广东新岸线计算机系统芯片有限公司

Guangdong Nufront CSC Co., Ltd

NL6621 固件更新

设计概念

林辉

2015 年 4 月 30 日

Change Log

Date	Version	Types (New/Delete/ Modify)	Editor	Description
2015-03-26	0.01.00	New	林辉	创建文档
2015-04-25	0.02.00	ADD	林辉	添加 Chapter4 Bootloader 和 chapter 5 SDK 的详细设计
2015-05-03	0.03.00	Modify	林辉	修改 Norflash 分区地址 修改 NeDevTool 的下载和上传描述。

目录

1. 固件更新需求汇整

1.1 固件更新需求

由于 NL6621M 烧录工具软件，主要应用于研发和生产阶段，PC 端 NuDevTools 与 NL6621 设备端的软件配合，用于固件烧写，用户数据、启动参数和 RF 校准参数的读写。

注：NuDevTools 软件需要在编译时打开和关闭某些功能的编译选项，如适用对象为研发人员，功能全开；生产人员，那么只提供 Image 的烧写，用户数据，启动参数和 RF 校准参数的写功能。整理需求如下：

- 研发人员，提供所有功能；
- 只面向生产人员时，用户参数与 RF 校准参数的读写；

1.2 固件更新需求——研发阶段

在研发阶段，需要对 Norflash 空间进行读写。以保存和读取记录在其中的关键信息。例如调试参数的写入和读取、校准参数的写入和读取、系统日志信息的读取等功能。提供固件双启动机制。

1.3 固件更新需求——生产阶段

在生产阶段，需要对固件的写入、校准参数的写入、用户数据的写入等功能。只提供固件信息的写入功能，而不能读取 NorFlash 中的任何数据。

2. 固件更新总体设计

2.1 固件更新功能的总体概述

固件更新工具不仅包括烧录阶段 Image 以及用户数据等的读写，还用于日常系统的维护。在这里统一称为 NuDevTool 工具。固件更新的总体框图如下所示：

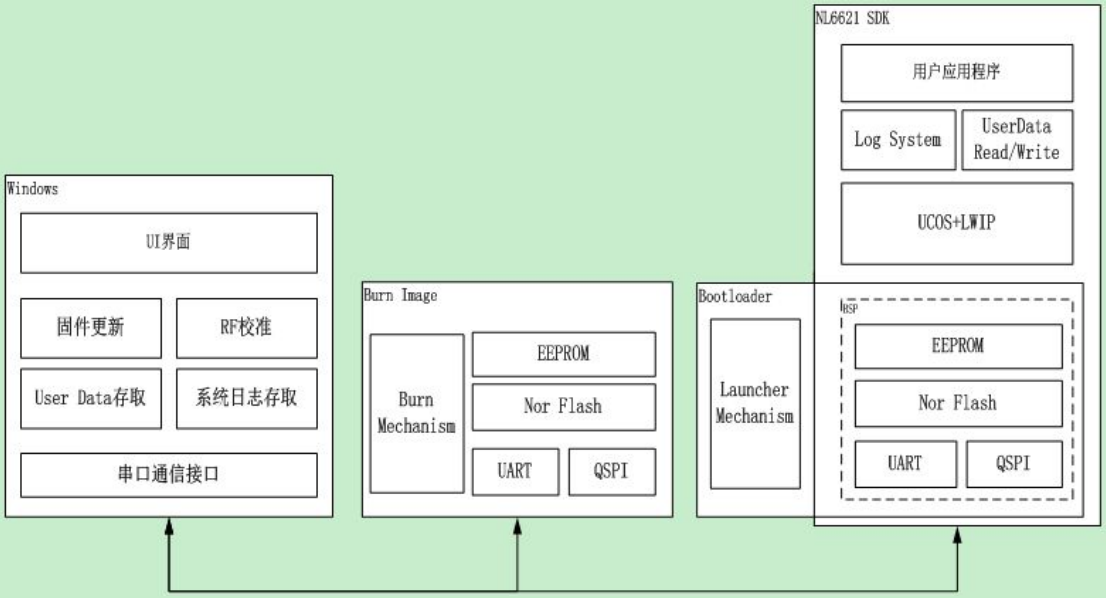


Figure 1 固件更新框图

由上图可知，固件更新框图由四大块组成：Windows 固件更新工具 NuDevTool 工具、烧录程序 BurnTool、Bootloader、NL6621 SDK。

NuDevTool 工具：PC 端上位机工具，用于 Burn Image、Bootloader、NL6621SDK 固件、RF 校准数据、用户数据以及系统关键日志的烧录或者上传。

BurnTool：烧录镜像工具，所有烧录到 NL6621 存储空间的必要工具。能够将 PC 上位机的数据烧录到 NL6621 开发板或者模组上的 NorFlash 或者 EEPROM。

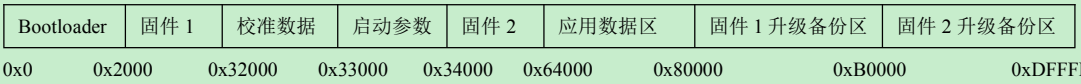
Bootloader：NL6621 的启动镜像，能够记录系统启动次数、启动不同的 SDK 镜像。

NL6621 SDK：NL6621 的 SDK 镜像。在固件更新框架内，用于记录用户参数、系统关键性日志。

2.2 NorFlash 存储规划

由于 NL6621 使用外扩 NorFlash 或者 EEPROM 存储固件和用户数据，因此固件更新的所有数据都存储于 NL6621 的片外 NorFlash 中。从固件更新的数据需求来看，一共包括 8 个数据区域，分别为：Bootloader（8KBytes），固件 1（192KBytes），RF 校准数据（4KBytes），启动参数（4KBytes），固件 2（192KBytes），应用数据区（112KBytes），固件 1 升级备份区（192KBytes），固件 2 升级备份区（192KBytes）。

Nor Flash:


Bootloader
固件 1
校准数据
启动参数
固件 2
应用数据区
固件 1 升级备份区
固件 2 升级备份区
0x0
0x2000
0x32000
0x33000
0x34000
0x64000
0x80000
0xB0000
0xDFFFF

3. 固件更新——BurnTool 详细设计

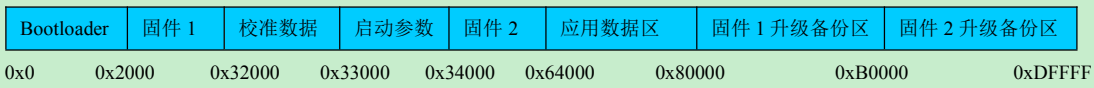
烧录镜像工具，所有烧录到 NL6621 存储空间的必要工具。能够将 PC 的串口数据烧录到 NL6621 开发板或者模组上的 NorFlash 或者 EEPROM 以及从 NorFlash 或者 EEPROM 中读取相应的数据信息传回 PC 的固件更新工具。

PC 端与 BurnTool 需要建立一套稳定的通信协议以完成固件或者相关数据的通信。分为串口数据的下载和上传。

1、BurnTool 的数据下载

BurnTool 工具最重要的功能时固件的更新，因此，需要 BurnTool 工具能够下载数据到 NL6621 上。由于 NL6621 使用的是外部 NorFlash，因此 NorFlash 的大小由用户决定，其中如下图的蓝色色块为固定的烧录地址，不可更改，超过 0xDFFFF 区域，用户可以自由的烧写数据。

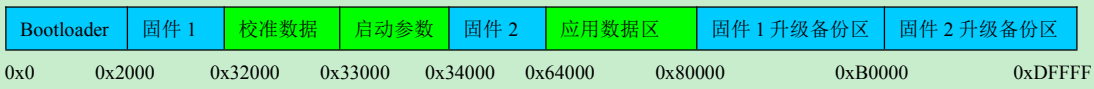
Nor Flash:



2、BurnTool 的数据上传

BurnTool 工具同时具备将 NorFlash 数据读取到 PC 端功能。例如将应用数据区、校准数据区、启动参数区中的数据读取到 PC 端进行数据分析。如下图绿色部分。

Nor Flash:



3.1 BurnTool 通信协议

BurnTool 与 PC 上位机通信流程图如下所示：

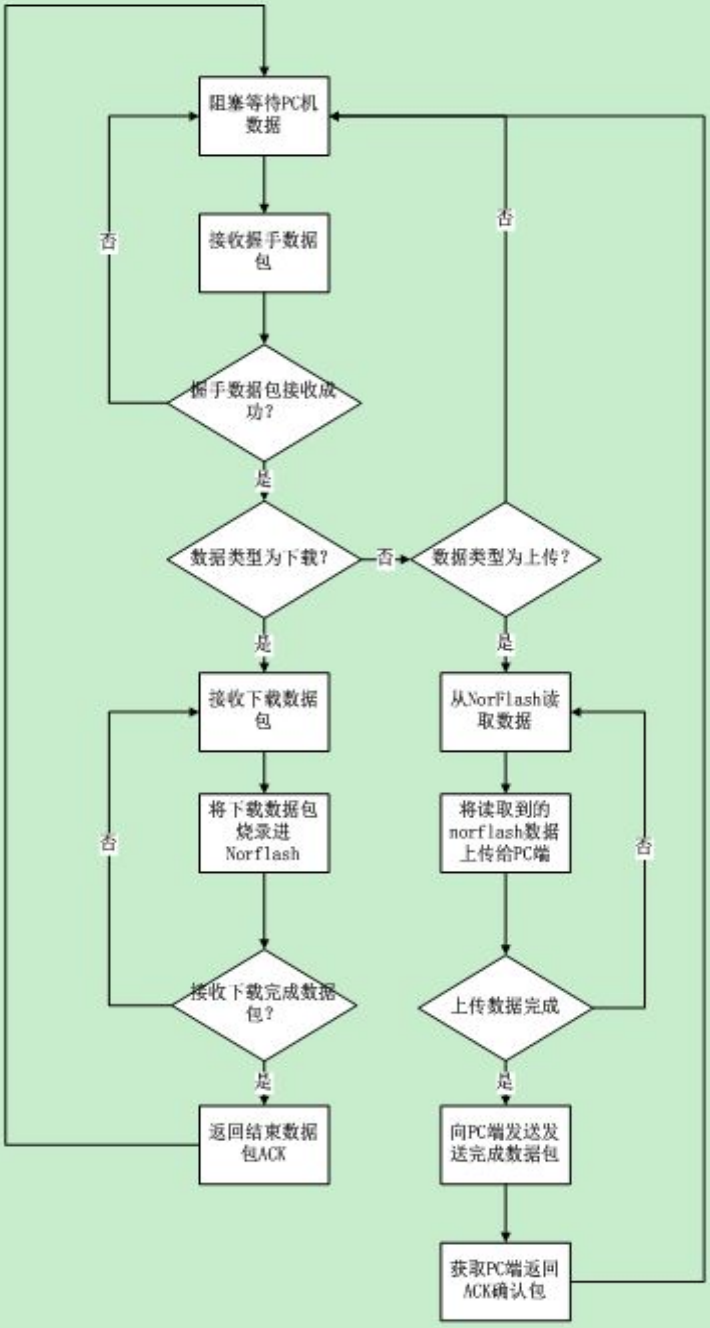


Figure 2 BurnTool 与 PC 上位机流程图

从上图可以看到，PC 上位机与设备之间的通信采用简单的应答式数据传输，为了减低烧录的时间不对传输数据进行校验。

3.1.1 通信协议数据定义

根据 PC 机与设备端的通信机制，需要定义一个通信协议栈，该协议栈的具体描述如下所示:

Type	Function	Length

Seq_num
Data

在整个通信协议数据中，定义通信协议结构体。具体定义如下所示：

```
Typedef struct {
    unsigned char type;       /*  报文信息类型  */
    union {
        unsigned char action;          /*  报文信息功能  */
        unsigned char result;          /*  报文 ACK 结果  */
    } function;
    unsigned short length;             /*  报文长度  */
    unsigned int seq_num;              /*  报文序列号  */
}__attribute__((packed)) UartHead;

typedef struct    {
    UartHead head;                  /* Uart  通信协议报文头  */
    unsigned char data[NT_DATA_MAX_NUM];     /*  存储数据报文  */
} __attribute__((packed)) UartPack;
```

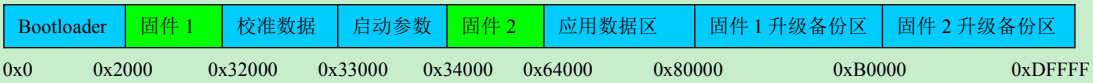
通信协议中，如果是下载固件，最初下载到设备上的数据为固件头数据。该部分已有数据不能随意更改，但可以在后面增加相应的字段（增加数据长度不能超过 204Bytes）。

```
typedef PACKED struct _FW_HDR
{
    UINT8    StartFlag[8]; // Nu_link
    UINT32 FwSize; // total firmware image len, including fw hdr and tail
    UINT32 FwVerTime; /* Unix32 bit time (seconds since Jan 1 1970.).
                                        The version format is defined in [2]. It consists:
                                        o    Major: bit 27-31
                                        o    Minor: bit 20-16
                                        o    Patch: bit 12-19
                                        o    Build: bit 0-11 */
    UINT32 AppEntryAdr;
    UINT16 CheckFlag;    //check flag, b1..0 , 0    not check 1: CRC, 2: SUM8
    UINT16 SdioBurstTransLen; // default 2K for SDIO, it must be multiple of 4 bytes
    UINT16 I2cSpeed; // 0: standard mode, 1: fast mode
    UINT16 SpimClkDiv; // SPI clk div, SPI_CLK=40MHZ/clk_div
    UINT32 UartBaudrate; // UART baudrate

    /* bootloader 升级时保持前面的不动，在后面追加字段  */

} FW_HDR;
```

4. 固件更新——Bootloader 详细设计

Bootloader 用于启动保存在 NorFlash 中的固件。Bootloader 分别需要从固件 1 或者固件 2 或者 0xDFFFF（固件备份区后的任意空间）之后的空间处加载固件到 CodeSram 运行。如下图绿色所示为需要加载的固件地址区。

Nor Flash:



Bootloader 每次启动需要充启动参数去读取相应的参数用于判断当前是从固件 1 处还是固件 2 处加载固件或者 0xDFFFF（固件备份区后的任意空间）后的地址启动代码。

4.1.1 启动参数

Bootloader 的启动依靠启动参数，启动参数的原理如下。

启动地址	启动标志	启动状态	SDK1 启动次数	SDK2 启动次数
4Bytes	2Bytes	2Bytes	4Bytes	4Bytes

启动地址：启动 SDK 的启动地址，其中 SDK1 和 SDK2 的地址固定。该值由 BurnTool 写入。

0x2000:固件 1

0x34000：固件 2

>0xDFFFF:其他测试固件

启动标志：标志启动固件的类型。该值由 BurnTool 和 SDK 写入。

0x1：启动固件 1

0x2：启动固件 2

0x3：启动地址>0xDFFFF 的固件

启动状态：标志启动的固件是否成功。该值由 Bootloader 和 SDK 写入。

0x0：没有启动固件，

0x1：启动固件 1 成功

0x2：启动固件 2 成功

0x3：启动>0xDFFFF 的固件成功

SDK1 启动次数、SDK2 启动次数：该值标示相应 SDK 的启动次数，由 SDK 成功启动后写入。每次读取然后累加 1.

注：

1、BurnTool 每次进行烧录行为时，都需要更新启动地址、启动标志和启动状态三栏。不能对 SDK1 和 SDK2 启动次数的数据进行修改。

2、Bootloader 只允许修改启动状态，每次启动时都需要将该位置为 0.

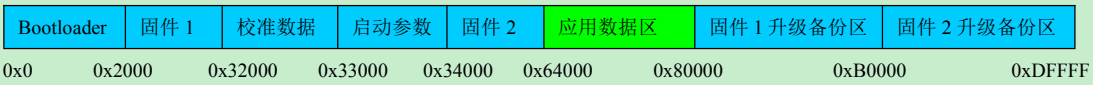
3、SDK 启动时通过检查 SDK1 和 SDK2 的启动次数，判断是否为 0xffffffff 作为是否第一次启动，然后进行初始化并进行累计工作，如果不是则直接进行累计。

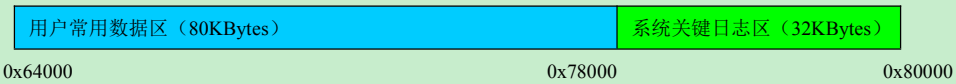
5. 固件更新——NL6621 SDK 详细设计

当产品交付给客户后，当系统出现异常等状况，需要系统将该部分的关键日志信息进行保存，而目前我们的开发板以及模块普遍采用 NorFlash 作为引导和数据保存，因此需要采用相关机制将该部分的数据信息保存到 NorFlash 中。当产品出现异常等情况时，可以通过 PC 端的工具将保存在系统 关键日志区中的数据读取出来进行分析。

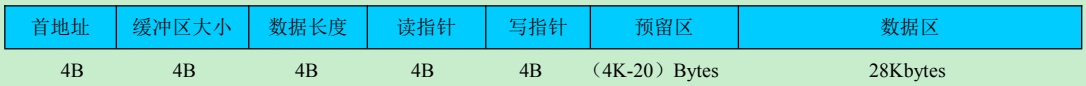
由于部分客户考虑成本等问题，NorFlash 的最小容量为 512KBytes，且 NorFlash 的所有分区已经做了规划，在不考虑改动大分区的情况下，我们能够使用的只有应用数据区。

Nor Flash:





从上图的规划来看，系统关键日志区拥有 32KBytes 的空间保存数据，为了保持关键日志的数据的读写信息，前面 4Kbytes 用于保存后面 28Kbytes 数据的读写信息状态头。而系统关键日志区的的结构框图如下所示：



首地址：数据区的起始地址，这里为 0x79000

缓冲区大小：标志数据区的大小，这里为固定值 28Kbytes

数据长度：标志数据区中存储的可用的数据信息

读指针：标志可以读取 norflash 关键日志的地址

写指针：标志数据区当前写入数据的地址

预留区：保留后续使用

数据区：具体保存关键日志的数据空间

6. 固件更新——NuDevTool 详细设计

NuDevTool 上位机工具主要是用于 Burn Image、Bootloader、SDK 固件、RF 校准数据、用户数据以及系统关键日志的烧录或者上传。

功能主要分为三大模块：

1、下载 BurnTool、Bootloader、SDK 固件、RF 校准数据、用户数据以及启动参数；

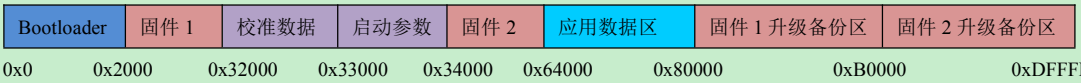
2、上传 RF 校准数据、用户数据以及启动参数；

3、输出烧录信息和上传调试信息日志。

6.1 功能详细设计

从 Nor Flash 的分区来分析，系统总共分成 8 个区，分别分成 4 种类型：Bootloader、SDK 固件、校准参数和启动参数、应用数据区。其中前面的三种类型是必须烧录的数据。如下图所示：

Nor Flash:



从上图可知，Bootloader、校准数据、启动参数和固件完成一个正常的固件加载的所有功能。因此 NuDevTool 工具需要提供者 4 中类型的数据能够正确的烧录进 NorFlash。其中与 BurnTool 的通信协议参考 BurnTool 详细设计部分。

6.1.1 NuDevTool 下载功能

NuDevTool 每次烧录或者更新固件，都需要通过串口烧录 BurnTool 工具，因此第一个通过串口写入设备的固件为 BurnTool.bin。这部分在 NuDevTool 设为默认功能。

NuDevTool 可以选择 Bootloader、固件 1、固件 2、其他固件进行烧录：

1、 Bootloader 烧录，通过 Bootloader 的 bin 文件进行烧录

2、 所有的 SDK 固件都通过 bin 文件进行烧录

3、 用户启动参数通过通过界面的文本框进行烧录

4、 RF 校准参数通过外部文件进行烧录

其中其他固件的烧录需要用户填写启动参数区的启动地址（启动地址必须大于 DFFFF）和启动标志（启动第三个固件，启动标志设为 0x03）

6.1.2 NuDevTool 上传功能

NuDevTool 工具的上传功能，主要包括 RF 校准参数、启动参数、用户数据区这三部分。其中数据的读取，以保存为文件和在界面显示两部分：

1、 将读取到的 RF 校准数据保存为文件；

2、 启动参数直接显示在界面；

3、 用户数据区的前半部分（80KBytes）保存为文件
4、 用户数据区的后半部分（32KBytes）直接显示在界面（显示在 scoller 中）。

7. 附录

本文的参考资料请参考下表：

表格 1 参考资料

名称	日期	出处
[1]		
[2]		
[3]		
[4]		
[5]		
[6]		
[7]		

8. 术语

本文使用的术语请参考下表：

表格 2 术语

名称	描述

。