型別 [Type] 8
3.1.2 DDR 初始化文件 [DDR] 8
3.1.3 開機程式 [Loader] 8
3.1.4 環境變數文件 [Env] 8
3.1.5 資料[Data] 8
3.1.6 使用者定義 [UserDefine] 9
3.2 config 文件範例 10
4 源代碼簡介 12
4.1 開發環境 12
4.2 源代碼文件功能 12
5 修改歷史 14

1 NUC980 SD WRITER 簡介

NUC980 SD Writer 是一個量產或韌體更新的工具。當 NUC980 從 SD 開機時，SD Writer 從 SD 卡中讀取設定文件，根據設定文件中的設定將 SD 卡根目錄中的文件燒錄到 NAND flash, SPI NOR/NAND flash, 或 eMMC 中。 這個使用手冊會說明如何將 SD Writer 以及要燒錄的文件放置到 SD 卡中，並且介紹如何修改設定文件。

2 SD CARD 內容

SD Writer (*SD_Writer.bin*) 需要燒錄在 MBR 之後，文件系統開始的保留區域。而 SD 卡文件系統中必須放置設定文件 (config) 以及要燒錄的文件。

用戶使用 SD Writer 之前要先透過 NuWriter 將 SD Writer 燒錄到 SD 卡中，並且對 SD 卡格式化，然後再透過個人電腦將要燒錄的文件複製到 SD 卡。這個章節會提供兩個範例，分別針對 Linux 內核以及無作業系統 (Non-OS)，SD 卡中會放置不同的文件。

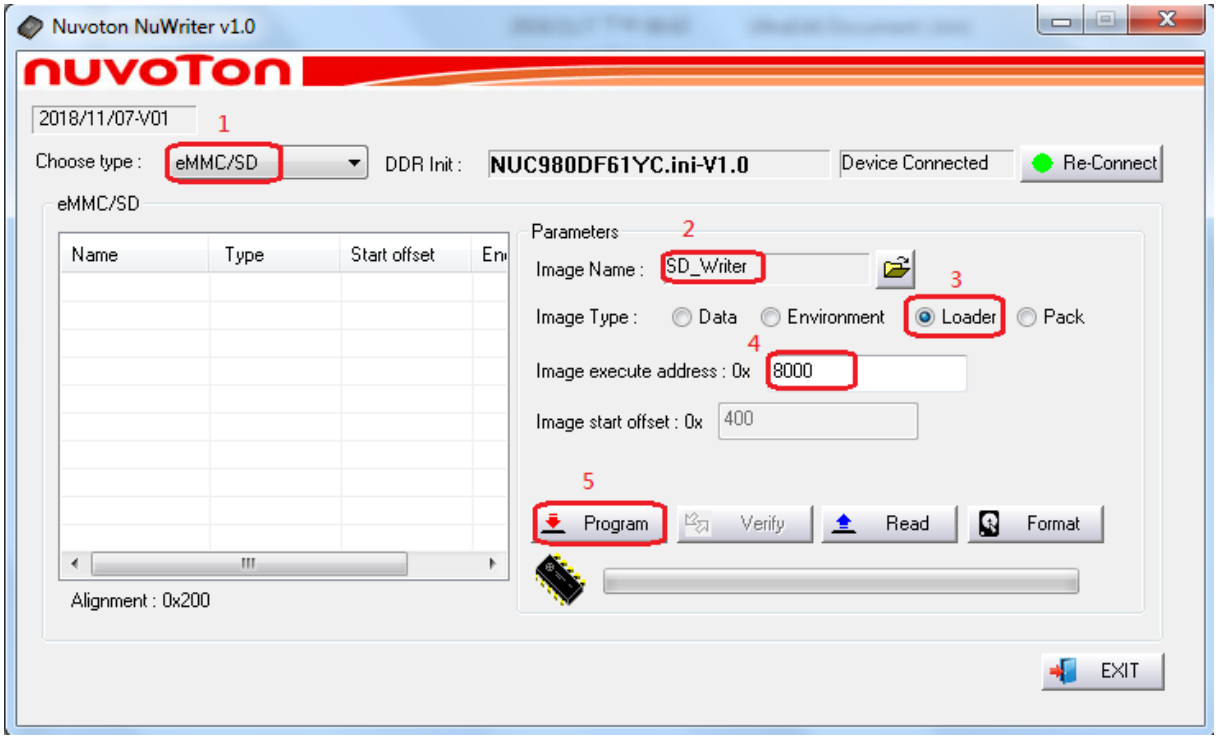
2.1 燒錄 SD Writer 以及格式化 SD 卡

使用者必須先透過 NuWriter 將 SD Writer 燒錄到 SD card 並且對 SD card 格式化。

2.1.1 燒錄 SD Writer

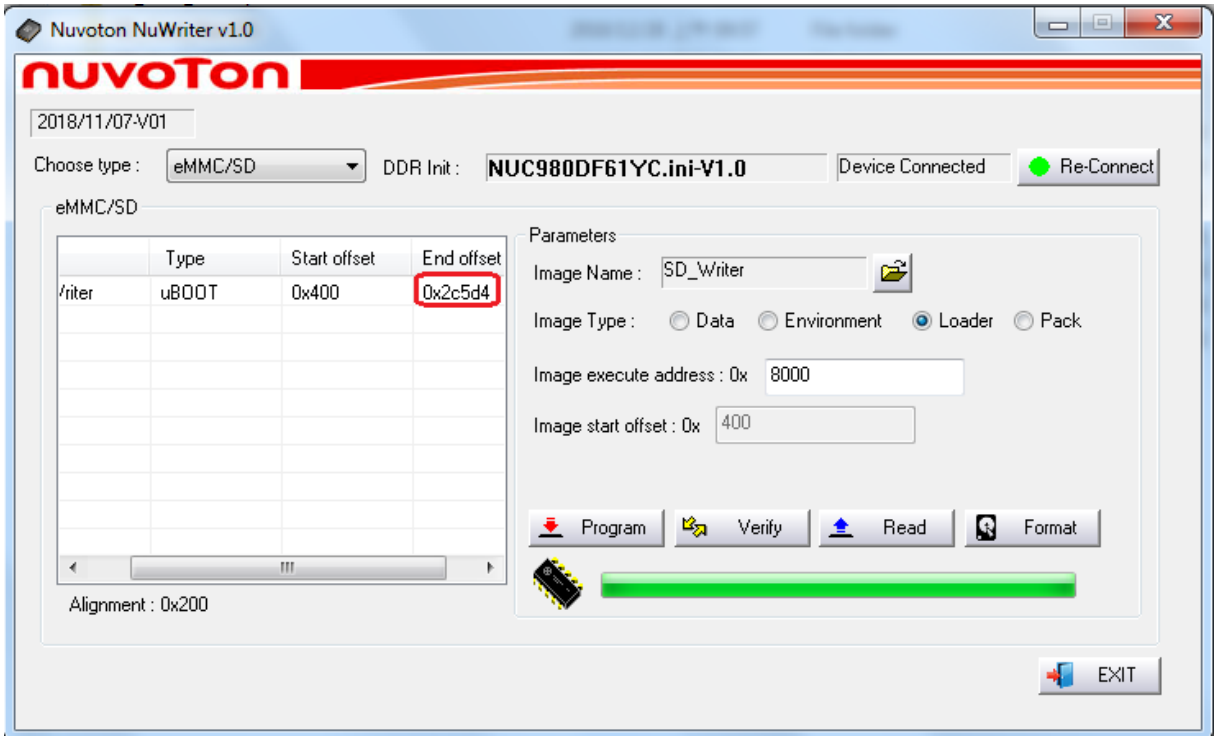
將 SD Writer 燒錄到 SD 卡的步驟如下:

1. Choose type : 選擇 “**eMMC/SD**”
2. Image Name : 選擇 *SD_Writer.bin*
3. Image Type : 選擇 “**Loader**”
4. Image execution address : 設置 0x8000
5. 按下 “**Program**” 按鍵



圖表 2-1 SD Writer 燒錄設定

當燒錄結束，NuWriter 會顯示 SD Writer 存放在 SD 卡的最後位址。當格式化 SD 卡時，必須保留這個空間給 SD Writer。



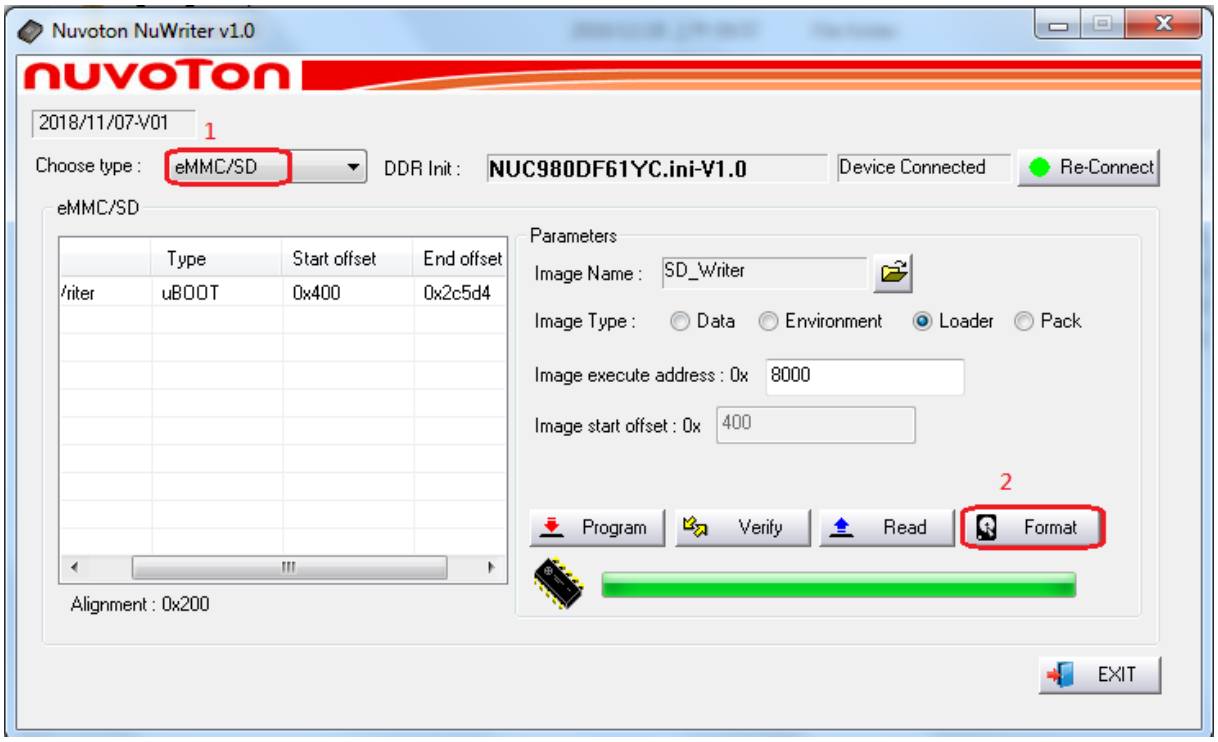
圖表 2-2 查看 SD Writer 映像檔大小

在這個例子中，SD Writer 存放在 SD card 中的 0x400~0x2c5d4。SD 卡一個 sector 是 512 位元，也就是說，SD Writer 最後一個 sector 是 (0x2c5d4/512 = 355)。因此，格式化 SD 卡時至少必須保留 355 個 sector 給 SD Writer。

2.1.2 格式化 SD 卡

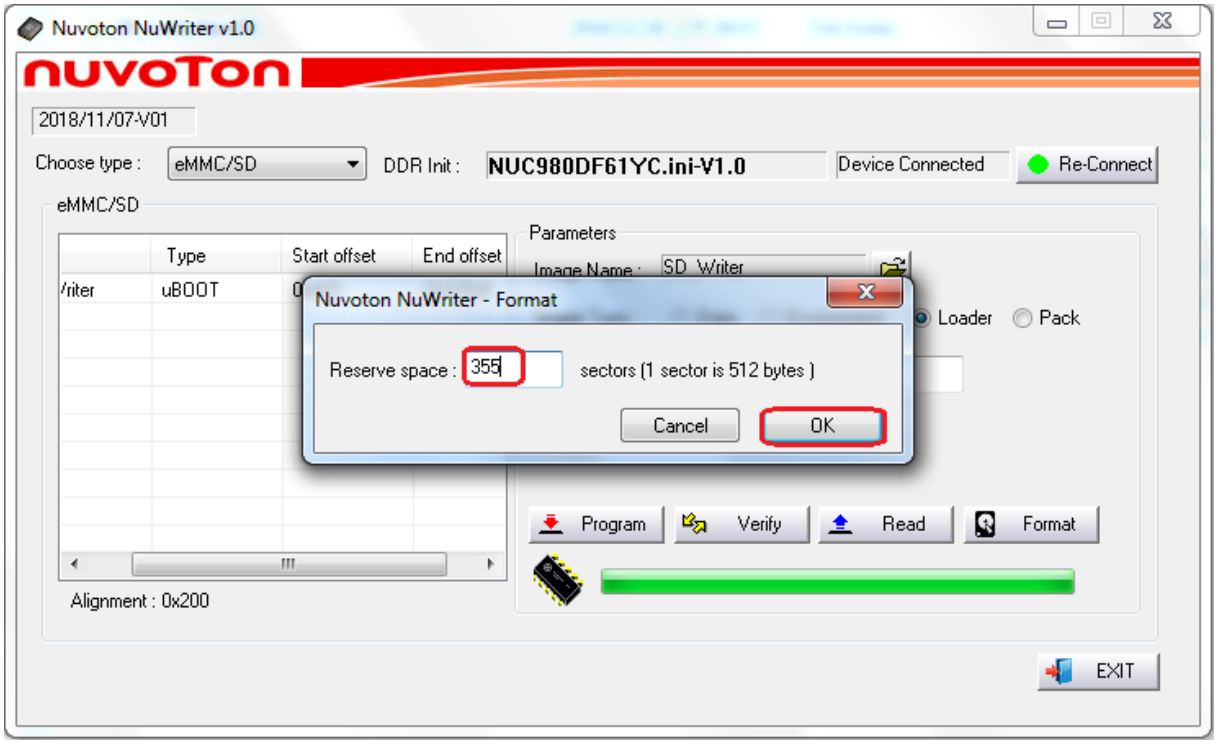
透過 NuWriter 格式化 SD 卡。 記得保留空間給 SD Writer。保留空間的大小會和 SD Writer 執行檔的大小有關，在本範例使用章節 2.1 範例中提到的，保留 355 個 sector 給 SD Writer。

1. Choose type : 選擇 “**eMMC/SD**”
2. 按下 “**Format**” 按鍵



圖表 2-3 格式化 SD卡

在 Reserve space: 輸入 355，然後按下 “**OK**” 按鍵。



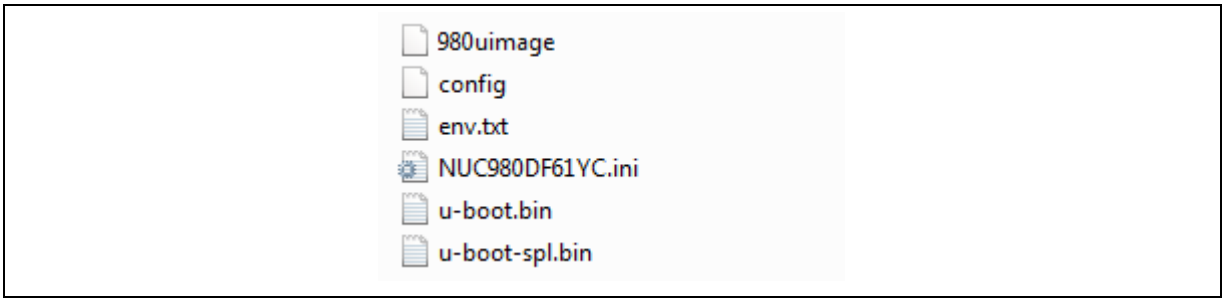
圖表 2-4 設定保留區域大小

2.2 SD card 存放內容

完成燒錄 SD Writer 以及對 SD 卡格式化後，就可以透過個人電腦將要燒錄的文件複製到 SD 卡中。 放置到 SD 卡中的文件會隨著使用者的應用不同，所需的文件也會不一樣。以下介紹 Linux 內核和無作業系統所需放置到 SD 卡中的文件。

2.2.1 Linux 內核

針對 Linux 內核的需求，SD 卡中需要放置 SD Writer 設定文件， DDR 初始化文件，U-Boot 文件，U-Boot 環境變數文件以及 Linux 內核文件。

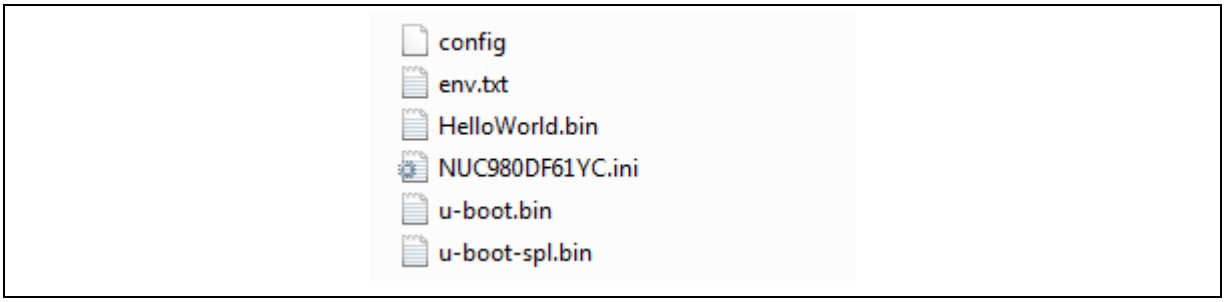


圖表 2-5 Linux 系統 SD卡內容

- 980uimage : NUC980 Linux 內核文件
- config : 存放如何燒錄文件的相關設定文件
- env.txt : U-Boot 環境變數文件
- NUC980DF61YC.ini : NUC980DF61YC 的 DDR 初始化文件
- u-boot.bin : 主要的 U-Boot 文件
- u-boot-spl.bin : SPL U-Boot 文件

2.2.2 無作業系統

如果使用者的應用是不需要作業系統，需要存放到 SD 卡中的文件會有些許差異。 當然，SD Writer 設定文件是必須的，SD 卡中還需要放置 DDR 初始化文件，U-Boot 文件，U-Boot 環境變數文件以及應用程式。下面是無作業系統所需存放到 SD 卡中的範例



圖表 2-6 Non-OS 系統 SD卡內容

- config : 存放如何燒錄文件的相關設定文件
- env.txt : U-Boot 環境變數文件
- HelloWorld.bin : 應用程式
- NUC980DF61YC.ini : NUC980DF61YC 的 DDR 初始化文件
- u-boot.bin : 主要的 U-Boot 文件
- u-boot-spl.bin : SPL U-Boot 文件

3 CONFIG 文件

config 文件是必要存在 SD 卡中根目錄的文件，其餘文件就依據使用者的需求來增減。此文件是告訴 SD Writer 如何燒錄 SD 卡中的文件。config 文件檔名必須為 “config”，所有字母都是小寫。用戶透過修改 config 文件，可以進行不同的燒錄。config 文件包含多個項目，這個章節會逐一介紹這些項目。

這個章節會提供一個 config 文件範例，將 U-Boot 文件，U-Boot 環境變數以及 Linux 內和燒錄到 NAND flash 中。

3.1 CONFIG 所支援的設定

3.1.1 燒錄型別 [Type]

燒錄型別的項目名稱為 [Type] ，格式請依欲寫入的儲存媒體填入 (1~4) 。

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

3.1.2 DDR 初始化文件 [DDR]

DDR 初始化文件的項目名稱為 “[DDR] ，格式請填寫 DDR 初始化文件名稱。DDR 初始化文件可以從 NuWriter 中的 “sys_cfg” 目錄中取得，目前提供五個 DDR 初始化文件。

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

未來可能新增或更改 DDR 初始化文件，使用者可從 NuWriter 中取得最新的 DDR 初始化文件。

3.1.3 開機程式 [Loader]

開機程式的項目名稱為 [Loader]。格式為： 開機程式名稱，執行位址。

例如， NAND 開機程式名稱為 u-boot-spl.bin，執行位址為 0x200，使用者必須填寫：

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

3.1.4 環境變數文件 [Env]

環境變數文件項目名稱為 [Env]。格式為：環境變數文件名稱，位移。

例如，環境變數文件名稱是 env.txt，位移 0x80000，使用者必須填寫:

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

U-Boot 環境變數文件的默認位移設置是 0x80000，如果使用者在這裡做不同的設置，也請同時修改 U-Boot 的設置。

3.1.5 擦除[Erase]

SD writer 支持擦除功能，包括整個芯片的擦除以及部分擦除。

使用者設定 EraseAll=1 達成擦除整個芯片的功能。

```
[Erase]
EraseAll=1
```

若要做部分擦除，使用者設定 “Start”=第一個擦除的塊編號,“Length”=擦除的塊個數. 下面範例就是要擦除塊編號 5 到 10。

```
[Erase]
Start=5,Length=6
```

3.1.6 資料[Data]

SD writer 支持最多燒錄 10 個資料文件，項目名稱為 [Data0] ，[Data1] ，[Data2]… 至 [Data9] 。使用者可任意從這 10 個項目名稱中做選擇，例如，使用者需要燒錄 3 個資料文件，項目名稱可選擇 [Data0] ，[Data1] ， [Data2] 或是 [Data1] ，[Data3] ，[Data5] 。

資料的格式為: 文件名稱，位移 。例如，使用者想要燒錄 u-boot.bin 到 0x100000 以及燒錄 980uimage 到 0x200000 時，使用者必須填寫:

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

[Data1]
980uimage, 0x200000
```

3.1.7 使用者定義 [UserDefine]

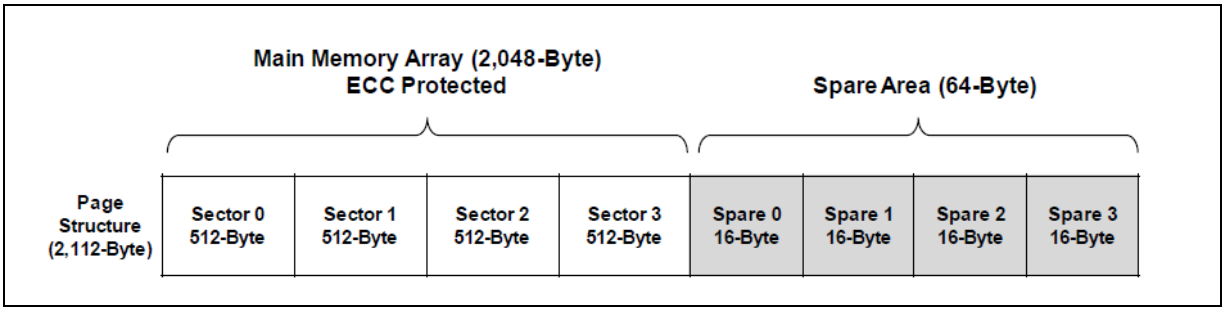
SD writer 針對 SPI NAND flash 提供使用者一個比較有彈性的方式去定義 SPI NAND flash 的 page 大小以及 spare area 大小。此外，使用者也可以定義 SPI Quad 模式讀取命令， 讀取 SPI flash 狀態寄存器命令，寫入 SPI flash 狀態寄存器命令，Quad 模式數值，以及 dummy 位元個數。使用者定義的項目名稱為 [UserDefine] 。

SPI NAND flash 設定格式為: PageSize=SPI NAND page 大小，SpareArea= SPI NAND spare area 大小。請注意位元大小是採用 10 進制。

例如，使用者定義 SPI NAND page 大小為 2048 位元，spare area 大小為 64 位元，使用者可以填入:

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

SPI NAND flash page 大小以及 spare area 大小會隨著不同的 SPI NAND flash 型號而不同，相關的資料可以從 NAND flash 規格書中獲得。以下為 SPI NAND flash 規格書中描述 page 大小以及 spare area 大小的範例。

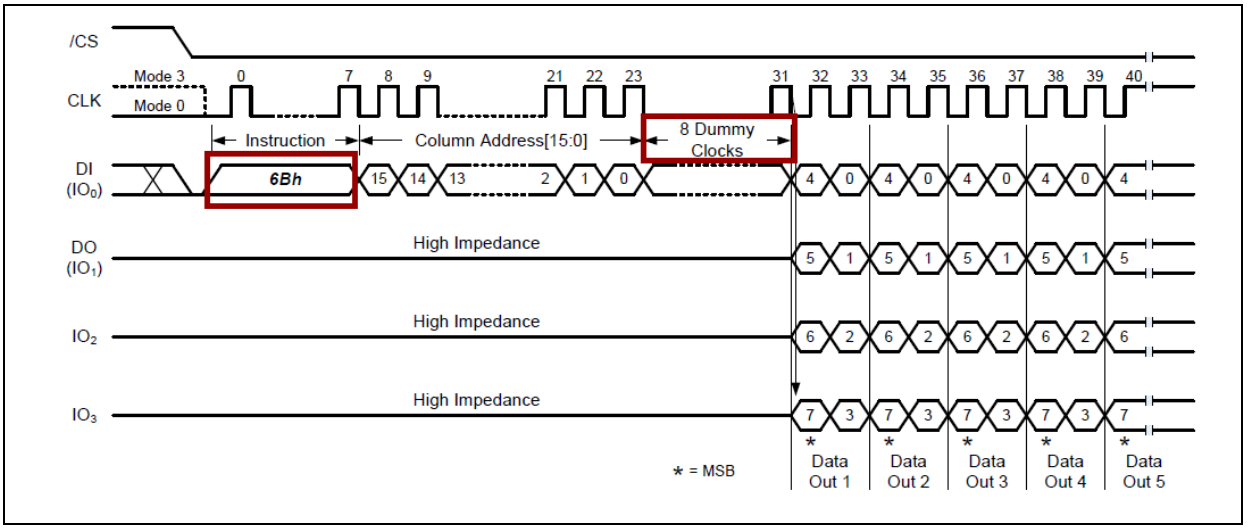


圖表 3-1 SPI NAND Flash 架構

Quad 模式格式為: QuadReadCmd=Quad 模式命令，ReadStatusCmd=讀取狀態寄存器命令，WriteStatusCmd=寫入狀態寄存器命令，StatusValue=狀態寄存器中Quad 模式狀態為所在的位置，DummyByte=quad 命令所需的 dummy 位元個數，命令數值是採用 16 進制。如下範例:

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

SPI Quad 模式讀取命令同樣也可以從規格書中取得。



圖表 3-2 SPI NAND Flash Quad Read 命令

3.2 config 文件範例

下面這個 config 文件範例可將 SPL U-Boot (u-boot-spl.bin)，Main U-Boot (u-boot.bin) ，環境變數文件(env.txt) ，及內核文件(980uimage) 寫入到 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
```

config 文件格式有一些限制要注意:

- 每一行的開頭請不要有空白
- 每一行的開頭可以用 "//" 代表注釋
- "[ ]" 中的字串不能更改

4 源代碼簡介

這個章節會簡介 SD Writer 源代碼的功能，使用者可以根據需求來修改源代碼。

4.1 開發環境

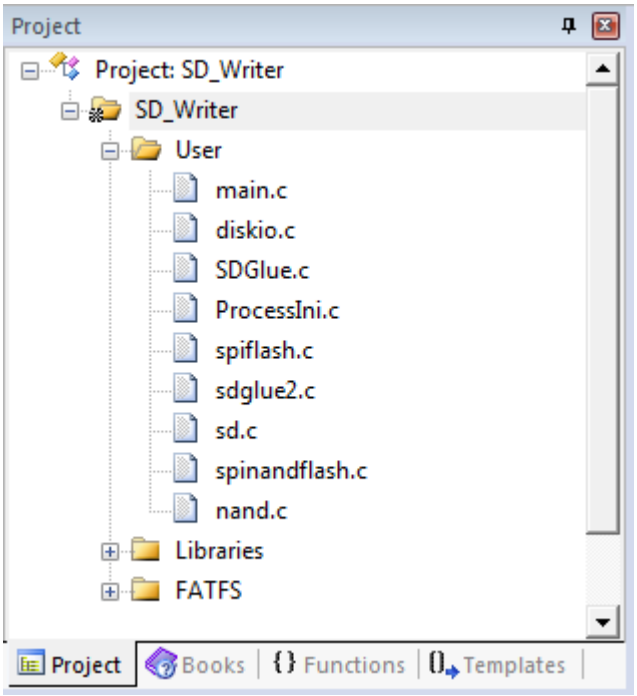
SD Writer 的開發環境為 Keil IDE。要支援 ARM9，需使用 MDK Plus 或是 Professional 版本。

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	✔	✔	✘	✘

圖表 4-1 MDK 版本選擇

4.2 源代碼文件功能

在 Keil 打開 SD Writer 專案後，可看到以下專案架構。包含以下源代碼文件。



圖表 4-2 SD Writer 計畫架構

客製化 SD Writer 可能需要更改的檔案如下:

- main.c : SD Writer 最主要的程式

- diskio.c : FatFS 底層 I/O 存取程式
- SDGlue.c 及 sdglue2.c : SD 相關程式
- ProcessIni.c : 解析 config 文件
- spiflash.c : SPI NOR flash 驅動
- sd.c : SD 驅動
- spinandflash.c : SPI NAND flash 驅動
- nand.c : NAND 驅動

5 修改歷史

Date	Revision	Description
2018.12.28	1.00	1. Initially issued.
2019.04.22	1.01	1. 编缉性修改.

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.