**Busybox简介**

这里对busybox进行简单地介绍。  
busybox是Linux的一套工具集合，它将常用的linux命令例如"ls"，"echo"，"cd"，甚至是"sh"，"vi"，等用最简单的方式进行了实现，然后通过编译安装busybox就可以快速建立起一个基本的，可用的，轻量级的，linux文件系统（大小可以小到只有几百K）。在嵌入式领域制作自己的Linux文件系统，busybox是一个非常好用的工具。

内容：  
\*杂乱文档  
\*正确的实践步骤  
\*一些配置信息

\*杂乱文档  
===================  
官方网址：  
<http://www.busybox.net/>  
下载网址：  
<http://www.busybox.net/downloads/>

官方网址的README部分摘抄(也是源代码中的readme)：  
------------------------  
\*简介：  
busybox包含了许多的unix常用工具，相当于一个工具集合,例如:date,find等。  
它的体积非常的小,目的就是要用尽可能小的资源，它的模块化程度也非常高。可以用来构建嵌入式Linux文件系统（例如只需要再添加一个/dev,/etc,和一个linux内核）.  
它的可移植性非常强，遵从POSIX,在许多环境下，例如空间有限的嵌入式环境等都可以使用。  
\*使用：  
busybox高度可配置，你可以只编译自己需要的组件。使用"make config"或者"make menuconfig"来选择你需要的组件功能(使用"make help查看更多的帮助信息")。  
busybox的行为取决于它调用的程序名。将程序作为第二个参数来运行，例如"./busybox ls -l /proc".  
"standalone shell" 模式使用一个shell来调用内嵌的工具，可以非常方便地测试busybox.注意，如果从软盘或者从chroot环境来进行的话需要保证/proc被挂载上去。  
编译的时候会自动生成一个"busybox.links"文件，这个文件在"make install"创建编译进去的命令的busybox的二进制文件链接的时候会用到。  
使用PREFIX环境变量可以指定从哪里安装busybox的二进制文件和符号链接树（例如，'make PREFIX=/tmp/foo install'或者'make PREFIX=/tmp/foo install-hardlinks'，后者用于你喜欢硬链接的情况)。  
源代码的下载在：<http://busybox.net/downloads/>

install(源代码中的install)  
------------------------  
\*编译：  
编译过程和编译Linux内核的过程差不多,大体有配置编译选项，编译及安装三步：  
1)配置编译选项：  
$make menuconfig  
这将在当前目录下面生成一个".config"  
2)编译生成"busybox"可执行文件：  
$make  
3)安装：  
$make install  
这里也可以使用"make CONFIG\_PREFIX=/path/from/root install"来指定路径进行安装.

更多的信息可以参见：  
$make help

\*快速开始：  
第一次尝试busybox并且不用安装它的一个简单方式是:把它的所有特性都打开，使用"standalone shell"模式，将环境变量设置为空。  
打开所有的特性的方法是使用"make defconfig",这样会生成一个最为通用的配置(比allyesconfig 小一些):  
$make defconfig  
$make  
$PATH= ./busybox ash  
"standalone shell"模式导致busybox的内部shell可以直接运行busybox内嵌的应用程序,不用寻找外部的程序(???)。设置为空路径导致busybox之能够找到它内嵌的程序。  
注意：standalone shell 需要CONFIG\_BUSYBOX\_EXEC\_PATH被合适地配置，这取决于/proc/self/exe是否可用。如果你没有/proc那么把那个配置选项指定为你的busybox二进制文件路径，一般是/bin/busybox.

\*配置Busybox:  
busybox注重大小上的优化，将所有的功能打开会导致生成的可执行文件较大--静态链接会超过1M.为了节省空间，可以对busybox的程序进行按需配置。最小的配置会屏蔽所有的功能，生成的可执行文件只有4k，非常的小，但是没有任何作用。  
手动配置修改已的配置可以运行"make menuconfig"(如果系统没有ncurses可以使用"make config"来代替).常用的生成初始配置的方法是"make allnoconfig"(所有的功能都屏蔽了，之后你需要什么加什么)和"make defconfig"(所有的功能都被激活,之后你不需要什么就移走什么).如果运行menuconfig的时候没有已经存在的配置文件那么它会首先运行make defconfig来创建一个初始的配置文件作为起点。  
其他生成初始配置文件的方法还"make allbareconfig","make allyesconfig","make randconfig",不详细说了。  
配置Busybox会生成一个.config文件，可以把它保存起来用于以后的使用。运行"make oldconfig"会把一个旧版本的busybox的.config文件更新成为合适新版本的busybox的.config文件。

\*安装Busybox:  
busybox和其他可执行程序一样，是个单个的可执行程序。busybox使用程序名作为它的一个参数，这决定了busybox的行为。  
安装busybox包括为busybox中每一个被选中的程序创建busybox的二进制文件连接,并且确保这些符号连接在shell的环境变量PATH中.运行"make install"来创建这些符号链接或者"make install-hardlinks"创建硬链接。安装的过程会用到文件"busybox.links"(由make创建的)，文件包含配置选中的程序以及在哪里安装它们。(拷贝的是符号连接还是本身？)  
为busybox安装连接并不总是需要的.一个特殊的程序名"busybox"的第一个参数会使用它的第一个参数作为运行的程序行为（例如./busybox cat good）,如果运行busybox程序的时候没有任何参数，那么会列出一个所有可以使用的程序列表。  
(下面的没有看懂不会翻译)  
The standalone shell can also call busybox applets without links to busybox under other names in the filesystem.  You can also configure a standaone install capability into the busybox base applet, and then install such links at runtime with one of "busybox --install" (for hardlinks) or "busybox --install -s" (for symlinks).  
If you enabled the busybox shared library feature (libbusybox.so) and want to run tests without installing, set your LD\_LIBRARY\_PATH accordingly when running the executable:  
LD\_LIBRARY\_PATH=`pwd` ./busybox

\*在源代码树之外编译：  
默认的情况下，busybox会把编译的临时文件放在源代码树的下面。然而从一个只读的目录下编译或者从相同的源代码目录中进行多种配置的不同编译时，这就需要将临时文件放在其他的路径之下了。  
在源代码目录之外进行编译的时候，先进入一个空目录中，然后在那里配置busybox,如下：  
$make -f /path/to/source/Makefile defconfig  
$make  
$make install  
也可以在配置的时候，使用O=$BUILDPATH 选项（要使用绝对路径），如下：  
$make O=/some/empty/directory allyesconfig  
$cd /some/empty/directory  
$make  
$make CONFIG\_PREFIX=. install

\*更多的信息  
参见busybox FAQ,  
FAQ一般在 <http://www.busybox.net/FAQ.html>  
或者打包文件中的：docs/busybox.net/FAQ.html .

我的实践步骤1  
------------------------  
１挂载windows盘，并且进入:  
#mount /dev/sda2 /mnt  
#exit  
$cd /mnt  
２解压源代码：  
$tar -xjvf busybox-1.15.3.tar.bz2  
３建立目录结构：  
$mkdir busyboxSrc  
$mv busybox-1.15.3 busyboxSrc  
$cd busyboxSrc  
$mkdir build  
$mkdir target  
这里，源代码目录是busybox-1.15.3，编译的临时文件在目录build中，安装到target中。  
４开始编译:  
4.1)一次错误的尝试：  
$cd busybox-1.15.3  
$make O=/mnt/busyboxSrc/build defconfig  
$cd ../build  
$make  
这里，第２行是生成配置文件，这里就使用defconfig表示使用最普适的默认配置。并且使用-O指定配置文件以及编译生成的临时文件等都放在build目录中（要使用绝对路径）.最后进入build目录进行make编译。  
至此，make的时候报错，错误信息显示没有权限建立include2,于是我尝试不用windows盘，将/mnt改用/home/quietheart路径.  
4.2)再次尝试,这里是正确的编译：  
$mv busyboxSrc /home/quietheart/lvkTmp  
$cd /home/quietheart/lvkTmp/busyboxSrc/busybox-1.15.3  
$make O=/home/quietheart/lvkTmp/busyboxSrc/build/ defconfig  
$cd ../build  
$make  
这样，可以继续进行下去了。不到５分钟，编译完毕。已经在build中生成了可执行文件busybox.  
５安装：  
$make CONFIG\_PREFIX=../target install  
这样就将busybox装入到target中了,运行:$ls ../target输出如下：  
bin  linuxrc  sbin  usr  
target目录占用空间800k左右，总共不到1M.这里，安装的可执行文件就是bin/busybox.  
６使用：  
这里举个例子：  
假设使用busybox中的ls命令，方法如下：  
方法1):  
$./busybox ls  
方法2):  
$ln -s busybox ls  
$./ls  
这里，要么直接用busybox运行ls(方法1),要么建立一个名为"ls"的指向busybox的软链接再运行./ls(方法2)。实际在安装的时候，已经自动建立好了ls链接，不用我们手动建立了。

我的实践步骤2  
------------------------  
使用menuconfig进行配置。  
1)安装libncurses5-dev:  
$sudo apt-get install libncurses5-dev  
这样，才能支持命令行下面的图形彩色菜单。

\*正确的实践步骤  
===================  
最简单的安装过程：  
--------------  
1)解压源代码：  
$tar -xjvf busybox-1.15.3.tar.bz2  
这里，源代码和下载地址在如下网址可以找到：  
<http://www.busybox.net/>  
<http://www.busybox.net/downloads/>

2)建立目录结构：  
$mkdir /home/quietheart/lvkTmp/busyboxSrc  
$mv busybox-1.15.3 /home/quietheart/lvkTmp/busyboxSrc  
$cd /home/quietheart/lvkTmp/busyboxSrc  
$mkdir build  
$mkdir target  
这里，源代码目录是busybox-1.15.3，编译的临时文件在目录build中，安装目录是target。需要注意的是，编译的时候，不要在挂载的windows盘上进行，我实践过不行。

3)编译：  
$cd /home/quietheart/lvkTmp/busyboxSrc/busybox-1.15.3  
$make O=/home/quietheart/lvkTmp/busyboxSrc/build/ defconfig  
$cd ../build  
$make  
这样，不到５分钟，编译完毕。已经在build中生成了可执行文件busybox和其他相关文件.

4)安装：  
$make CONFIG\_PREFIX=../target install  
这样，就将busybox装入到target中了,运行:$ls ../target输出如下：  
bin  linuxrc  sbin  usr  
target目录占用空间800k左右，总共不到1M.这里，安装的可执行文件就是bin/busybox.

5)使用：  
这里举个例子：  
假设使用busybox中的ls命令，方法如下：  
方法1):  
$./busybox ls  
方法2):  
$ln -s busybox ls  
$./ls  
这里，要么直接用busybox运行ls(方法1),要么建立一个名为"ls"的指向busybox的软链接再运行./ls(方法2)。实际在安装的时候，已经自动建立好了ls链接，不用我们手动建立了。之后，我们可以把安装好的target目录打包，拷贝到其他需要用到的地方使用了。

\*一些配置信息  
===================  
下面个目录是很多人都会感兴趣的：  
Build Options --->  
Installation Options --->  
Login/Password Management Utilities --->  
在Build Options里面有是否使用交叉编译的选项（Do you want to build BusyBox with a Cross Compiler）。如果要对其他平台进行编译就要选择它并设置相应的编译程序前缀。  
在Installation Options里面可以设置安装的路径,即设置为新文件系统的根目录。当然也可以用缺省的 \_install目录,安装之后再复制到新文件系统中去。  
在设置Login/Password Management Utilities的时候,为了免去配置glibc的麻烦,最好设置使用busybox自己的password和shadow文件的功能（Use internal password and group functions rather than system functions）。  
busybox 提供的命令无疑比较全面,但它的缺省配置还不能构成一个功能比较完备的文件系统。必须要添加的命令有Login/Password Management Utilities里面的getty、login和passwd,这些命令从前是由另外一个软件包tinylogin所提供的。当然,如果不需要一个交互的登录界面,这些命令也可以去掉。通过设定内核参数init=/bin/sh可以在系统启动busybox的编译配置界面后直接得到一个shell。  
tlenetd命令是一个远程登录服务程序,把它编译进busybox中会为将来的调试提供很多的方便。当然也可以通过串口终端的方式管理嵌入式设备,但网络登录的方式却更为方便。

本文是工作过程中的一些经验总结，如果有问题或者建议，可以联系我。  
作者：QuietHeart  
Email：[quiet\_heart000@126.com](mailto:quiet_heart000@126.com)