龙芯 1B 简单开发指导

2012-04

1 龙芯 1B 处理器简单介绍

龙芯 1B 一款实现 MIPS32 兼容且支持 EJTAG 调试的双发射处理器，通过采用转移预测、寄存器重命名、乱序发射、路预测的指令 CACHE、非阻塞的数据 CACHE、写合并收集等技术来提高流水线的效率，1B 芯片具有以下关键特性：

- 集成一个 GS232 双发射龙芯处理器核，主频 266MHZ,指令和数据 L1 Cache 各 8KB
- 集成一个 24 位 LCD 控制器，最大分辨率可支持到 1920*1080@60Hz/16bit
- 集成 2 个 10M/100M/1000M 自适应 MAC
- 集成 1 个 32 位 133MHz DDR2 控制器，兼容 16 位 DDR2
- 集成 1 个 USB 2.0 接口，兼容 EHCI 和 OHCI
- 集成 1 个 8 位 NAND FLASH 控制器,最大支持 32GB
- 集成中断控制器，支持灵活的中断设置
- 集成 2 个 SPI 控制器，支持主从模式，支持系统启动
- 集成 AC97 控制器
- 集成 2 个全功能串口，4 个两线串口（最高 12 个双线串口）
- 集成 3 路 I2C 控制器，兼容 SMBUS
- 集成 2 个 CAN 总线控制器
- 集成 62 个 GPIO 端口
- 集成 1 个 RTC 接口
- 集成 4 个 PWM 控制器
- 集成看门狗电路

2 调试开发环境搭建

2.1 开发环境搭建

在嵌入式开发过程中，通常由于目标板（开发板）没有足够的资源来运行开发和调试工具，所以需要借助建立了交叉编译环境的宿主机。开发环境的搭建主要有以下几个方案：

a,直接安装 linux 操作系统。

b,在 WINDOWS 下安装虚拟机后,再在虚拟机中安装 LINXUX 操作系统;

c,两台电脑，一台 linux 服务器，一台 windows 客户端。

我们这里对这几种方案都做一些简单的介绍：

a,直接安装 linux 操作系统。

这种方式适合对 linux 系统常用操作有一定了解，能在 linux 下进行程序编写，交叉编译的开发人员。

b，在 windows 中安装 linux 操作系统虚拟机方案。

这种方式适合对 linux 不是太了解，在 windows 下完成程序的编写工作，只在 linux 环境下进行交叉编译的开发人员。

c，一台 linux 服务器，多个 windows 客户端方案。

这种方式主要是方便多人同时开发，Linux 服务器用来进行交叉编译程序使用。Windows 客户端主要是通过 SSH 远程登录到 linux 服务器进行操作。

不管使用的哪种方式，最终交叉编译程序所使用的 linux 环境基本是一样的。对于所使用的 linux 操作系统（我们没有特殊要求，只要 linux 系统能够正常工作即可）安装好网络服务，TFTP 服务，NFS 服务就可以（这里不在讲述各种服务的搭建）。

2.2 交叉编译工具安装

交叉编译环境的安装配置：

a，将交叉编译工具(gcc-4.3-ls232.tar.gz)拷贝到 linux 下，进行解压缩。

命令：tar xzvf gcc-4.3-ls232.tar.gz －C /

这样会在/opt/安装交叉编译工具。

b，为交叉编译工具的路径添加环境变量

命令：export PATH=/opt/gcc-4.3-ls232/bin:$PATH

输入命令：mipsel-linux-gcc -v 查看交叉编译工具版本信息，来验证交叉编译工具是否正常安装（提示版本信息即认为交叉编译工具已经正常安装）。

注：

这条命令的作用只在当前终端有效，即交叉编译环境只在当前终端起作用。

需要在整个系统建立交叉编译环境，可以把

export PATH=/opt/gcc-4.3-ls232/bin:$PATH 添加到 /root/.bashrc 文件最后一行。

也可以运行命令:

echo "export PATH=/opt/gcc-3.4.6-2f/bin:$PATH" >>/root/.bashrc。

然后打开新终端切换到超级用户权限（sudo su），运行 echo $PATH 查看验证。

如果需要交叉编译程序只要指定交叉编译工具即可（一般的交叉编译程序会有 CROSS_COMPILE 和 ARCH

两个变量，只要指定 CROSS_COMPILE=mipsel-linux- ARCH=mips 就可以了；或者只有 CC 变量 这时只要指定 CC=mipsel-linux-gcc 就可以了）。

2.3 常用调试环境搭建

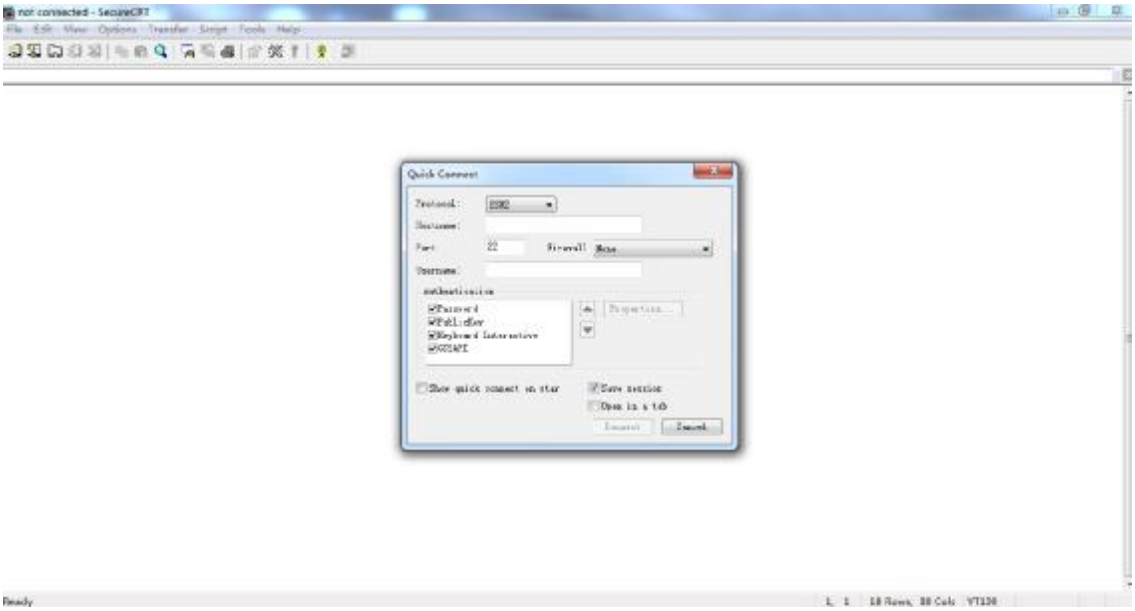
为了通过串口连接开发板，需要在 PC 机上使用一个终端仿真程序。

如果 PC 机是 Windows 操作系统，可以使用 SecureCRT 或者超级终端。Windows XP 上自带有超级终端软件，位于“开始-->程序-->附件-->通讯”，而 Windows 7 上需要额外安装，超级终端的设置请参考**“附录 Windows 超级终端使用说明”**。

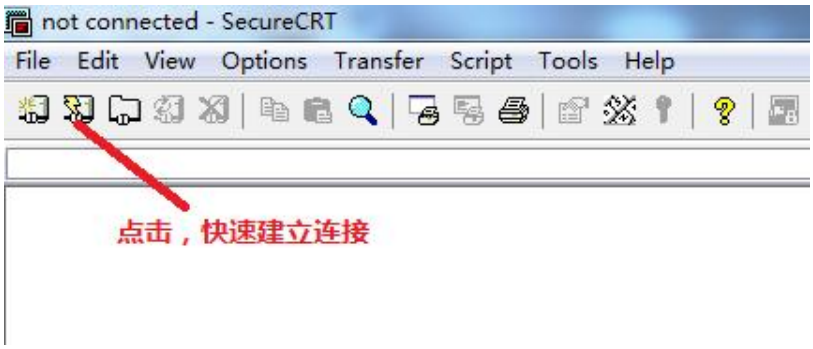
如果 PC 机是 Linux 操作系统，可以使用 minicom（详细请参考**“附录 Minicom 使用指南”**）。

下面以 Windows 操作系统上的 SecureCRT 终端仿真软件为例。

1）打开 SecureCRT，如下图：

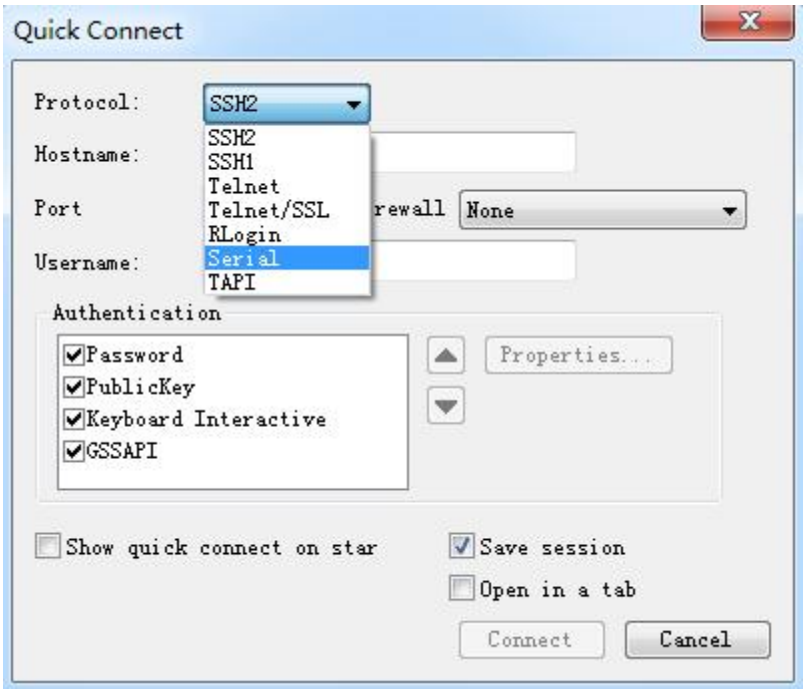


2）点击工具栏的带闪电图标，快速建立连接，如下图：

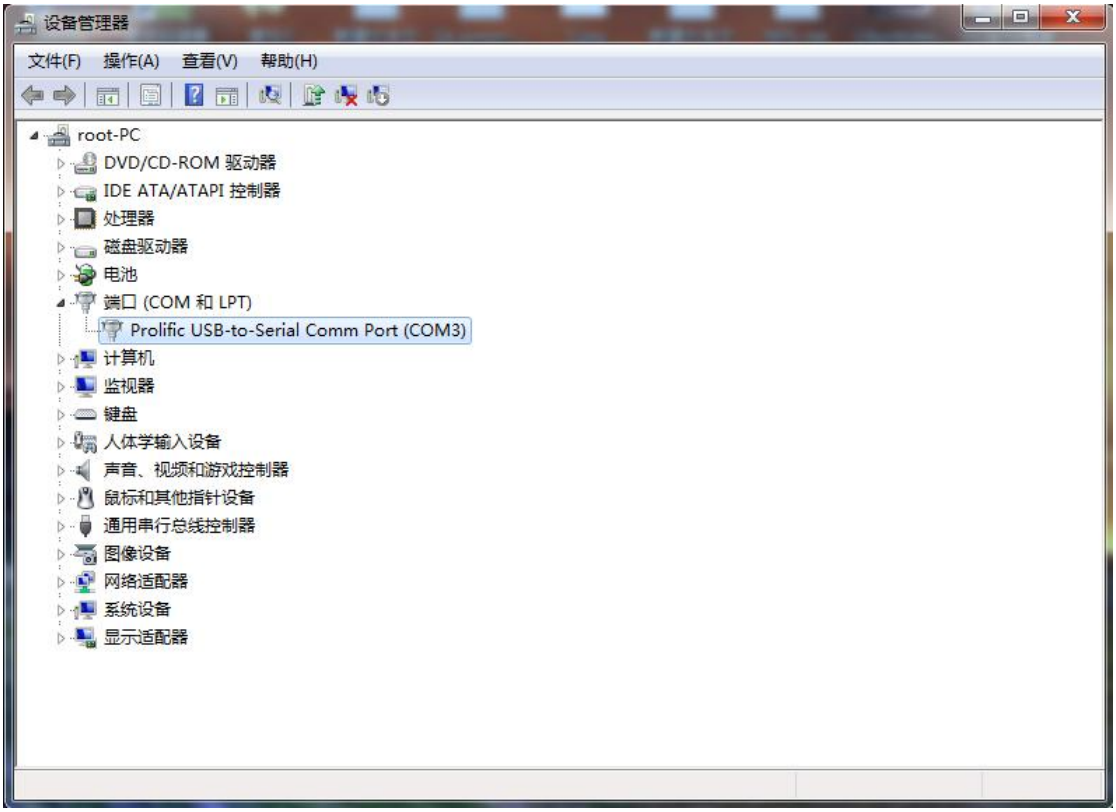


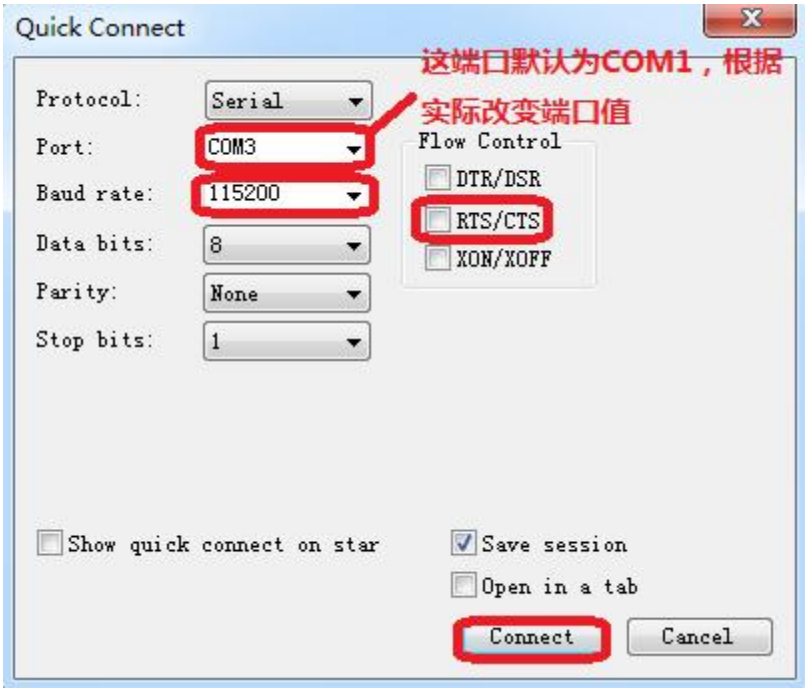
3）在 Quick Connect 对话框中，Protocol 的下拉菜单选择 Serial 协议选项，

选择串口协议，如下图：

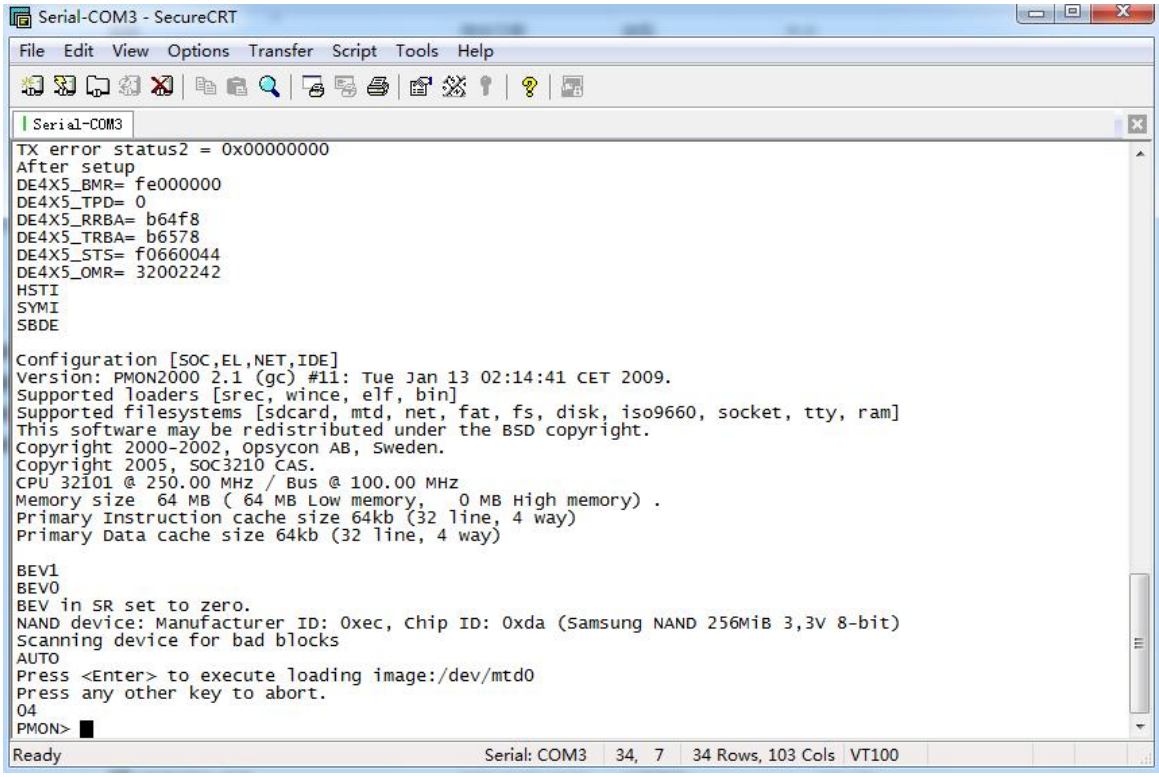


4）在 Quick Connect 对话框中，设置端口 Port 为 COM3，在“设备管理器”中可以看到串口的端口信息（注意：应根据具体信息设置正确的端口）；设置波特率 Baud rate 为 115200；去掉 Flow Control 中所有的勾选；最后 Connect，如下图：





5）开发板上电，按空格键，进入 BIOS（PMON），如下图：



3 基本开发

3.1 PMON

PMON 是一个兼有 BIOS 和 bootloader 部分功能的开放源码软件，多用于嵌入式系统。基于龙芯的系统采用 PMON 作为类 BIOS 兼 bootloader，并在其基础上做了很多完善工作，支持 BIOS 启动配置，内核加载，程序调试，内存寄存器显示、设置以及内存反汇编等等。

3.1.1 Pmon 编译

工具与依赖库安装

工具与依赖库的安装有两种方式：有互联网时使用 apt-get（ubuntu 系统）命令来安装；无互联网时使用源码包来安装。对于我们这里例子使用的 ubuntu 系统使用 apt-get 命令安装，要求 PC 机连接有互联网。

1）因为编译 PMON 过程需要使用到工具 pmoncfg，该工具 1b-pmon/tools/pmoncfg 目录下，编译该工具又需要依赖下面的工具：

```
#apt-get install bison
#apt-get install flex
```

2）解压 1b-pmon.tar.gz，编译生成 pmoncfg 工具：

```
#tar xzvf 1b-pmon.tar.gz
#cd 1b-pmon
#cd 1b-pmon/tools/pmoncfg
#make
```

3）编译完成后会在当前目录下生成 pmoncfg，拷贝该工具至用户工具目录或交叉编译工具链的 bin 目录（参看建立交叉编译环境）下。（推荐拷贝至交叉编译工具链目录中）

```
#cp pmoncfg /opt/gcc-4.3-ls232/bin
```

4）pmon 编译还依赖于工具 makedepend：

```
#apt-get install xutils-dev
```

配置与编译 pmon

1）建立交叉编译环境（第二章建立交叉编译环境）。

2）解决库与工具依赖以后，开始编译 pmon：

```
#cd 1b-pmon/zloader.ls1b
```

3）编译 bin 格式的 pmon

```
#make cfg ;make tgt=rom
```

执行后就在当前目录下生成了 gzrom.bin。

3.1.2 Pmon 更新

将新编译好的 pmon 文件（gzrom.bin）拷贝到 tftp 目录（确认能够正常访问 tftp 服务），设置好板子的 IP 地址，使用命令来更新应用新的 pmon。

1） 设置板子 IP 地址（参见 PMON 常用命令）

2）更新 PMON 命令：load –r –f 0xbfc00000 tftp://tftp-server-ip/gzrom.bin

该命令在执行完毕后会自动提示新 PMON 更新完成。重新启动即可应用新的 PMON。

注： a，在这里我们只介绍在现有的 PMON 之上更新新的 PMON。

b,在整个更新 Pmon 的过程中，注意不要断电。

3.1.3 Pmon 常用命令

参见附录

3.2 Kernel

Linux 内核很庞大，linux 初学者以及致力于 linux 应用软件开发的技术人员，熟悉内核的好的开始就是对内核进行配置，得到符合自己需求的经过裁剪的内核，并将编译后的内核下载到开发板中运行使用。本篇内容是为想要对内核进行个性配置的人员准备的，不涉及到代码编写，学习 linux 不必一切从“零”开始，一切可从学会配置、编译、下载运行开始。Linux 内核的编译分为两个步骤，一、内核配置；二、内核编译。开发包默认提供一个配置文件，用户可以根据此配置文件对内核进行裁剪或者增加新的功能。

注： Linux-2.6.21 内核为例，描述整个 Linux 系统的配置编译更新运行过程。

3.2.1 Kernel 编译

1）建立交叉编译环境（参见交叉编译工具安装和环境搭建）。

2）解压缩并进入 linux 源代码树根目录。

命令： **tar xzvf 1b-linux-2.6.21.tar.gz**
cd 1b-linux-2.6.21

3）确认安装图形化配置“make menuconfig”依赖的工具 Ncurses（如果安装过跳过此步骤）：

命令：**#apt-get install libncurses5-dev**

4）图形化配置(调整终端窗口到合适的大小)：

命令：**make menuconfig**

注： a，在内核的源代码中我们会提供一个默认的配置文件（名字为 config.def）。
b，参见附录中内核常用配置说明。

3.2.2 内核的编译

最终在内核源码树跟目录中，生成内核映像文件 vmlinux。

命令： **make vmlinux**

mipsel-linux-strip vmlinux (去除无用的调试和符号信息来减小内核的大小)

注：如果在环境变量中没有指定 CROSS_COMPILE 和 ARCH，需要指定这两个环境变量的值（参见交叉编译工具安装），CROSS_COMPILE=mipsel-linux- ARCH=mips

3.2.3 Kernel 更新

将新编译好的内核文件（vmlinux）拷贝到 tftp 目录（确认能够正常访问 tftp 服务），设置好板子的 IP 地址，使用命令来更新应用新的内核（此操作在 PMON 中进行）。

1）设置板子 IP 地址（参见 PMON 常用命令）
2）使用命令：devcp tftp://tftp-server-ip/vmlinux /dev/mtd0 将内核烧写到 nandflash 中。
3）设置启动参数：set al /dev/mtd0

4 文件系统

4.1 文件系统制作

说明：这里只介绍在提供的 ramdisk 基础上如何修改编辑文件系统内容,如要定制自己全新的文件系统,请参考网上资料。

首先将 ramdisk-fs.tar.gz 文件系统拷贝到一个目录下（假设：是 path 目录），解压,然后挂载到 /mnt/ramdisk-fs 目录

命令： **tar xzvf ramdisk-fs.tar.gz -C path**
mkdir -p /mnt/ramdisk-fs
mount -o loop -t ext2 path/ramdisk-fs /mnt/ramdisk-fs

修改自启动文件 /mnt/ramdisk-fs/etc/init.d/rcS

命令 ： vim /mnt/ramdisk-fs/etc/init.d/rcS

添加需要的命令后，保存退出。

卸载制作好的文件系统：

命令：umount /mnt/ramdisk-fs

这样文件系统就制作完成。

4.2 文件系统烧写

挂载制作好的文件系统到 /mnt/ramdisk-fs

命令：mount -o loop -t ext2 ramdisk-fs /mnt/ramdisk-fs

4.2.1 yaffs2

使用 yaffs2 文件系统格式，并将文件系统烧写到 nandflash 上面;

1）使用 yaffs2 镜像制作工具制作 yaffs2 的镜像文件 rootfs-yaffs2.img。

#mkyaffs2image /mnt/ramdisk-fs rootfs-yaffs2.img

#chmod 777 rootfs-yaffs2.img //修改文件系统权限，防止出现无法烧写的情况

2）将制作好的 yaffs2 镜像文件(rootfs-yaffs2.img)烧写到 nandflash 上面（这里提供通过网络烧写的命令）。

命令：devcp tftp://tftp-server-ip/rootfs-yaffs2.img /dev/mtdxc

注： 这里的/dev/mtdx 指代是 nandflash 分区，参见附录 nandflash 分区。

4.2.2 jffs2

使用 jffs2 文件系统格式，并将文件系统烧写到 nandflash 上面;

1）使用 jffs2 镜像制作工具制作 jffs2 的镜像文件 rootfs-jffs2.img。

```
#mkfs.jffs2 -r /mnt/ramdisk-fs -o rootfs-jffs2.img -e 0x20000    --pad=0x2000000 -n
#chmod 777 rootfs-jffs2.img          //修改文件系统权限，防止出现无法烧写的情况
```

mkfs.jffs2 各参数的意义：

-r：指定要生成 image 的目录名。

-o：指定输出 image 的文件名。

-e：每一块要擦除的 block size，不同的 flash，其 block size 会不一样。这里为 128KB。

--pad：用 16 进制来表示所要输出文件的大小，也就是 rootfs-jffs2.img 的大小，如果实际大小不足此设定的大小，则用 0xFF 补足。

-n，-no-cleanmarkers：指明不添加清楚标记（nandflash 有自己的校检块，存放相关的信息）。

2） 将制作好的 jffs2 镜像文件(rootfs-jffs2.img)烧写到 nandflash 上面（这里提供通过网络烧写的命令）。

```
命令：devcp tftp://tftp-server-ip/rootfs-jffs2.img    /dev/mtdx
```

注：这里的 nandflash 分区。

4.2.3 cramfs

使用 cramfs 文件系统格式，并将文件系统烧写到 nandflash 上面;

1） 使用 cramfs 镜像制作工具制作 jffs2 的镜像文件 rootfs-cramfs.img

```
#mkfs.cramfs    /mnt/ramdisk-fs      rootfs-cramfs.img
#chmod 777 rootfs-cramfs.img          //修改文件系统权限，防止出现无法烧写的情况
```

2） 将制作好的 cramfs 镜像文件(rootfs-cramfs.img)烧写到 nandflash 上面（这里提供通过网络烧写的命令）。

```
命令：devcp tftp://tftp-server-ip/rootfs-cramfs.img    /dev/mtdx
```

注：这里的 nandflash 分区。

附录

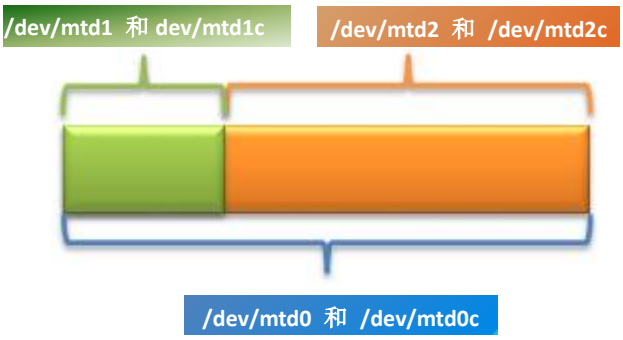
1 Pmon 常用命令

类型	命令	说明	例子	例子含义
帮助	h	查看帮助信息	h	列出所有可以使用命令
			h ping	查看 ping 命令的用法
调试	d1	读某个地址的值(读一个 byte)	d1 0x80300000	查看地址 0x80300000 处的值
	d2	读某个地址的值(读一个 halt word)	d1 0x80300000	查看地址 0x80300000 处的值
	d4	读某个地址的值(读一个 word)	d4 0x80300000	查看地址 0x80300000 处的值
	m1	在某个地址处写入一个值(写入一个 byte 大小)	m1 0x80300000 0x12	在地址 0x80300000 处写入 0x12
	m2	在某个地址处写入一个值(写入一个 halt word 大小的值)	m2 0x80300000 0x1234	在地址 0x80300000 处写入 0x1234
	m4	在某个地址处写入一个值(写入一个 word 大小的值)	m4 0x80300000 0x12345678	在地址 0x80300000 处写入 0x12345678
				读出此地址值为 0x12345678
内存	mt	内存测试命令	mt	测试板的内存是否正常
	load	下载 linux 内核到内存	load tftp://192.168.3.18/vmlinux	从网络 IP 为 192.168.3.18 的主机下载内核 vmlinx 到内存
网络	ifaddr	设置板的 ip 地址(只当次有效，断电后会丢失)	ifaddr syn0 192.168.3.25	设置板的 ip 地址为 192.168.3.25

	ifconfig	查看设置的 ip 地址信息	Ifconfig syn0	查看网络口 syn0 的信息
	ping	测试网络	ping 192.168.3.1	测试与 192.168.3.1 网口是否连通
环境管理	set	设置环境变量；设置的参数会保存到 norflash 高位地址，在 pmon 一开始运行时就会自动去调用	set	列出所有已经设置好的环境变量
	ping	测试网络	set ifconfig syn0:192.168.3.88	设置开发板的 IP 地址，重启开发板后 IP 地址固定存在
flash 管理	set env devcp	设置环境变量；设置的参数会保存到 nandflash 高位地址，在 pmon 一开始运行时就会自动去调用 查看板上已经设好的环境变量 PMON 上的拷贝下载，通常拷贝从网络下载的文件到 Nand Flash 中	set al /dev/mtd0	自动从 nandflash 的 mtd0 分区 load 内存，设置板在一上电时自动执行 load 内 核到内存操作
			set append 'console=ttyS0'	设置板的运行的启动 参数
			set	列出所有环境变量
			devcp tftp://192.168.3.18/vmlinux /dev/mtd0	从网络下载 vmlinux 到内存中并拷贝到 nandflash 中
	mtd_erase	擦除 Nand Flash 某分区的数据	mtd_erase /dev/mtd1	擦除 Nand Flash 分区 1 的数据
其他	reboot	重启 PMON	reboot	重启 PMON
	devls	查看设备列表	devls -n	查看网络设备

2 NandFlash 分区说明

一般我们只添加了两个 nand 分区。在 PMON 中分别对应设备名字是/dev/mtd1 和 /dev/mtd2 来表示。



如图所示 /dev/mtd1 指的是第一个分区/dev/mtd2 指的是第二个分区；而/dev/mtd0 指的是整个 nandflash，将整个 nandflash 看成一个分区（/dev/mtd0 /dev/mtd1 /dev/mtd2 不包括 spare（oob）区域）。

而设备文件 /dev/mtd0c /dev/mtd1c /dev/mtd2c （名字的后面多一个字符“c”），指的是包括 spare（oob）区域对应分区。

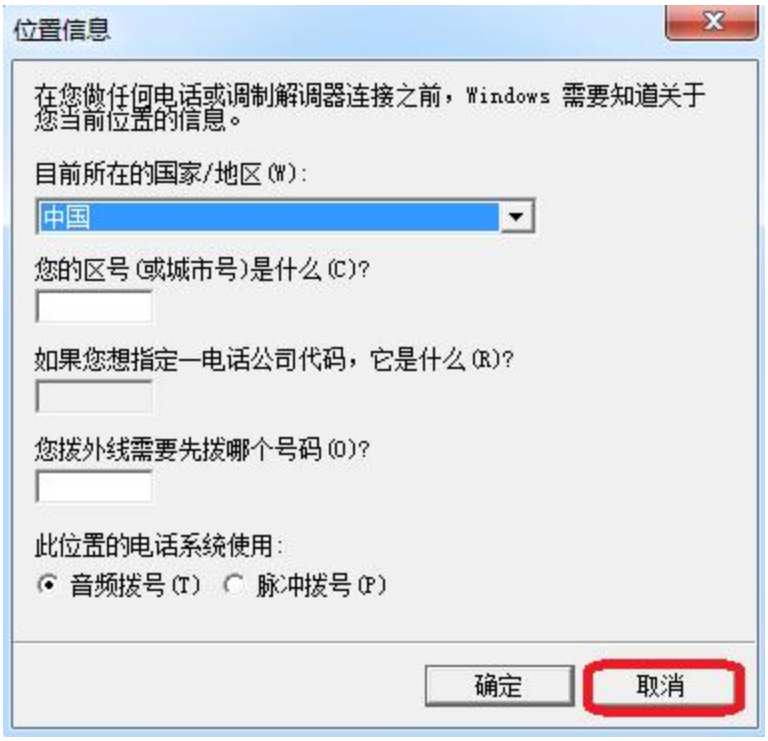
在烧写 yaffs2 镜像文件的时候需要读写 nandflash 的 spare（oob）区域，所以用的设备文件是/dev/mtd0c /dev/mtd1c 或者 /dev/mtd2c。而在烧写 jffs2 或者 cramfs 镜像文件的时候，不需要操作 spare（oob）区域。所以使用的是/dev/mtd0 /dev/mtd1 或者 /dev/mtd2。

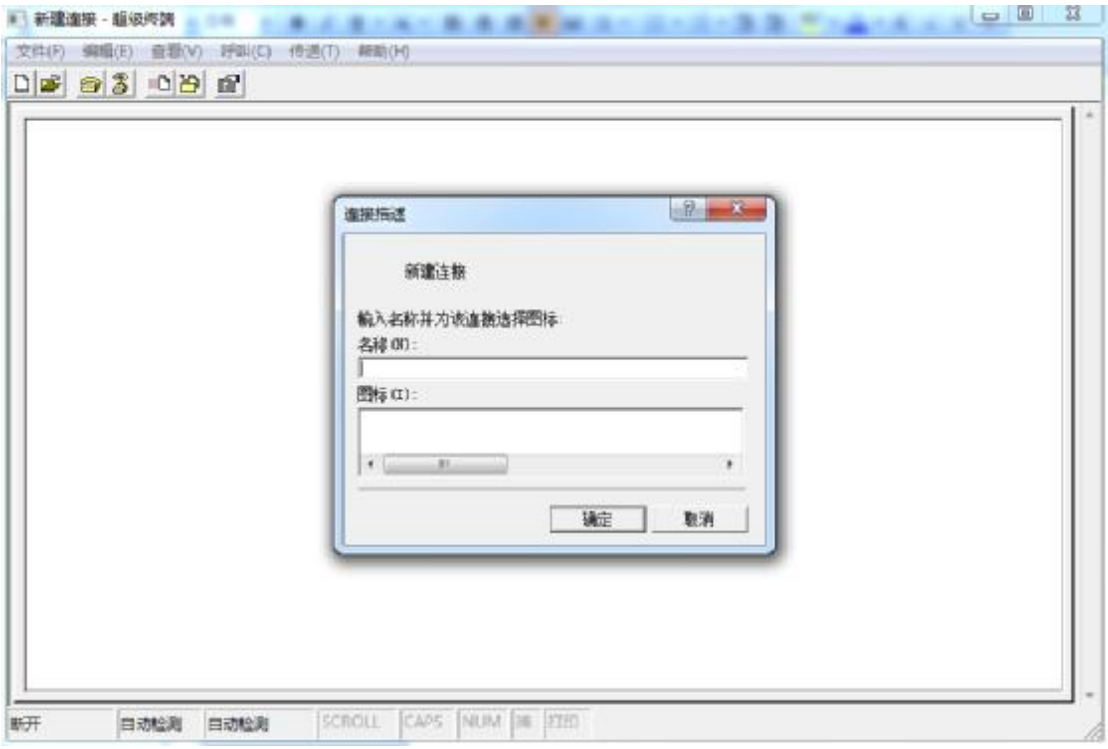
设备文件名	说明
/dev/mtd0	整个 nandflash ，不包括 spare（oob）区域
/dev/mtd1	第一个 nandflash 分区 ，不包括 spare（oob）区域
/dev/mtd2	第二个 nandflash 分区，不包括 spare（oob）区域
/dev/mtd0c	整个 nandflash ，包括 spare（oob）区域
/dev/mtd1c	第一个 nandflash 分区 ，包括 spare（oob）区域
/dev/mtd2c	第二个 nandflash 分区，包括 spare（oob）区域

3 内核编译常用配置

4 Windows 超级终端使用说明

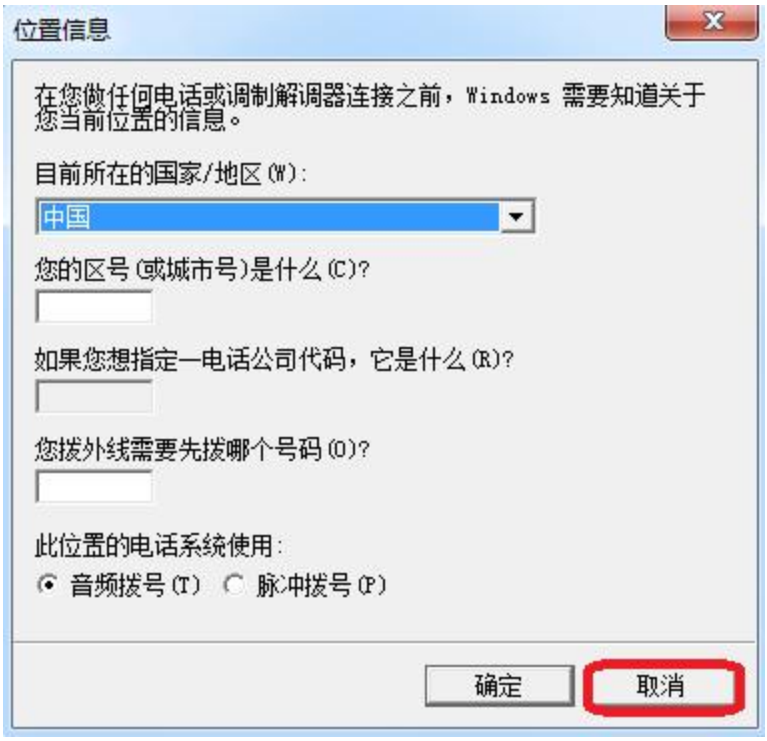
（1）打开超级终端：



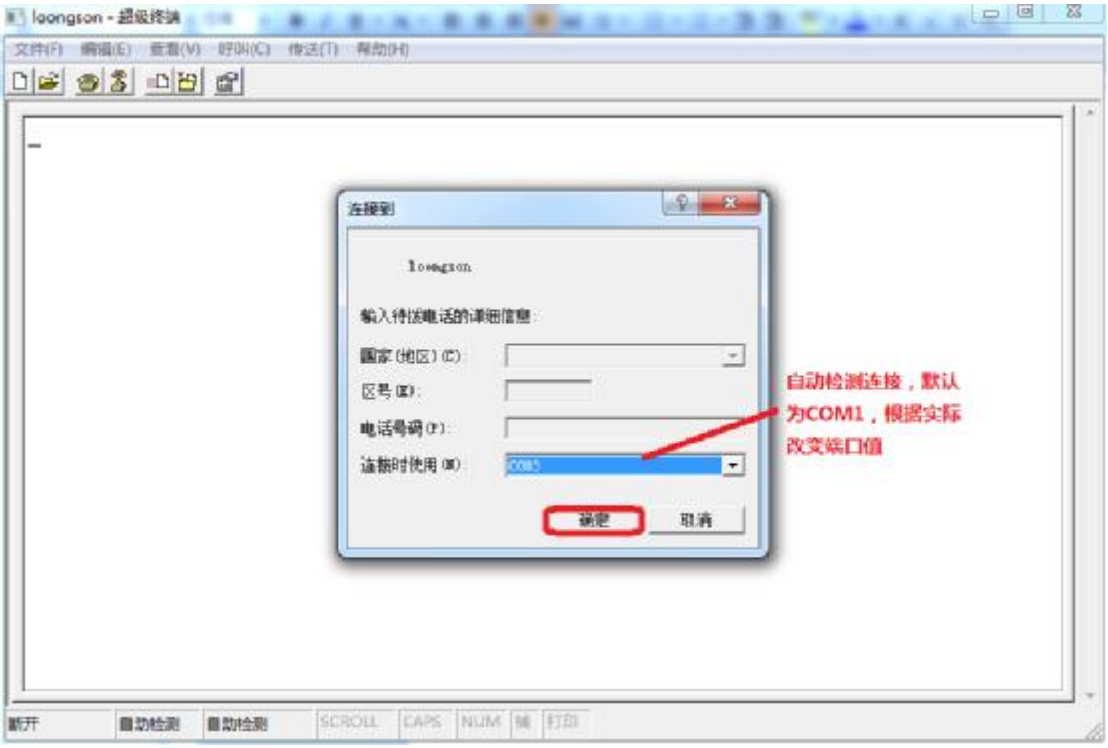


（2）新建连接，修改名称：

连接描述
新建连接
输入名称并为该连接选择图标:
名称(N):
loongson
图标(I):
确定
取消


位置信息
在您做任何电话或调制解调器连接之前，Windows 需要知道关于您当前位置的信息。
目前所在的国家/地区(W):
中国
您的区号(或城市号)是什么(C)?
如果您想指定一电话公司代码，它是什么(R)?
您拨外线需要先拨哪个号码(O)?
此位置的电话系统使用:
音频拨号(T)
脉冲拨号(P)
确定
取消

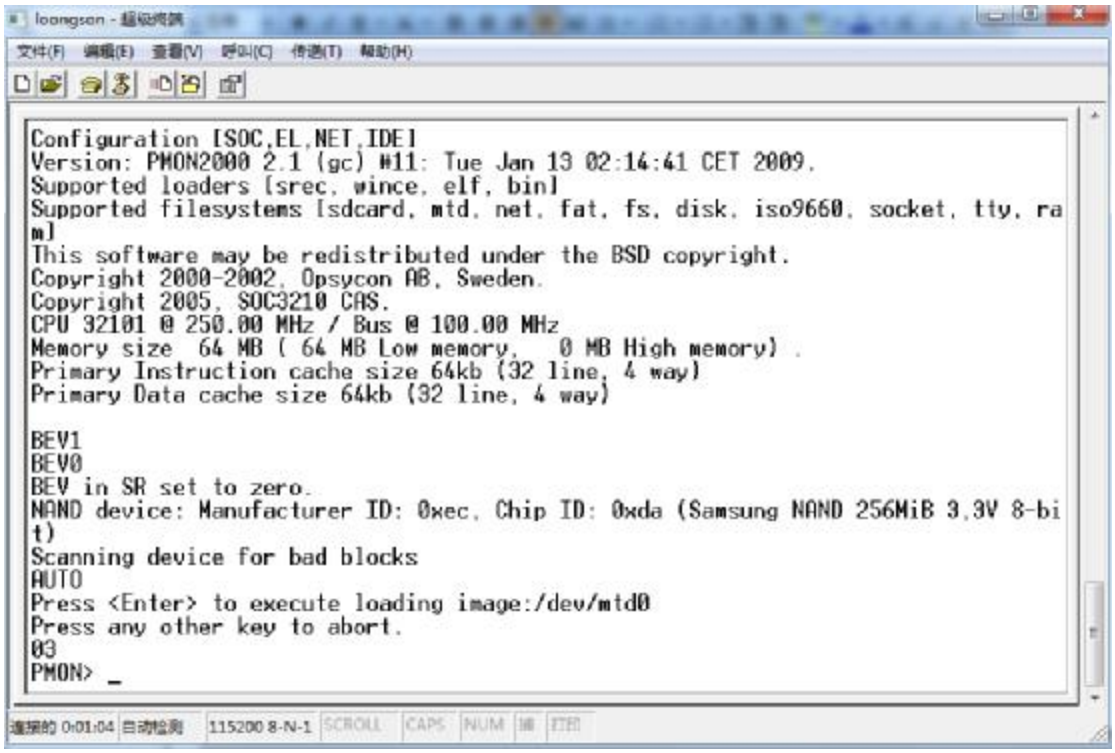
（3）设置：


loongson - 超级终端
连接到
loongson
输入待拨电话的详细信息:
国家(地区)(C):
区号(E):
电话号码(P):
连接时使用(N):
COM5
确定
取消
自动检测连接，默认为COM1，根据实际改变端口值

COM3 属性
端口设置
位/秒(B):
2400
数据位(D):
8
奇偶校验(P):
无
停止位(S):
1
数据流控制(F):
硬件
还原为默认值(R)
确定
取消
应用(A)

COM3 属性
端口设置
位/秒(B): 115200
数据位(D): 8
奇偶校验(P): 无
停止位(S): 1
数据流控制(F): 无
还原为默认值(R)
确定
取消
应用(A)

（4）开发板上电：


Configuration [SOC,EL,NET,IDE]
Version: PMON2000 2.1 (gc) #11: Tue Jan 13 02:14:41 CET 2009.
Supported loaders [srec, wince, elf, bin]
Supported filesystems [sdcard, mtd, net, fat, fs, disk, iso9660, socket, tty, ram]
This software may be redistributed under the BSD copyright.
Copyright 2000-2002, Opsycon AB, Sweden.
Copyright 2005, SOC3210 CAS.
CPU 32101 @ 250.00 MHz / Bus @ 100.00 MHz
Memory size 64 MB (64 MB Low memory, 0 MB High memory) .
Primary Instruction cache size 64kb (32 line, 4 way)
Primary Data cache size 64kb (32 line, 4 way)
BEV1
BEV0
BEV in SR set to zero.
NAND device: Manufacturer ID: 0xec, Chip ID: 0xda (Samsung NAND 256MiB 3.3V 8-bit)
Scanning device for bad blocks
AUTO
Press <Enter> to execute loading image:/dev/mtd0
Press any other key to abort.
03
PMON> _

5 Minicom 使用指南

minicom 是 Linux 上最常用的终端仿真程序，它类似于 Windows 下的“超级终端”的程序，一般完全安装大部分发行版的 Linux 时都会包含它，下面介绍它的使用方法。

安装 minicom

如果没有安装 minicom，先安装：

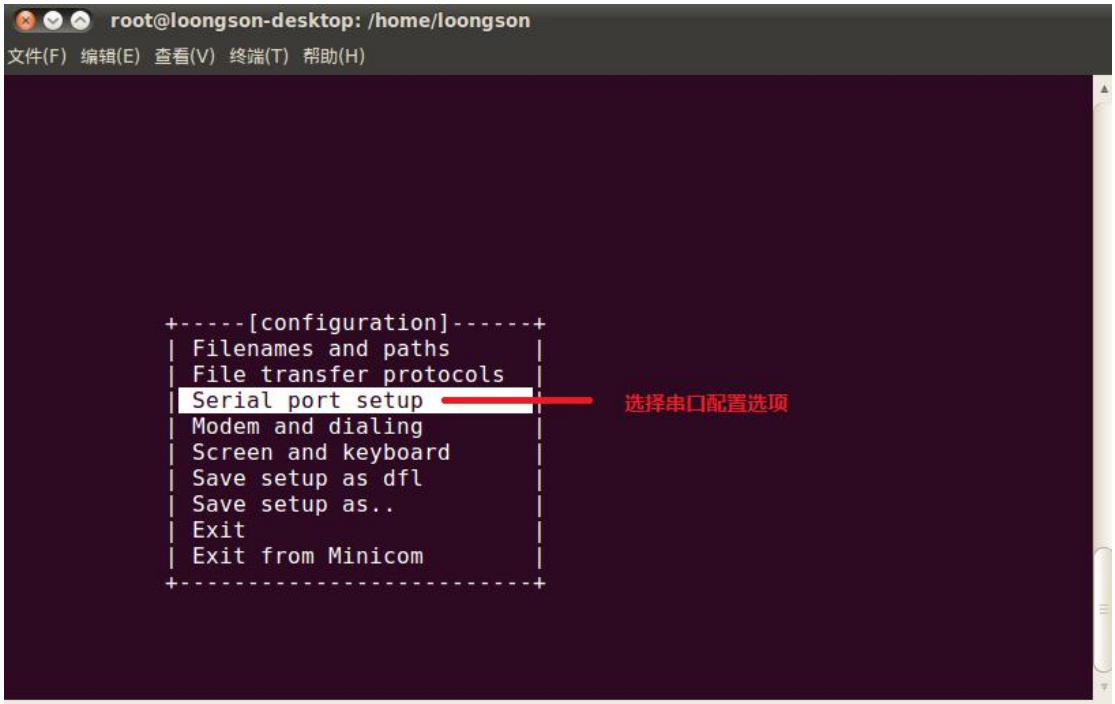
apt-get install minicom

配置 minicom

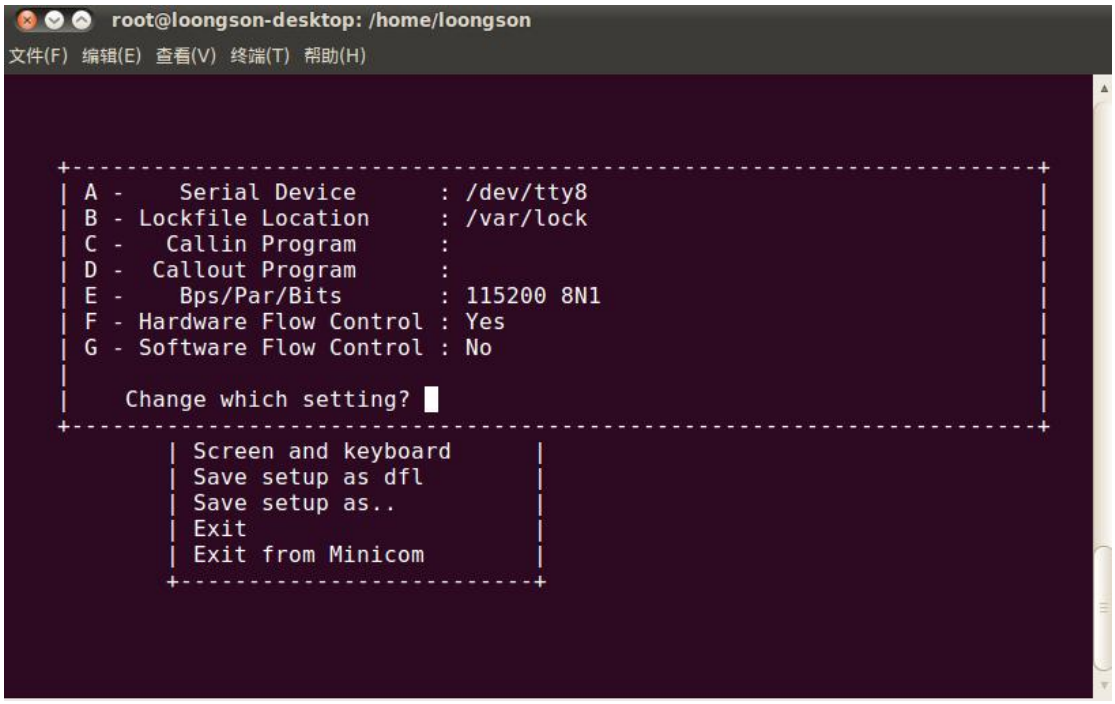
使用 minicom 之前先设置一下：

（1）Linux 终端输入

minicom -s



（2）进入“Serial port setup”：



提示：

A 对应的是键盘‘A’键，按下‘A’键配置串口设备。通常的配置是/dev/ttyS0 或/dev/tty0 或者/dev/ttyUSB0, 对应于 windows 操作系统的 COM1。

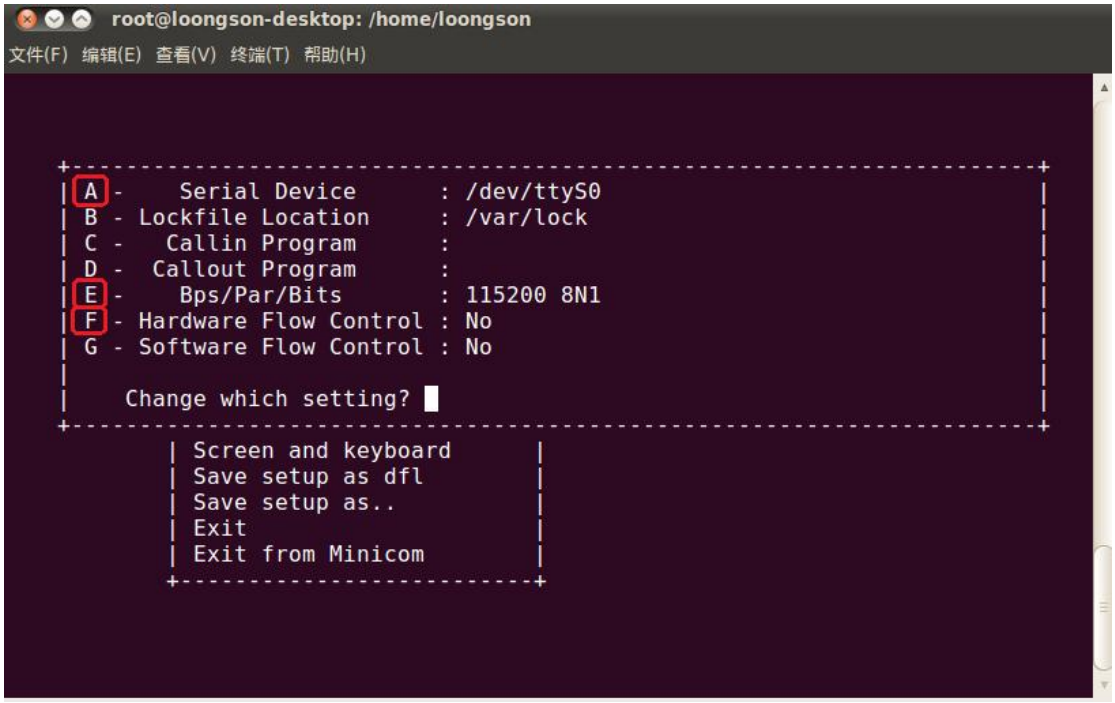
F 对应的是硬件流设置，按‘F’进行配置，通常选择 NO 。

E 对应的是键盘‘E’键，是用来配置串口波特率的，按键‘E’选择波特率 。

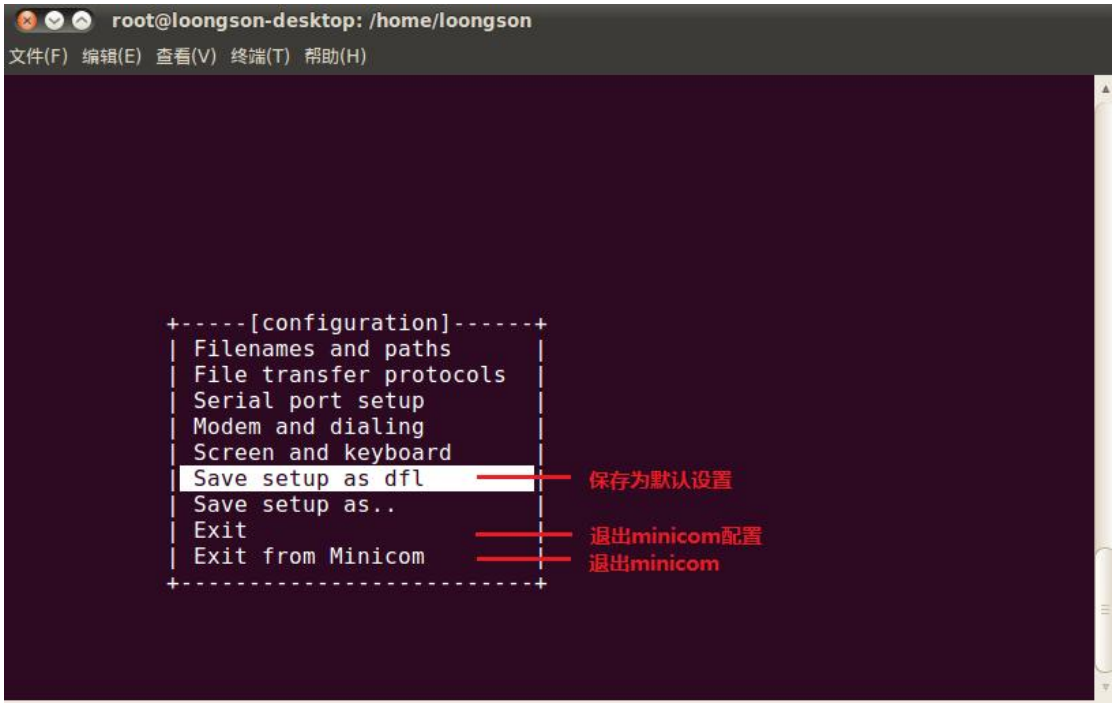
键盘‘ESC’键，退出，返回上一层。

键盘‘Enter’键，确定，保存。

（3）配置串口设备、波特率和硬件流：



（4）保存退出到主配置界面：



使用 minicom

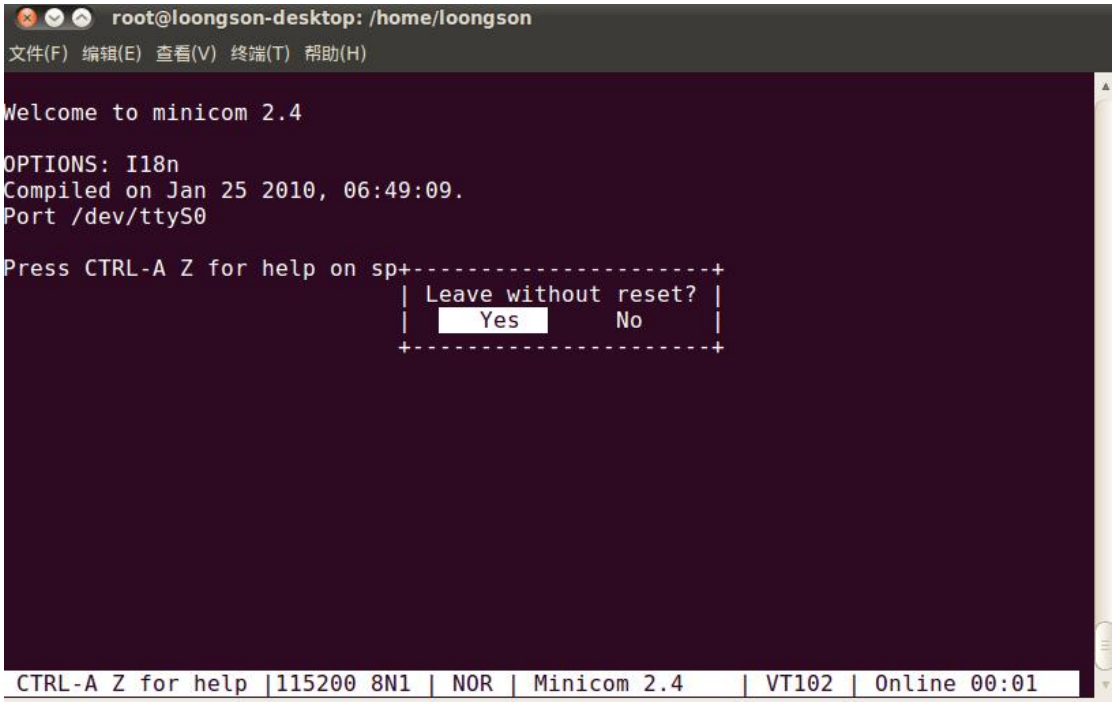
先连接好串口线或 USB 转串口。

退出 minicom 配置，进入 minicom：

```
root@loongson-desktop: /home/loongson
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)

Welcome to minicom 2.4

OPTIONS: I18n
Compiled on Jan 25 2010, 06:49:09.
Port /dev/ttyS0

Press CTRL-A Z for help on special keys
```

```
root@loongson-desktop: /home/loongson
文件(F) 编辑(E) 查看(V) 终端(T) 帮助(H)
SBDE

Configuration [SOC,EL,NET,IDE]
Version: PMON2000 2.1 (gc) #11: Tue Jan 13 02:14:41 CET 2009.
Supported loaders [srec, wince, elf, bin]
Supported filesystems [sdcard, mtd, net, fat, fs, disk, iso9660, socket, tty,]
This software may be redistributed under the BSD copyright.
Copyright 2000-2002, Opsycon AB, Sweden.
Copyright 2005, SOC3210 CAS.
CPU 32101 @ 250.00 MHz / Bus @ 100.00 MHz
Memory size  64 MB ( 64 MB Low memory,   0 MB High memory) .
Primary Instruction cache size 64kb (32 line, 4 way)
Primary Data cache size 64kb (32 line, 4 way)

BEV1
BEV0
BEV in SR set to zero.
NAND device: Manufacturer ID: 0xec, Chip ID: 0xda (Samsung NAND 256MiB 3,3V 8)
Scanning device for bad blocks
AUTO
Press <Enter> to execute loading image:/dev/mtd0
Press any other key to abort.
01
PMON>
```

在进入 minicom 后，如果想退出，则可以按下 ctrl+A，松开 ctrl+A 后再紧接着按下 Q 就可以看到退出提示了，选择 Yes 退出。



6 Pmon 下常用操作

1） 查看设备

命令： devls

2） 配置 ip 地址

命令：ifaddr syn0 xxx.xxx.xxx.xxx （syn0 为用 devls 查看得到的网卡名字）

3） 简单测试网络

命令：ping other-ip （other-ip 为在同一网段其他 PC 机器 IP）

4） pmon 下通过网络加载内核到内存

命令：load tftp://tftp-server-ip/vmlinux

注：此命令正常执行会打印加载信息

5） PMON 下通过网络烧写更新替换新 pmon

命令：load –r –f 0xbfc00000 tftp://tftp-server-ip/gzrom.bin

注：0xbfc00000 为龙芯处理器启动的第一条指令地址（映射到 Spiflash 中，即从 Spiflash 启动。

gzrom.bin 为 pmon 的二进制名字。

6） 烧写内核到 nandflash（通过网络）

命令：devcp tftp://tftp-server-ip/vmlinux /dev/mtd0

注：这里也可以是 /dev/mtd1 或者 /dev/mtd2 ，分别烧写到不同分区，需要在内核加

载参数中设置不同的内核启动位置。

7） 从 nandflash 中加载内核

命令： load /dev/mtd0

注：/dev/mtd0 需要根据内核的实际位置来修改（可能是/dev/mtd1 /dev/mtd2 等）。

8） 内核添加参数启动

命令：g console=ttyS0,115200 rdinit=/sbin/init

注：使用方式是 g 后面跟内核参数，例子中的参数根据实际的使用来修改。

8）设置内核启动加载位置环境变量

命令：set al /dev/mtd0

注：/dev/mtd0 需要根据内核的实际位置来修改（可能是/dev/mtd1 /dev/mtd2 等）。

PMON 启动后会自动根据 al（autoload）的值来自动加载内核。

9） 设置内核启动参数的环境变量

命令：set append "console=ttyS0,115200 rdinit=/sbin/init"

注：内核启动参数根据实际情况修改。

PMON 在启动后根据 al 的值自动加载内核，而后自动用 append 的值做为内核启动参数来启动内核系统。

10）设置延时启动时间环境变量

命令：set bootdelay 3

注：单位为秒（3：为延时 3 秒）

PMON 启动后根据 bootdelay 的值来延时加载启动系统内核。

11）重启 PMON

命令：reboot