

g-bios 开发者手册（第 1 卷，使用入门）

MaxWit 魔鬼训练营

April 26, 2011

Contents

1	初识 g-bios	1
1.1	MaxWit g-bios 简介	1
1.2	获取源码	2
1.3	g-bios 体系结构	2
1.4	源码目录结构	2
2	g-bios 的编译及烧录	5
2.1	g-bios 配置	5
2.1.1	基于命令行的配置方式	5
2.1.2	基于图形界面的配置方式	5
2.1.3	配置选项详解	6
2.2	编译	6
2.3	烧录	7
2.3.1	Burning Top-Half	7
2.3.2	烧录 g-bios BH (下半部分)	7
2.3.3	串口	13
2.3.4	SD 卡	13
2.3.5	网络	13
3	g-bios 命令详解	14
3.1	flash 读写	14
3.1.1	flash	14
3.1.2	part	14
3.1.3	ls	15
3.1.4	cd	15
3.2	MMC/SD 卡操作	15
3.2.1	mmc	15
3.3	网络连接	15
3.3.1	ifconfig	15
3.3.2	ping	16
3.3.3	tftp	16
3.3.4	dhclient	16

3.4	串口协议及工具	16
3.4.1	kermit	16
3.4.2	ymodem	16
3.5	Graphics 和 Display	17
3.5.1	lcd	17
3.6	memory 数据查看及指令跳转	17
3.6.1	memory read 和 write	17
3.6.2	memory set	17
3.6.3	任意地址跳转	17
3.7	其他命令	17
3.7.1	help	17
3.7.2	led	17
3.7.3	sysconf	18
4	引导操作系统	19
4.1	OS 引导	19
4.2	TFTP + NFS	19
4.3	FLASH + NFS	20
4.4	Booting from Flash	20
5	附录	22
5.1	g-bios commands overview	22
5.2	S3C24XX 系列平台烧录方法	22
5.3	AT91SAM 系列平台烧录方法	22
5.4	S3C64XX 系列平台烧录方法	22
5.5	OMAP3 系列平台烧录方法	22

Chapter 1

初识 g-bios

1.1 MaxWit g-bios 简介

g-bios（以下简称 g-bios）是由 Intel、IBM、Qualcomm、AMD 的几名资深软件工程师与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、UBI、NFS 等。
3. 命令行自动补全（Tab）键及历史记录（上、下键）支持。
4. Flash(MTD) 分区支持，帮助 Linux、Android 内核识别分区。
5. 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。
6. 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。
7. 引入全新的架构及 NB 技术（即 Never Burn-down，又称“烧不死”技术）。开发人员可在没有仿真器的情况下大胆开发 Bootloader。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！
8. 支持完整的中断机制。开发者可简单地通过一个编译选项选择 IRQ 或 Polling 两种模式中的任何一种。
9. 优秀的网络子系统，并提供符合 POSIX 规范的 Socket API，方便二次开发。
10. 支持多种常用外设，包括：WDT、UART、NAND、NOR、SD/MMC、USB、LCD、Touchscreen, ...
11. 集成硬件调试/测试程序，大大提高了 bring-up 的工作效率。
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 论坛上的 g-bios 版块（<http://bbs.chinaunix.net/forum-238-1.html>）。

1.2 获取源码

请确认 git (一个版本管理软件) 已经安装, 然后执行如下命令:

```
$ git clone git://github.com/maxwit/g-bios.git
```

此时会在当前目录 (方便描述起见, 假定为 HOME 目录) 下将会创建一个名为 “g-bios” 的目录, 该目录中为 g-bios 源码。

1.3 g-bios 体系结构

1.4 源码目录结构

```
$ ls /g-bios/
bh  build  doc  include  Makefile  th
```

g-bios {

- th: stage 1 代码
- bh: stage 2 代码, 即 g-bios 上半部分启动代码
- Makefile: *g-bios Makefile*
- include: 头文件 (stdio.h, string.h, g-bios.h 等)
- doc: 使用开发手册
- build: 与编译相关文件夹 (配置文件, 编译规则)

bh 目录:

```
$ ls /g-bios/bh
app  arm  base  driver  filesys  lib  Makefile  mm
```

bh: stage 2 {

- app: g-bios 应用程序、命令所在目录 (ping, ls, reboot 等)
- arm: arm 体系结构相关代码
- base: g-bios 下半部分公共文件夹 (main, shell, sysconfig 等)
- driver: 驱动程序源码, 通用 driver
- fileys: 文件系统相关源码
- lib: 库文件 (stdlib, string, net 等)
- mm: 内存管理文件夹 (内存分配函数代码等)

app 目录

```
$ ls /g-bios/app
boot  flash  led  Makefile  memory  misc  net  serial  shell  sysconf
```

将不同功能的 application 分成类, 置于不同的文件夹下。每个文件下都有一个 Makefile 文件。从文件夹的命名就可知道其文件夹下有哪些文件。app 文件下的每个.c 文件都对应 g-bios 中的

某个命令，文件名就是 g-bios shell 中的命令名。

```

bh/app {
    boot: 引导程序 (boot kernel or reboot, etc)
    flash: flash 操作相关 app
    led: 控制 led 灯程序
    Makefile
    memory: memory 操作函数
    misc: 杂项程序
    net: 网络程序
    serial: 串口程序
    shell: clear, ls, etc
    sysconf: configure 程序
}

```

目前 g-bios 仅支持 arm 体系结构的处理器，因此仅有 arm 目录。

```

$ ls /g-bios/bh/arm/
arm_heap.c  at91sam926x  exception.c  g-bios-bh.lds  head.S  lib  Makefile
omap3530   s3c24x0     s3c6410     udelay.S

```

```

bh/arm {
    arm_heap.c: heap init
    at91sam926x
    exception.c: exception handle
    g-bios-bh.lds: 编译 bh 所需的链接脚本。
    head.S: stage 2 入口
    lib: arm 体系结构相关库文件
    Makefile
    omap3530
    s3c24x0
    s3c6410
    udelay.S: udelay
}

```

其中 at91sam926x、s3c24x0、s3c6410 为平台相关代码：包括 soc 相有关的 driver，board 级初始化代码。

嵌入式开发要了解的几个概念：板级，SOC 级，CPU 级，CPU core 级。板级：与硬件开发板的布线有关。现在嵌入式处理器，通常将 SOC 级，CPU 级，CPU core 级做成一颗芯片。也就是所说的 SOC，例如 s3c6410，s3c2440，at91sam9261 等这就是一个 SOC。每个芯片公司将一个常用的外设芯片集成在一个 cpu 芯片里，就生产出了自己的 SOC 处理器芯片。这也就是说不同的 SOC，他们所集成的外设可能会不一样。因此对于不同的 SOC，他们的相对的处理代码也就可能不一样了。SOC 里集成了外设，所以也可将 SOC 叫做 platform 或“板子”。cpu core 也就是所说的体系结构例如：arm v5，arm v6，arm v7，core 2 等。cpu 就将 cpu core，MMU，cache 等等集成在一起的芯片，不同的 CPU，它们的 cache 个数、大小可能不同，因此

处理 cpu 级的程序代码也可能不同。(例如: arm 9, arm 11, cortex-a8 等)。其它厂商可以在此基础上加入一此外设从而制造处一个 SOC。因此, 对于不同的 SOC, 他们的 cpu 和 cpu core 可能相同 (s3c2410, s3c2440 他们的都是 arm 9 系列 cpu, 因此他们的 cpu 和 cpu-core 就是一样)。

有了以上基础后再来看 ARCH 相关的目录结构就十分简单了。s3c6410, at91sam926x, s3c24x0 为不同 cpu 的处理程序, 里含各个 SOC 处理的相关的代码。*.c, *.s 体系结构公共相关的代码。lib 体系结构相关库文件。

th 目录:

```
1 $ ls /g-bios/th/
2 arm base Makefile
```

g-bios 上半部分启动程序, arm 目录与上面所讲的 arm 目录类似, 里不再重述。base 目录上半部分与体系结构无关代码 (main, ymodem, kermit 等)。

driver 目录:

```
$ ls /g-bios/driver
flash gpu Makefile mmc net uart
```

不同外设的驱动程序源码。

driver	{	flash: flash 驱动程序 (nand flash, nor flash, dataflash 等)
		gpu: 显卡驱动程序 (lcd 等)
		mmc: mmc 相关驱动 (sd 卡等)
		net: net 相关驱动 (dm9000 网卡驱动, cs8900 驱动, 协议栈等)
		uart: 串口相关驱动。

Chapter 2

g-bios 的编译及烧录

2.1 g-bios 配置

与编译 Linux kernel 类似，进入在 g-bios 源码目录，执行 `make PLAT_defconfig`（PLAT 指的是具体硬件平台名称，例如 `s3c6410_defconfig` 或者 `beagle_defconfig`，g-bios 所支持的各硬件平台的默认配置文件位于 g-bios 源码 `build/configs` 目录下），用默认选项编译 g-bios，然后执行 `make` 进行编译，如需要将编译产生的 `image` 文件拷贝到 `tftpboot` 目录下，还需要执行 `make install` 命令。如果需要修改默认的编译选项，可以直接执行 `make menuconfig`，在随后出现的 GUI 中进行配置。

2.1.1 基于命令行的配置方式

```
$ cd g-bios
$ make mw6410_defconfig
$ make
$ make install
```

2.1.2 基于图形界面的配置方式

```
$ cd g-bios
$ make menuconfig
$ make
$ make install
```

g-bios 的 `configure` 程序支持图形界面的配置，使用者可在图形界面的配界相应功能。接下来分析一个各个配置选项的功能作用。

Platform: g-bios 运行的 platform，可以是 `at91sam9263`、`at91sam9261`、`s3c2410`、`s3c2440` 或 `s3c6410` 等，这是目前 g-bios 支持的几个 Platform。
Toolchain: 编译 g-bios 源码所选用的编译工具，默认使用的是 `lablin` 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）。
Image Patch 编译 g-bios 后生成的 image 路径，默认为 `/var/lib/tftpboot` 目录。
Server IP 服务器 IP，**local IP** 开发板 IP，将二者设为同一网段。此二项，也可不配。
MAC Addr 此项不用理会，**Nfs Path:** g-bios 引导内核时，如用 `nfs` 加载 `rootfs` 时，指定 `rootfs` 路径，默认路径 `~/maxwit/rootfs`。
Flash ECC mode 选择 ECC 校验模式（硬件 ECC，软件 ECC，也可不使用 ECC）。
IRQ/Polling Mode g-bios 使用中断模式还是非中断模式（Polling）。

g-bios 配置程序所完成的功能:

类别	选项	功能说明
general	Platform	g-bios 运行的目标 Platform
	TooLchain	编译 g-bios 的编译工具
	Image Path	编译生成 image 的目录
Network	Server IP	服务器 IP
	Local IP	目标机 IP
	MAC addr	MAC 地址
	NFS root path	lablin 的 rootfs 路径
Flash ECC Mode	Hardward	支持硬件 ECC
	Software	软件 ECC
	None	无 ECC 较验
IRQ/Polling Mode	IRQ Enabled	支持中断
	Polling Mode	查询模式 (非中断)

2.1.3 配置选项详解

- Gernerall
- Tophalf
- Uart
- Memory
- Flash
- NetWork
- Interrupt
- Logo
- Boot

2.2 编译

上面通过 configure 配置的 g-bios 编译特性, 生成了 Makefile。本节将编译 g-bios。

```
$ make
$ make install
```

编译后会在/var/lib/tftboot(configure 中配置的 Image Path) 目录下生成 g-bios-th.bin 和 g-bios-bh.bin 二个文件。

2.3 烧录

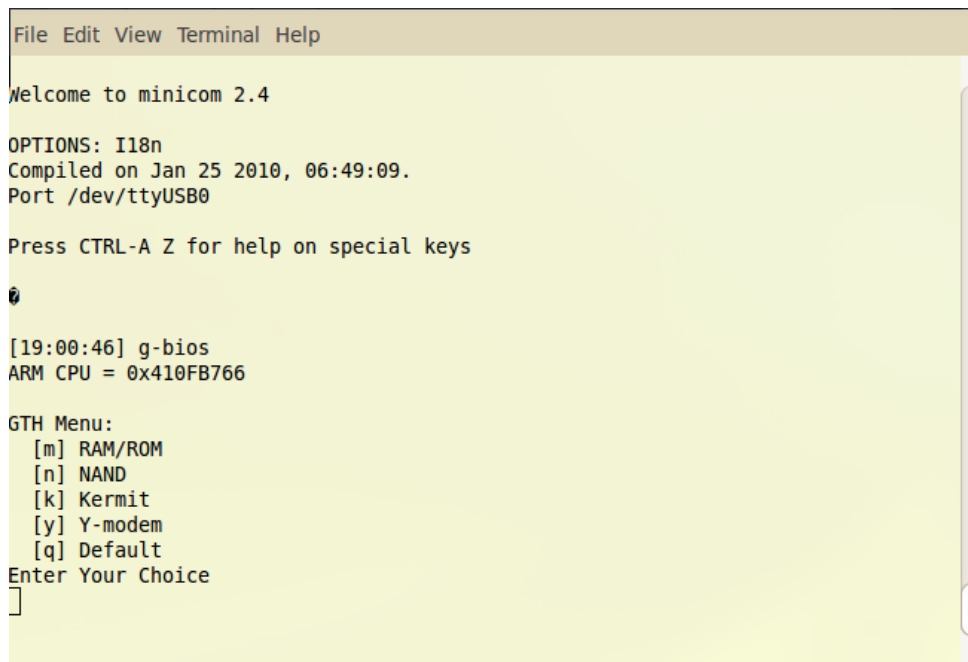
2.3.1 Burning Top-Half

g-bios 上半部分的烧录方法与其他的 bootloader 一样，都依赖于具体的板子，请大家参考板子的手册烧录 TH。g-bios 上半部主要是 Load 下半部，为下半部服务。Load BH 的方式有如下几种：

1. 从串口下载下半部并运行
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。

2.3.2 烧录 g-bios BH (下半部分)

上电或重起后，连接任意键可即可进入 g-bios TH 的引导菜单。若不按键刚 TH 默认从 Nand Flash 中 Load BH 并执行将执制权交给 BH。通过 TH Load BH 的菜单如下所示：



```
File Edit View Terminal Help
Welcome to minicom 2.4

OPTIONS: I18n
Compiled on Jan 25 2010, 06:49:09.
Port /dev/ttyUSB0

Press CTRL-A Z for help on special keys

[19:00:46] g-bios
ARM CPU = 0x410FB766

GTH Menu:
[m] RAM/ROM
[n] NAND
[k] Kermit
[y] Y-modem
[q] Default
Enter Your Choice

```

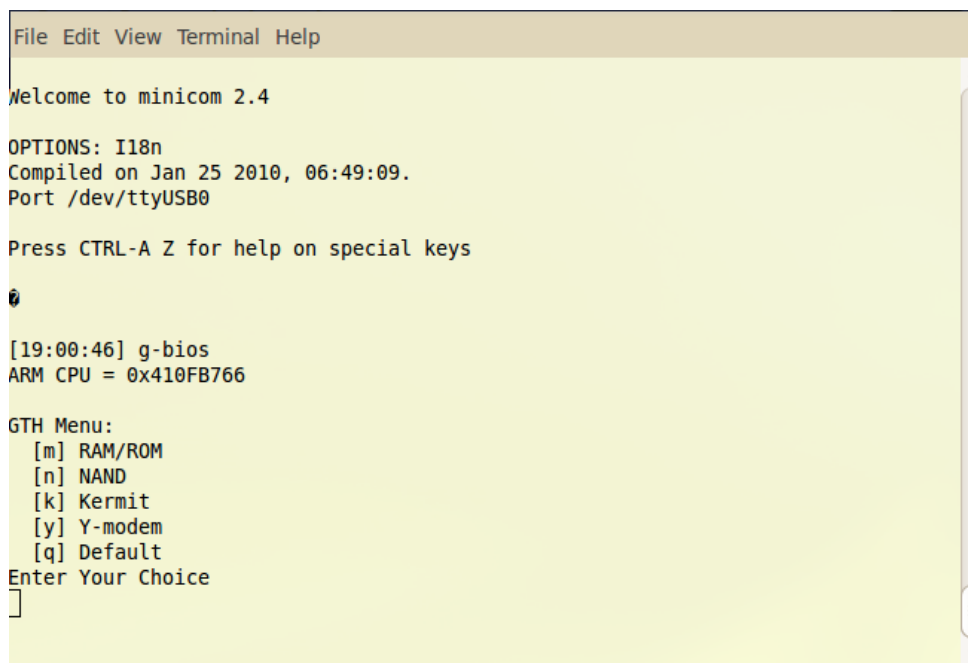
选择不同的选择即可以不同的方式 Load BH。以 Kermit 和 Minicom 为例从串口引导 g-bios bh。

1. Ymodem:(注意：minicom 在以下过程中要求使用速度很快。)

- (a) 连接电源，串口线 (开发板上的 COM1)，网线。
- (b) 打开 minicom 软件。

```
$ minicom
```

- (c) 按下空格键 (一直接)。打开电源开关。



```
File Edit View Terminal Help

Welcome to minicom 2.4

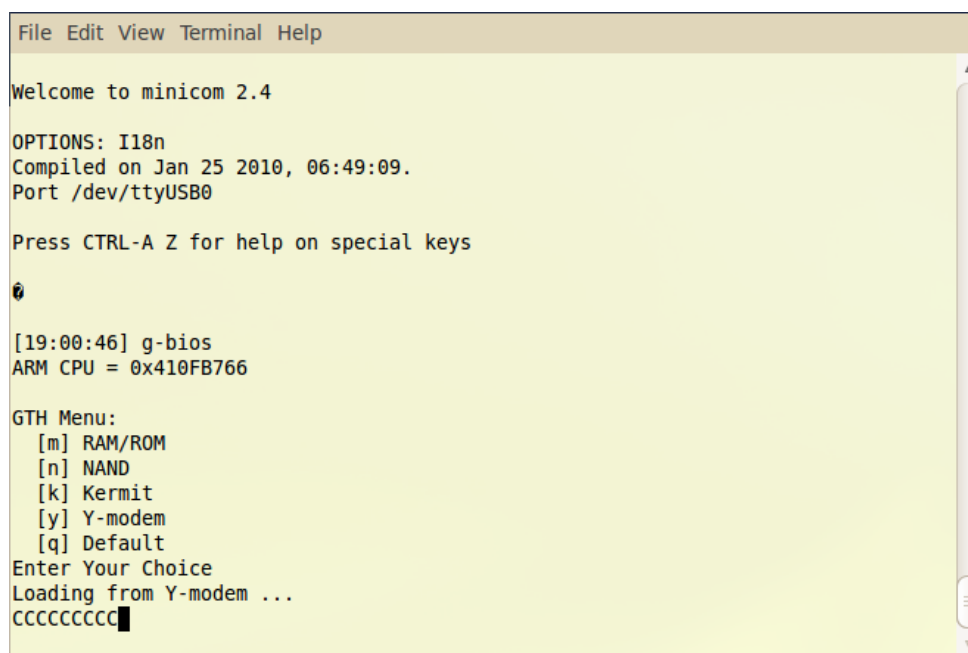
OPTIONS: I18n
Compiled on Jan 25 2010, 06:49:09.
Port /dev/ttyUSB0

Press CTRL-A Z for help on special keys

[19:00:46] g-bios
ARM CPU = 0x410FB766

GTH Menu:
[m] RAM/ROM
[n] NAND
[k] Kermit
[y] Y-modem
[q] Default
Enter Your Choice
y
```

(d) 按下 'Y' 键



```
File Edit View Terminal Help

Welcome to minicom 2.4

OPTIONS: I18n
Compiled on Jan 25 2010, 06:49:09.
Port /dev/ttyUSB0

Press CTRL-A Z for help on special keys

[19:00:46] g-bios
ARM CPU = 0x410FB766

GTH Menu:
[m] RAM/ROM
[n] NAND
[k] Kermit
[y] Y-modem
[q] Default
Enter Your Choice
y
Loading from Y-modem ...
CCCCCCCC
```

(e) 同时按下 “Ctrl” + 'a', 再按's'.

```
File Edit View Terminal Help

Welcome to minicom 2.4

OPTIONS: I18n
Compiled on Jan 25 2010, 06:49:09.
Port /dev/ttyUSB0

Press CTRL-A Z for help on sp| zmodem |
                               | ymodem |
                               | xmodem |
                               | kermi  |
                               | ascii  |
                               +-----+

[19:00:46] g-bios
ARM CPU = 0x410FB766

GTH Menu:
[m] RAM/ROM
[n] NAND
[k] Kermit
[y] Y-modem
[q] Default
Enter Your Choice
Loading from Y-modem ...
CCCCCCCCCCCCC
CTRL-A Z for help |115200 8N1 | NOR | Minicom 2.4 | VT102 | Offline
```

(f) 用方向键选择“ymodem”，回车。

```
File Edit View Terminal Help

Welcome to minicom 2.4

OPTIONS: I18n
Compiled on Jan 25 2010, 06:49:09.
Port /dev/ttyUSB0

Press CTRL-A Z for help on sp| zmodem |
                               | ymodem |
                               | xmodem |
                               | kermi  |
                               | ascii  |
                               +-----+

[19:00:46] g-bios
ARM CPU = 0x410FB766

GTH Menu:
[m] RAM/ROM
[n] NAND
[k] Kermit
[y] Y-modem
[q] Default
Enter Your Choice
Loading from Y-modem ...
CCCCCCCCCCCCC
CTRL-A Z for help |115200 8N1 | NOR | Minicom 2.4 | VT102 | Offline
```

(g) 选择“Okay”，回车。

```

File Edit View Terminal Help

We+-----[Select one or more files for upload]-----+
|Directory: /var/lib/tftpboot
OP| [...]
Co| g-bios-bh.bin
Po| g-bios-bh.dis
| g-bios-bh.elf
Pr| g-bios-th.bin
| g-bios-th.dis
CC| g-bios-th.elf
| g-bios.th
[1| gb.bin
AR| head.bin
| led.bin
GT| main
| main.bin
| rootfs.rd
| rootfs.yaffs2
|
| ( Escape to exit, Space to tag )
+-----+
Enter Your Choice
Lo          [Goto] [Prev] [Show] [Tag] [Untag] [Okay]
C
CTRL-A Z for help |115200 8N1 | NOR | Minicom 2.4 | VT102 | Offline

```

- (h) 输入 `/var/lib/tftpboot/g-bios-bh.bin`。回车。此处输入的为 `g-bios-bh.bin` 文件的路径，可视具体情况更改。

```

File Edit View Terminal Help

We+-----[Select one or more files for upload]-----+
|Directory: /var/lib/tftpboot
OP| [...]
Co| g-bios-bh.bin
Po| g-bios-bh.dis
| g-bios-bh.elf
Pr| g-bios-th.bin
| g-bios-th.dis
CC| g-bios-th.elf
| g-bios.th
[1| gb.bin          +-----+
AR| head.bin        |No file selected - enter filename: |
| led.bin          |> /var/lib/tftpboot/g-bios-bh.bin|
GT| main
| main.bin
| rootfs.rd
| rootfs.yaffs2
|
| ( Escape to exit, Space to tag )
+-----+
Enter Your Choice
Lo          [Goto] [Prev] [Show] [Tag] [Untag] [Okay]
C
CTRL-A Z for help |115200 8N1 | NOR | Minicom 2.4 | VT102 | Offline

```

- (i) 开发传输文件，传送完成后如下图所示。再次回车。即可进入 `g-bios Shell`。

```

File Edit View Terminal Help

Welcome to minicom 2.4

OPTIONS: I18n
Compiled on Jan 25 2010, 06:49:09.
Port /dev/ttyUSB0

+-----[ymodem upload - Press CTRL-C to quit]-----+
Press CTRL|Retry 0: NAK on sector
           |Retry 0: NAK on sector
0          |Bytes Sent: 136192   BPS:7921
           |
[19:00:46]|Transfer complete
ARM CPU = |
           |READY: press any key to continue...|
GTH Menu: +-----+
[m] RAM/ROM
[n] NAND
[k] Kermit
[y] Y-modem
[q] Default
Enter Your Choice
Loading from Y-modem ...
C
CTRL-A Z for help |115200 8N1 | NOR | Minicom 2.4 | VT102 | Offline

```

(j) 进入 g-bios Shell。如图所示。

```

File Edit View Terminal Help

-----+-----
0      0x00000000 - 0x00020000 128K    g-bios-th  "g-bios"
1      0x00020000 - 0x00060000 256K    g-bios-bh  "g-bios"
2      0x00060000 - 0x00080000 128K    sysconfig  "g-bios"
3      0x00080000 - 0x00280000 2M      linux      "linux"
4      0x00280000 - 0x00480000 2M      ramdisk    "ramdisk"
*5     0x00480000 - 0x04480000 64M      jffs2      "rootfs"
6      0x04480000 - 0x08480000 64M      yaffs2     "data_1"
7      0x08480000 - 0x10000000 123M512K ubifs      "data_2"
-----+-----

+-----+
| Welcome to MaxWit g-bios! |
| http://www.maxwit.com    |
| Dec 5 2010, 20:40:21     |
+-----+

g-bios running in polling mode!

Detecting ethernet speed for eth0(DM9000A) PHY1 ... 100M Full Duplex activated!

g-bios: 1# dm9000 link status changed!

```

如果上述操作失败，以返回第三步 (c 步) 重复操作，在第'r' 步，不再重新输入，而是直接回车。

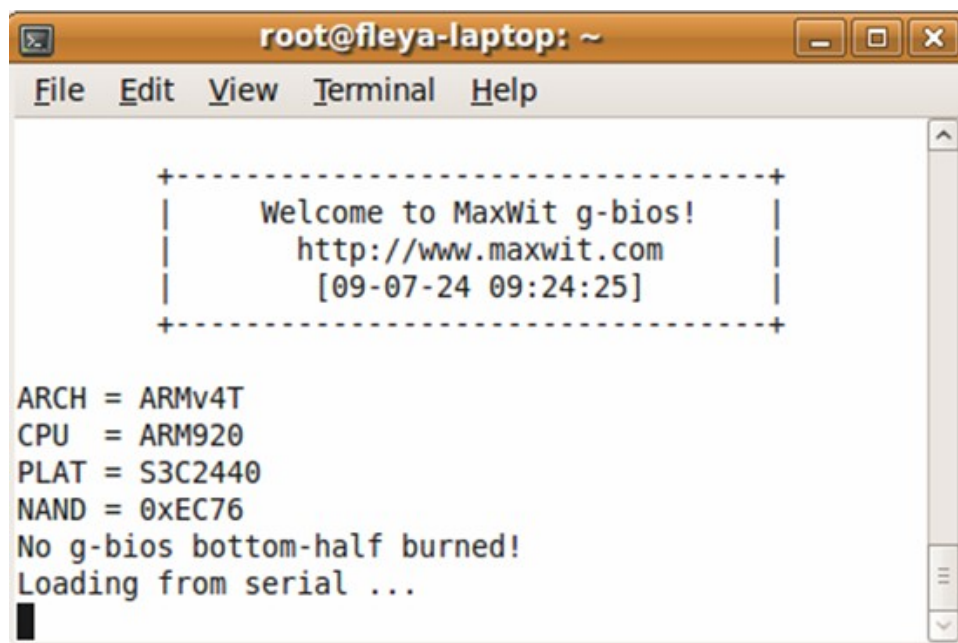
2. Kermit

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 kermit

```

1 $cd /var/lib/tftpboot
2 $kermit
3 C-kermit>c (回车)

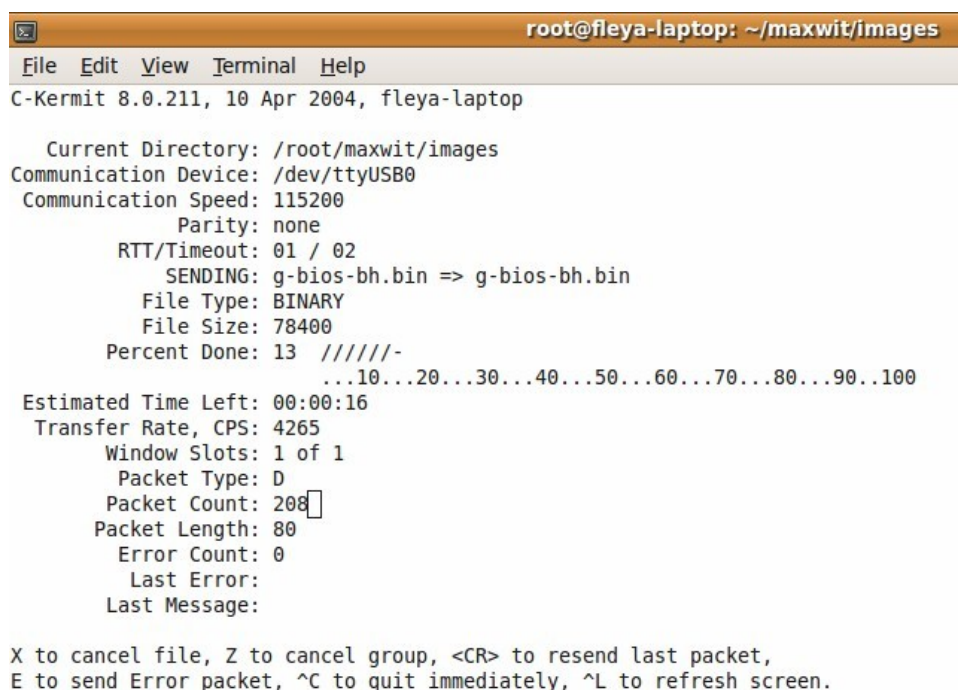
```



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)

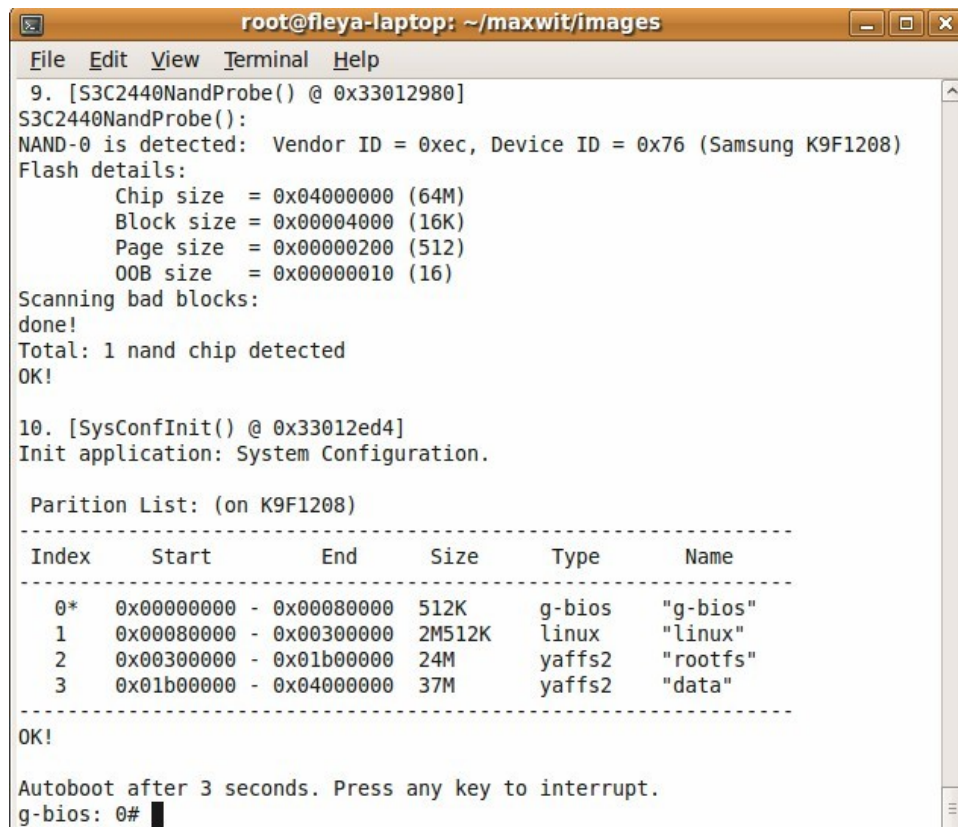
注：g-bios 的上半部会自动检测 Flash 上是否已烧录下半部 g-bios-bh.bin，若下半部已烧录则直接从 Flash 上将下半部 Load 到 Sdram 并运行，若未烧录则如上图所示，提示下半部未烧录，并需要通过从串口 Load 下半部并启动。下半部支持通过网络和串口两种方式烧录指定文件到 Flash 中。也在上半部启动过程中按任意键启动串口 Load 的功能。

第三步，选择 “k” 回车，然后同时按下 “CTRL” 和 “\” 键，再按下 “c”



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

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# █

```

2.3.3 串口

2.3.4 SD 卡

2.3.5 网络

1. Write TH Image to SD Card
2. Booting g-bios TH from SD
3. Load BH to RAM
4. Burning TH and BH

Chapter 3

g-bios 命令详解

3.1 flash 读写

3.1.1 flash

usage: flash < 子命令 > [可选参数 < 值 >]

子命令名称	命令说明
dump	查看 flash 页内容，以 page 为单位，包括 oob
read	从 Flash 中加载数据到 DDR
write	Flash 中写数据 from DDR
erase	以 block 为单位，擦除 flash 一块内容
scanbb	扫描指定分区上的坏快

参数	功能
-a	设定起始地址
-l	设定长度
-p	设定分区，不与 -a 和 -l 并用
-m	mem 的起始地址
-f	强制 erase（无论是否存在坏块）
-c	擦除的同时写入 cleanmark

命令使用示例：

- flash erase -a 1M -l 32K
- flash erase -a 100block -l 16block -f

3.1.2 part

usage: part [options]

参数说明:

参数	功能
-l	显示分区表

3.1.3 ls

显示当前分区的具体信息

3.1.4 cd

切换分区

3.2 MMC/SD 卡操作

3.2.1 mmc

usage: mmc < 子命令 > [options]

子命令名称	命令说明
scan	扫描所有 MMC/SD 设备，解析并打印设备信息
dump	查看 MMC/SD 上的数据，每次显示一个 block

MMC/SD 命令选项及参数介绍:

参数	功能
-a	设定起始地址

3.3 网络连接

3.3.1 ifconfig

usage: ifconfig [interface] [address] [netmask <address>] [hw [HW] <address>]

选项及参数介绍:

选项	功能描述
interface	指定网络设备对象，如“eth0”。缺省为系统中第一个网络设备。
address	配置网络设备 IP 地址为 address
netmask <address>	配置 netmask 为 address
hw [HW] <address>	配置设备的 MAC 地址为 address。其中 HW 缺省为“ether”

不加任何 option 时显示 interface 的信息，具体包括:

1. NIC 芯片名称（ID 及字符串表示）
2. PHY 信息：ID、地址

- 3. 连接状态, 包括速率
- 4. RX/TX bytes
- 5. error count

3.3.2 ping

usage: ping [DestIp]

若 DestIp 不指定, 则默认的 server ip

3.3.3 tftp

usage: tftp [options] [filename]

参数介绍:

选项	功能描述
-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

3.3.4 dhclient

usage: dhclient [options]

参数介绍:

选项	功能描述
-s	同时将 Server IP 更新为 DHCP Server

3.4 串口协议及工具

3.4.1 kermit

usage: kermit [options]

作用概述: 串口文件传输

选项及参数介绍:

选项	功能描述
-m [address]	将下载的数据放在内存中, 而不直写到 storage (如 Flash) 上。其中 address 为可选参数, 表示 memory 地址; 若不指定 address, 则由系统自动分配一块空间。

3.4.2 ymodem

usage: ymodem [options]

作用概述: 串口文件传输

选项及参数介绍:

选项	功能描述
-m [address]	将下载的数据放在内存中，而不直写到 storage（如 Flash）上。其中 address 为可选参数，表示 memory 地址；若不指定 address，则由系统自动分配一块空间。

3.5 Graphics 和 Display

3.5.1 lcd

usage: lcd [options]

参数介绍:

选项	功能描述
-l [all]	列出 LCD 的 video mode。all 表示所有 video mode，不加则仅显示当前的 video mode
-s <N>	将当前 LCD 的 video mode 设置为第 N 种 mode。

3.6 memory 数据查看及指令跳转

命令名称	命令说明
md	显示 memory 数据
mw	将数据写入 memory
memset	将某个 memory 空间写入值

3.6.1 memory read 和 write

3.6.2 memory set

3.6.3 任意地址跳转

usage: go <address>

address 跳转的目标地址，可十进制表示，也可十六进制表示。

示例: go 0xc000000 跳转到 0xc000000 处执行。

3.7 其他命令

3.7.1 help

列出 g-bios 系统中当前所有可用的命令

3.7.2 led

LED 灯测试

3.7.3 sysconf

命令名称	命令说明
-r <all net boot>	sysconf reset

Chapter 4

引导操作系统

boot command design ..

4.1 OS 引导

见第五章？

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]	用 ramdisk 启动。指定 filename, 则通过 tftp 下载 ramdisk image; 否则从本地 ramdisk 分区下载。
-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

4.2 TFTP + NFS

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 192.168.0.2:/home/maxwit/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

boot 程序具有记录功能, 即, 能记住用户输入的参数, 换句话, 再次输入 boot 时不再需要输入参数了, 除非你想重设参数。

4.3 FLASH + NFS

g-bios: 1# cd 3 (进入 Linux 分区)

g-bios: 3# ls (列出当前分区信息)

Partition Type = "linux"

Partition Base = 0x00080000 (512K)

Partition Size = 0x00200000 (2M)

Host Device = NAND 256MB 3.3V 8-bit

MTD Deivce = /dev/mtdblock3

Image File = "zImage" (1968220 bytes)

g-bios: 3# tftp zImage (下载 zImage 到当前分区)

"zImage": 192.168.2.101 => 192.168.2.100

1968220(1M898K92B) loaded

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:/hom

4.4 Booting from Flash

g-bios: 1# cd 3 (进入 Linux 分区)

g-bios: 3# ls (列出当前分区信息)

Partition Type = "linux"

Partition Base = 0x00080000 (512K)

Partition Size = 0x00200000 (2M)

Host Device = NAND 256MB 3.3V 8-bit

MTD Deivce = /dev/mtdblock3

Image File = "zImage" (1968220 bytes)

g-bios: 3# tftp zImage (下载 zImage 到当前分区)

"zImage": 192.168.2.101 => 192.168.2.100

1968220(1M898K92B) loaded

g-bios: 3# cd 5 (进入 Rootfs 分区)

g-bios: 5# tftp rootfs_1.jffs2 (下载 zImage 到当前分区)

g-bios: 5# boot -t -f 5

【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

Chapter 5

附录

5.1 g-bios commands overview

5.2 S3C24XX 系列平台烧录方法

5.3 AT91SAM 系列平台烧录方法

5.4 S3C64XX 系列平台烧录方法

S3C64XX and S5P as cases

5.5 OMAP3 系列平台烧录方法