

# **MaxWit g-bios 开发者手册**

## **第 1 卷：使用入门**

MaxWit 开放实验室

<http://maxwit.googlecode.com>

2008 年 7 月 1 日

# 目 录

1	初识MaxWit g-bios .....	5
1.1	开源项目g-bios概述.....	5
1.2	获取最新g-bios源码.....	6
1.3	如何参与g-bios开发.....	6
2	Host端发行版的选择及软件安装.....	7
2.1	关于Linux发行版 .....	7
2.2	安装系统必备软件包.....	7
2.3	配置kermit .....	9
2.4	配置tftp Server.....	9
3	编译g-bios.....	10
3.1	制作交叉编译Toolchain.....	10
3.2	g-bios配置.....	11
3.3	g-bios编译.....	13
4	烧录g-bios.....	14
4.1	烧录g-bios上半部分（Top-half） .....	14
4.2	烧录g-bios下半部分（Bottom-half） .....	14
4.3	NB设计思想 .....	17
5	使用g-bios引导Linux系统 .....	18
5.1	g-bios引导命令（boot）设计特点.....	18
5.2	TFTP + NFS方式启动Linux .....	18
5.3	FLASH + NFS方式启动Linux.....	18
5.4	FLASH + FLASH方式启动Linux.....	19
6	g-bios命令详解.....	20
6.1	g-bios命令一览.....	20
6.2	flash命令 .....	20
6.3	网络命令.....	20
6.4	boot命令.....	23
6.5	其他命令.....	23





# 1 初识MaxWit g-bios

## 1.1 开源项目g-bios概述

MaxWit 开放实验室（MaxWit Open Lab）是由多家公司资助成立的，致力于研发开源项目和探讨软件开发技术的公益性组织。2008 年 1 月正式成立于上海浦东张江高科，目前开放实验室成员主要来源于 Google、Intel、TI、AMD、华为、Cisco、飞利浦等公司资深研发人员以及清华、浙大、上交大、中科院等科研院校的师生。

MaxWit g-bios（以下简称 g-bios）是由 MaxWit 开放实验室和开源社区共同研发的一个 Bootloader，或者说是一个嵌入式系统的 BIOS，相当于 PC 机的 BIOS + Bootloader。g-bios 不但借鉴了几乎所有主流 BSP/BIOS/Bootloader 的优点，而且加入不少独创的特性，包括：

- 1) 自动检测有待烧录的 image 文件类型，并智能自动烧录。
- 2) 支持多种文件系统，包括 YAFFS2、JFFS2、CRAMFS、NFS 等。
- 3) 支持两种用户界面：GUI（类似传统 PC BIOS）和命令行模式（面向嵌入式系统）。
- 4) 命令行自动补全（Tab 键）及历史记录（上、下键）支持。
- 5) Flash(MTD)分区支持，帮助 Linux、Android 内核识别分区。
- 6) 自动设置 Linux 内核启动参数（Linux kernel command line），极大地降低了参数设置的复杂度并减少了启动出错的概率。当然，同时也支持手动设置，以满足特殊要求。
- 7) 常用命令具有记忆功能。如 boot 命令，它能记住用户输入的参数，以后只需简单输入 boot 即可。
- 8) 引入全新的架构及 NB（Never Burn Down，烧不死）技术。核心设计思想是：把 g-bios 分为上半部分和下半部分，上半部分以最小的代码量完成 CPU 和 Memory 的初始化，并实现引导下半部分的功能；下半部分为 g-bios 主体。上半部分设计简单，调试周期短，完成后就固化在特定的引导区中不再更改；开发人员可在没有仿真器的情况下大胆开发下半部分代码（即 g-bios 主体），事实上，只需一根串口数据线应能轻松完成整个 g-bios 的开发。启动代码的地址无关性带来的麻烦？没有了！因为 bug 或不小改错了代码，甚至是数据线连接问题而导致启动黑屏？也不可能出现了！☺在调试完成并正试发布的产品时，若有必要，也可将上下两部分可合成一个整体——只需一个命令重新编译即可。
- 9) 优秀的子系统设计，包括：中断、网络、Flash、USB 子系统，等等。
- 10) 集成类似 PC 机版本的 Video BIOS。
- 11) 支持基于龙芯的 PC 机及嵌入式系统。
- 12) 完美支持 Google Android 操作系统，简化 Android 的系统移植过程。
- 13) 支持图形化配置，不但让新手很容易上手，而且使 g-bios 的移植和开发过程变得更简单。

更多详情，请登录项目主页 <http://maxwit.googlecode.com> 或 ChinaUnix 论坛（<http://linux.chinaunix.net/bbs>）上的 g-bios 版块。

## 1.2 获取最新g-bios源码

请确认 `subversion`（一个版本管理软件）已经安装，然后执行如下命令：

```
# svn co http://maxwit.googlecode.com/svn/trunk/ maxwit-read-only
```

当前目录（方便描述起见，假定为 `HOME` 目录）下将会创建一个名为“`maxwit-read-only`”的子目录，该子目录包含了 MaxWit 开放实验室的两个开源项目——MaxWit Linux 和 g-bios，以及 GNU 交叉编译工具。

## 1.3 如何参与g-bios开发

g-bios 开源社区采用 `maillist` 和 `bbs` 相结合的方式，任何人都可以通过这两种方式把自己的代码递交给 g-bios 项目维护者。若对文档有任何疑问或改进也可联系我们。

g-bios 论坛	<a href="http://linux.chinaunix.net/bbs/forum-70-1.htm">http://linux.chinaunix.net/bbs/forum-70-1.htm</a>
g-bios 邮件列表	<a href="mailto:maxwit@googlegroups.com">maxwit@googlegroups.com</a>
g-bios 项目维护者	Conke Hu < <a href="mailto:conke.hu@gmail.com">conke.hu@gmail.com</a> >
	Tiger Yu < <a href="mailto:tigerflying.yu@gmail.com">tigerflying.yu@gmail.com</a> >
	Fleya Hou < <a href="mailto:fleya.hou@gmail.com">fleya.hou@gmail.com</a> >
文档编辑	

## 2 Host端发行版的选择及软件安装

### 2.1 关于Linux发行版

目前已测试通过的发行版有（包括 64 位版）：Debian 5.0、Ubuntu 9.04、Ubuntu 8.10、Fedora Core 10，推荐使用 Debian 5.0。若有人有兴趣测试并支持其他 Linux 发行版版，欢迎把 patch 发给 g-bios 项目的维护者。

### 2.2 安装系统必备软件包

安装以下软件包列表：

```
gcc g++ make ckermit tftpd-hpa tftp-hpa subversion git-core patch
```

```
# apt-get install gcc g++ make ckermit tftpd-hpa tftp-hpa subversion git-core patch
```





## 2.3 配置kermit

第一步，下载 kermit 配置文件

```
# wget http://maxwit.googlecode.com/files/kermrc
# cp -v kermrc ~/.kermrc
```

第二步，编辑 kermrc

打开 ~/.kermrc，修改“set line”一行，确认你所用的串口设备，若用的是 USB-to-Serial 转接器，可以改成：“set line /dev/ttyUSB0”

## 2.4 配置tftp Server

第一步，更改 tftpd 下载目录

tftp 服务器的默认下载目录是 /var/lib/tftpboot，我们要改为 \${HOME}/maxwit/images。打开 /etc/inetd.conf，找到以“tftpd”开头的一行，将其中的 /var/lib/tftpboot 改为 \${HOME}/maxwit/images：

```
tftp dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s ${HOME}/maxwit/images
```

第二步，/etc/init.d/tftpd-hpa restart

第三步，测试 tftp server

```
# cd /tmp

# echo hello > ~/maxwit/images/test
# chmod 666 ~/maxwit/images/test
# tftp 192.168.0.111 (假定本机 IP 为 192.168.0.111)
> get test
> quit
# cat test
# rm test ~/images/test
```

## 3 编译g-bios

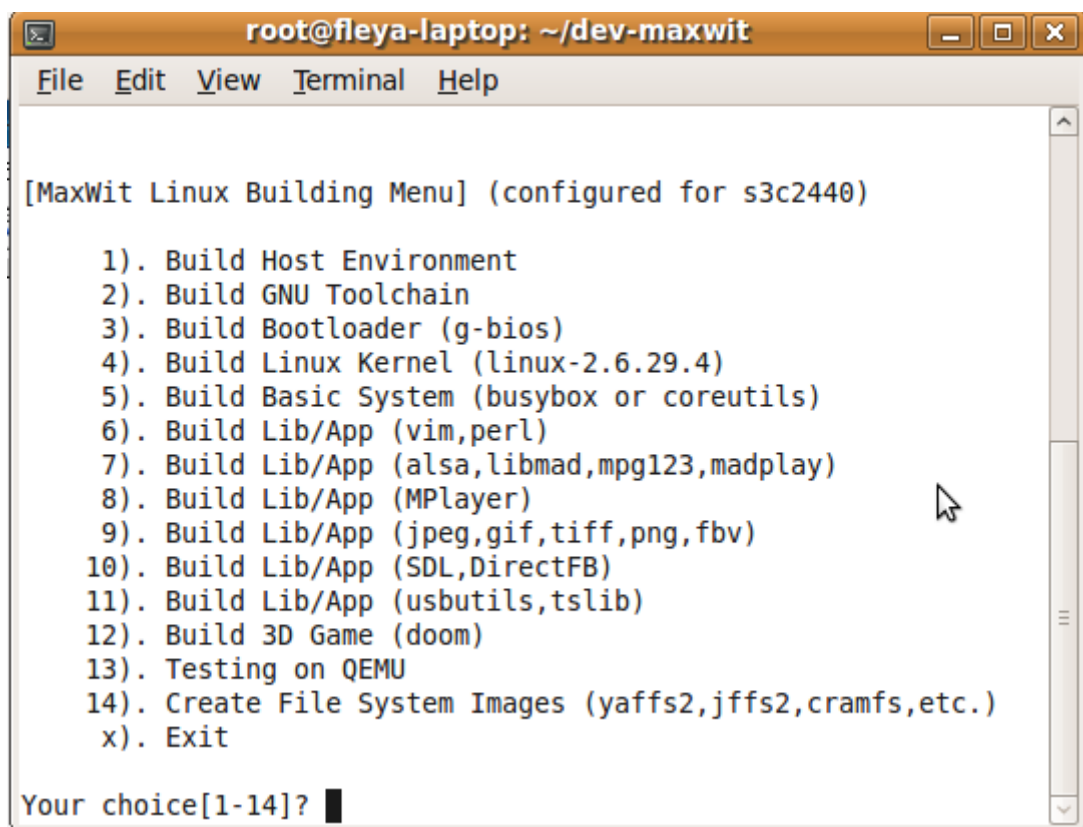
### 3.1 制作交叉编译Toolchain

若已有合适的交叉编译工具（如 ARM 平台需要支持 EABI 的交叉编译工具），则可跳过这一节。当然，也可用 MaxWit 源码包来重新 build 一个：

第一步，cd ~/maxwit-read-only，运行 build.：

```
# ./build
```

下图是 Maxwit Linux 的编译菜单（选项 2 是编译一个 arm 的交叉编译工具链）：



选择“2”，回车。

这个过程比较漫长，不过在推荐的系统上一定能过，因为已测过不知多少次了☺

第二步，toolchain 成功编译之后，查看生成 toolchain 的相关信息。

```
# arm-linux-gcc -v
```

Configured with: /root/maxwit/build/gcc-4.4.0/configure

--prefix=/usr --build=i486-linux-gnu

--host=i486-linux-gnu

--target=arm-maxwit-linux-gnueabi

--with-sysroot=/root/maxwit/sysroot

--with-gmp=/root/maxwit/tools

--with-mpfr=/root/maxwit/tools

--disable-multilib

--disable-nls

--enable-shared

--enable-\_\_cxa\_atexit

--enable-c99

--enable-long-long

--enable-threads=posix

--enable-languages=c,c++

--with-float=soft

--with-cpu=arm920t

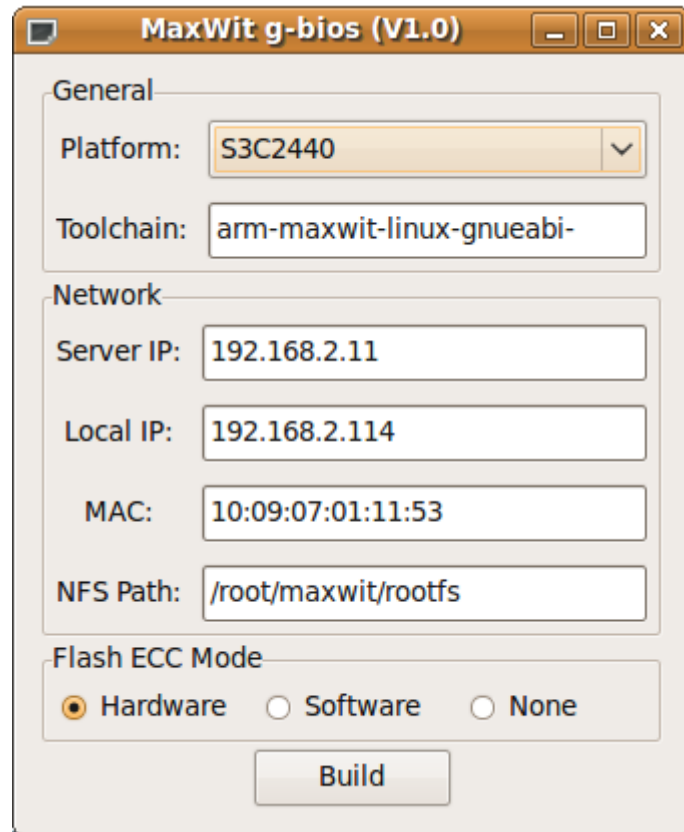
Thread model: posix

gcc version 4.4.0 from <http://maxwit.googlecode.com>

## 3.2 g-bios配置

```
# cd ~/maxwit-read-only/g-bios
```

```
# ./configure
```



其中 Platform 可以是 at91sam9263、at91sam9261、s3c2410 或 s3c2440 等。  
Toolchain 默认使用的是 MaxWit 源码包编译生成的 toolchain，也可以手工修改为系统上已有的 toolchain（注：Toolchain 要支持 EABI）。

g-bios 配置程序所完成的功能：

- 1 general
  - CPU
- 2 Flash
  - ECC mode
  - Partition Table
- 3 Networking
  - MMIO & IRQ
  - Local IP & Server IP & Netmask
  - MAC addr
  - NFS root path
- 4 Bootup Logo
  - banner
  - jpeg source
- 5 Linux & Android
  - command line

### 3.3 g-bios编译

# make

会生成 g-bios-th.bin 和 g-bios-bh.bin 并自动 copy 到\${HOME}/maxwit/images 目录下。

## 4 烧录g-bios

### 4.1 烧录g-bios上半部分（Top-half）

g-bios 上半部分的烧录方法与其他 bootloader 一样，都依赖于具体的板子，所以请大家参考板子的手册烧录上半部分（g-bios-th.bin 文件）。

g-bios 上半部主要是 Load 下半部，为下半部服务的。其功能如下：

1. 从串口 load 下半部并运行
2. 从 Flash 上 Load 下半部并运行
3. 自动检测 Flash 上的下半部是否存在，若存在则默认从 Flash 上 Load 下半部并运行，否则等待从串口 Load

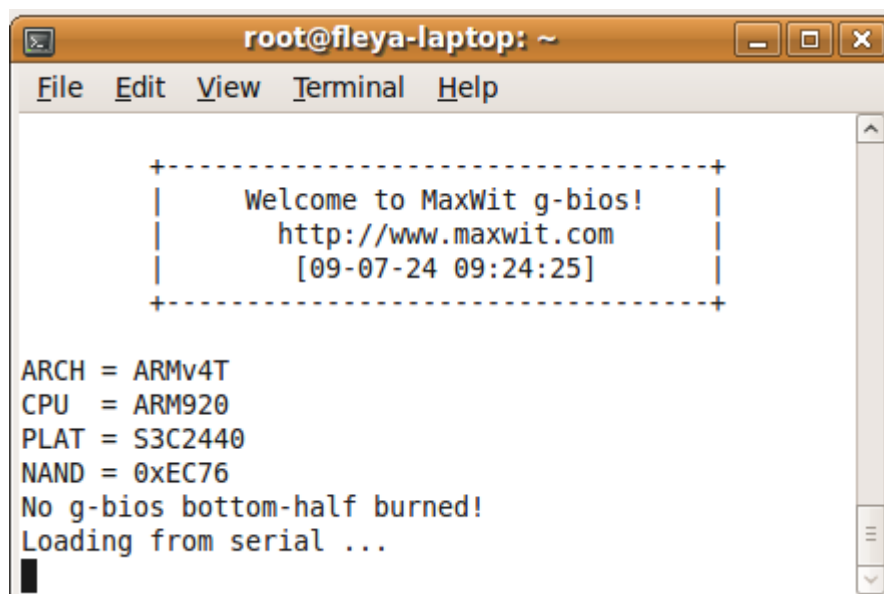
### 4.2 烧录g-bios下半部分（Bottom-half）

g-bios 下半部的烧录原理：利用上半部分的串口 Load 功能将下半部分 Load 到 SDRAM 并运行，再利用下半部的烧录功能将下半部的 image 文件烧录到 Flash 的特定位置上。

第一步，先启动上半部，使用串口线将开发板上的 COM1 口和 PC 机的 COM 口连接、并用网线连接开发板和 PC 机，在 Host 端打开 kermi

```
# cd ~/maxwit/images
# kermi
C-kermi>c （回车）
```

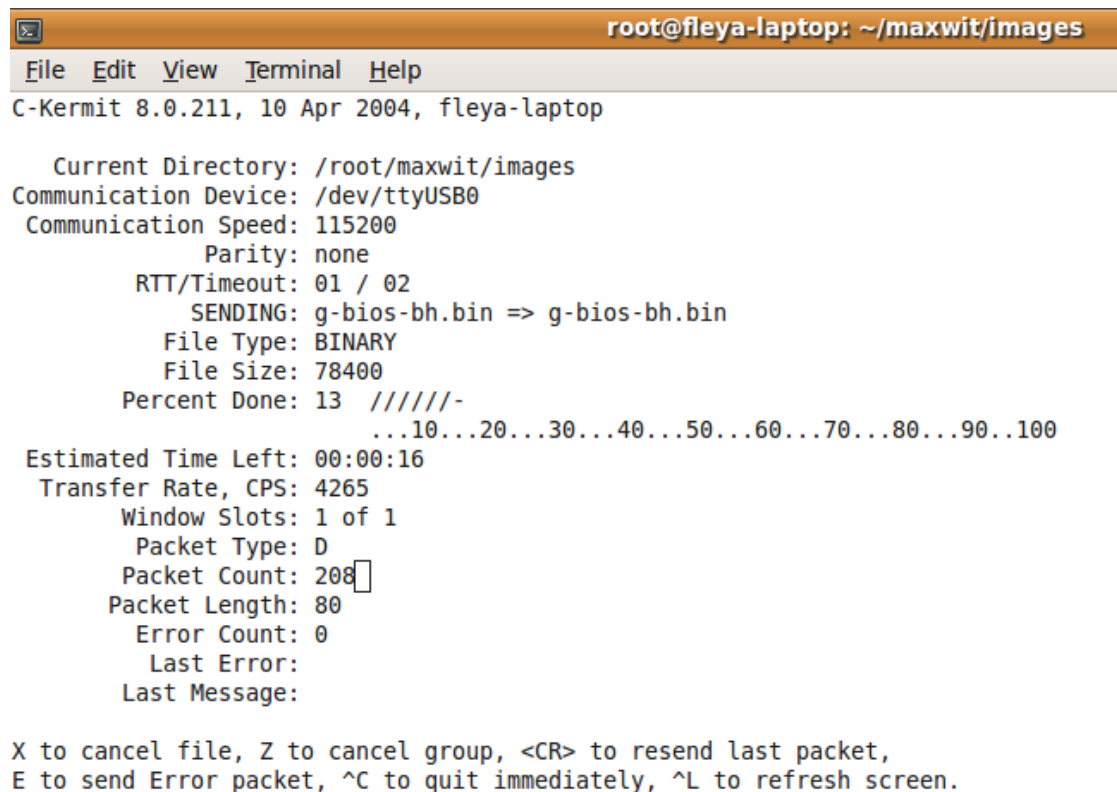
第二步，再按下开发板 Reset 键，将会进入 g-bios 上半部的启动界面（如图）



注：g-bios 的上半部会自动检测 Flash 上是否已烧录下半部 g-bios-bh.bin，若下半部已烧录则直接从 Flash 上将下半部 Load 到 Sdram 并运行，若未烧录则如上图所示，提示下半部未烧录，并需要通过从串口 Load 下半部并启动。下半部支持通过网络和串口两种方式烧录指定文件到 Flash 中。也在上半部启动过程中按任意键启动串口 Load 的功能

第三步，选择“2”回车，然后同时按下“CTRL”和“\”键，再按下“c”

C-Kermit> send g-bios-bh.bin



```
root@fleya-laptop: ~/maxwit/images
File Edit View Terminal Help
C-Kermit 8.0.211, 10 Apr 2004, fleya-laptop

Current Directory: /root/maxwit/images
Communication Device: /dev/ttyUSB0
Communication Speed: 115200
Parity: none
RTT/Timeout: 01 / 02
SENDING: g-bios-bh.bin => g-bios-bh.bin
File Type: BINARY
File Size: 78400
Percent Done: 13  //////-
                  ...10...20...30...40...50...60...70...80...90...100
Estimated Time Left: 00:00:16
Transfer Rate, CPS: 4265
Window Slots: 1 of 1
Packet Type: D
Packet Count: 208
Packet Length: 80
Error Count: 0
Last Error:
Last Message:

X to cancel file, Z to cancel group, <CR> to resend last packet,
E to send Error packet, ^C to quit immediately, ^L to refresh screen.
```

C-Kermit> c

进入 g-bios 下半部的启动界面，按任意键进入 g-bios 的命令行，否则 g-bios 将会自动 load kernel 并启动（如图）

```
root@fleya-laptop: ~/maxwit/images
File Edit View Terminal Help
9. [S3C2440NandProbe() @ 0x33012980]
S3C2440NandProbe():
NAND-0 is detected: Vendor ID = 0xec, Device ID = 0x76 (Samsung K9F1208)
Flash details:
    Chip size = 0x04000000 (64M)
    Block size = 0x00004000 (16K)
    Page size = 0x00000200 (512)
    OOB size = 0x00000010 (16)
Scanning bad blocks:
done!
Total: 1 nand chip detected
OK!

10. [SysConfInit() @ 0x33012ed4]
Init application: System Configuration.

Partition List: (on K9F1208)
-----
Index      Start          End          Size      Type      Name
-----
0*  0x00000000 - 0x00080000  512K      g-bios    "g-bios"
1   0x00080000 - 0x00300000  2M512K    linux     "linux"
2   0x00300000 - 0x01b00000  24M       yaffs2    "rootfs"
3   0x01b00000 - 0x04000000  37M       yaffs2    "data"
-----
OK!

Autoboot after 3 seconds. Press any key to interrupt.
g-bios: 0#
```

g-bios: 0# partshow  
Partition List: (on K9F1208)

Index	Start	End	Size	Type	Name
0*	0x00000000 - 0x00080000		512K	g-bios	"g-bios"
1	0x00080000 - 0x00300000		2M512K	linux	"linux"
2	0x00300000 - 0x01b00000		24M	yaffs2	"rootfs"
3	0x01b00000 - 0x04000000		37M	yaffs2	"data"

共有 4 个默认分区：

- 0 分区是 g-bios 分区
- 1 分区是 Linux 分区
- 2 分区是 rootfs 分区，分区类型支持 yaffs、yaffs2、jffs、jffs2、cramfs
- 3 分区是 data 分区

g-bios: 0# ifconfig （查看 IP）

local addr: 192.168.2.111 （开发板的 ip 地址）

server addr: 192.168.2.11 （主机的 ip 地址）

MAC addr: 10:09:06:16:22:03 （开发板的 MAC 地址）



g-bios: 0# ping 192.168.2.11 (测试开发板和 PC 机网络是否连通)

64 bytes from 192.168.2.11: icmp\_seq=1 ttl=64 time=12.7 ms

64 bytes from 192.168.2.11: icmp\_seq=2 ttl=64 time=12.7 ms

64 bytes from 192.168.2.11: icmp\_seq=3 ttl=64 time=12.7 ms

g-bios: 0# cd 0 (进入 g-bios 分区)

g-bios: 0# tftp g-bios-bh.bin

(这个命令会自动通过网络到 tftp 服务器下载 g-bios-bh.bin, 并烧录到当前的 Flash 分区中, 以后就可以直接从 Flash 中启动整个 g-bios 了。)

再按下开发板 Reset 键, 启动整个 g-bios, 按任意键进入 g-bios 命令行。

## 4.3 NB设计思想

核心设计思想是: 把 g-bios 分为上半部分和下半部分, 上半部分以最小的代码量完成 CPU 和 Memory 的初始化, 并实现引导下半部分的功能; 下半部分为 g-bios 主体。上半部分设计简单, 调试周期短, 完成后就固化在特定的引导区中不再更改; 开发人员可在没有仿真器的情况下大胆开发下半部分代码(即 g-bios 主体), 事实上, 只需一根串口数据线应能轻松完成整个 g-bios 的开发。启动代码的地址无关性带来的麻烦? 没有了! 因为 bug 或不小心改错了代码, 甚至是数据线连接问题而导致启动黑屏? 也不可能出现了! ☺在调试完成并正试发布的产品时, 若有必要, 也可将上下两部分可合成一个整体——只需一个命令重新编译即可。

## 5 使用g-bios引导Linux系统

### 5.1 g-bios引导命令（boot）设计特点

(coming soon☺)

### 5.2 TFTP + NFS方式启动Linux

其中 NFS 服务配置和编译 linux kernel 部分详情请参阅《MaxWit Lablin 开发者手册》  
第一卷

在 g-bios 命令行下，输入：

```
g-bios: 0# boot -t zImage -n 192.168.2:/root/maxwit/rootfs
```

#### 【说明】

- -t [filename]: 用 tftp 方式下载指定的 kernel image
- -n [nfs\_server:/nfs/path/]: 用 NFS 方式 mount rootfs。也可以加上参数，如：“-n 192.168.0.111:/path/to/nfs”。
- boot 程序具有记录功能，即，能记住用户输入的参数，换句话说，再次输入 boot 时不再需要输入参数了，除非你想重设参数☺

### 5.3 FLASH + NFS方式启动Linux

```
g-bios: 0# cd 1 (进入 Linux 分区)
```

```
g-bios: 1# ls (显示分区信息)
```

```
Partition Type = "linux"
MTD Device      = /dev/mtdblock1
Partition Base = 0x00080000 (512K)
Partition Size = 0x00280000 (2M512K)
Host Device     = K9F1208
Image File      = zImage.29 (1534080 bytes)
```

```
g-bios: 1# tftp zImage (下载 zImage 到当前分区)
```

```
192.168.2.11 => zImage: 01M464K
```

```
Total 1534080 (0x00176880) bytes received.
```

```
g-bios: 1# boot -t -n 192.168.2.11:/root/maxwit/rootfs
```

【说明】

- -t 不加参数，从 Linux 分区 Load kernel image
- -n [nfs\_server:/nfs/path/]: 用 NFS 方式 mount rootfs。也可以加上参数，如：“-n 192.168.0.111:/path/to/nfs”。

## 5.4 FLASH + FLASH方式启动Linux

g-bios: 0# cd 1 (进入 Linux 分区)

g-bios: 1# ls (显示分区信息)

Partition Type = "linux"

MTD Deivce = /dev/mtdblock1

Partition Base = 0x00080000 (512K)

Partition Size = 0x00280000 (2M512K)

Host Device = K9F1208

Image File = zImage.29 (1534080 bytes)

g-bios: 1# tftp zImage (下载 zImage 到当前分区)

192.168.2.11 => zImage: 01M464K

Total 1534080 (0x00176880) bytes received.

g-bios: 1# cd 2 (进入 Rootfs 分区)

g-bios: 2# tftp rootfs.img (下载 zImage 到当前分区)

g-bios: 2# boot -t -f 2

【说明】

- -t : 不加参数，从 Linux 分区 Load kernel image
- -f [N]: 指定 rootfs 的分区，N 为分区号

# 6 g-bios命令详解

## 6.1 g-bios命令一览

## 6.2 flash命令

flash 命令列表

命令名称	命令说明
partshow	查看 Flash 分区信息
flashdump	查看 flash 页内容，以 page 为单位，包括 oob
flasherase	以 block 为单位，擦除 flash 一块内容

命令名称：**flashdump**

参数介绍：

-b	读取指定 flash 块号，查看块内第一页内容
-p	读取指定 flash 页号，查看一页内容
-g	读取指定 flash 地址，查看从此地址起一页内容。

命令名称：**flasherase**

参数介绍：

-a	指定擦除的起始地址(以字节为单位)
-b	指定擦除的开始块号(以块为单位)
-d	擦除时忽略坏块
-m	擦除的同时写入 cleanmark
-l	指定擦除长度

命令名称：**parterase**

参数介绍：

-p	指定擦除分区的区号
-d	擦除时忽略坏块

## 6.3 网络命令

网络命令列表

命令名称	命令说明
------	------

ping	用来测试板子和主机是否连通
ifconfig	网络基本配置
tftp	配置 tftp 服务



命令名称: **ping**

参数介绍: ☺

命令名称: **ifconfig**

参数介绍:

-l	配置本地 IP
-s	配置服务器 IP
-m	配置 MAC 地址

命令名称: **tftp**

参数介绍:

-f	指定下载的文件名
-s	设定服务端 IP
-m	下载的内容放在内存里, 即, 不烧录到 flash 上

## 6.4 boot命令

命令名称	命令说明
boot	引导操作系统

命令名称: **boot**

参数介绍:

-t [filename]	若指定 filename, 则通过 tftp 下载 kernel image 文件; 否则从本地的 linux 分区下载 kernel image 文件
-r [filename]	使用 ramdisk 启动。若指定 filename, 则通过 tftp 下载 ramdisk image; 若不指定, 则从本地的 ramdisk 分区下载 image。
-f [N]	指定 rootfs 分区, N 为分区号
-n [ip:path]	用 nfs 方式 mount rootfs
-v	仅显示 kernel 启动参数, 但并不真正引导 OS

## 6.5 其他命令





命令名称	命令说明
confreset	恢复 g-bios 默认设置
Kermit	通过串口将指定的文件烧录到当前 Flash 分区
ls	查看当前分区信息
cd	切换分区

命令名称：**confreset**

参数介绍：无

命令名称：**kermit**

参数介绍：无

命令名称：**ls**

参数介绍：无

命令名称：**cd**

参数介绍：

[N]	目标分区号
-----	-------