|  |
| --- |
| Neusoft |
| 编译环境搭建 |
|  |
|  |
| **lv-k@neusoft.com** |
| **2010-11-16** |

|  |
| --- |
| 本文主要叙述编译环境搭建，以及编译方法。 |

目录

[一、代码路径 3](#_Toc277675053)

[二、svn基本使用(非图形) 3](#_Toc277675054)

[1,检出代码： 3](#_Toc277675055)

[2,提交之前检测： 3](#_Toc277675056)

[3,提交修改： 3](#_Toc277675057)

[4,查看从前提交的log： 4](#_Toc277675058)

[5,最新版本取出： 4](#_Toc277675059)

[6,取出以前版本： 4](#_Toc277675060)

[7,添加/删除文件： 4](#_Toc277675061)

[8.查看版本库管理得文件： 4](#_Toc277675062)

[9,查看当前路径工作拷贝得全局信息： 4](#_Toc277675063)

[10,撤消修改： 5](#_Toc277675064)

[注意 5](#_Toc277675065)

[三、编译相关 5](#_Toc277675066)

[1,环境搭建： 5](#_Toc277675067)

[1.1)centeros 安装. 5](#_Toc277675068)

[1.2)yum配置： 6](#_Toc277675069)

[1.3)安装编译环境： 6](#_Toc277675070)

[2,代码编译： 7](#_Toc277675071)

[2.1)开发时： 7](#_Toc277675072)

[2.2)release时： 7](#_Toc277675073)

[3,运行方法： 8](#_Toc277675074)

[四、添加一个新项目得方法 8](#_Toc277675075)

[五、补充 9](#_Toc277675076)

# 一、代码路径

http://10.1.29.125/svn/sony/trunk/Source/

应该使用svn对它们检出。

http://10.1.29.125/svn/sony/trunk/Source/Rootfs/diablo/diabloMainCpu

这个是本次base release时候修改得svn目录。

smb://10.1.29.125/SonyFS/src/MediaBlock\_MXFS6\_Release\_20100727-01.zip

svn代码就是基于这个路径得代码修改得,当前已经将最新代码合并到svn里面了。

# 二、svn基本使用(非图形)

## 1,检出代码：

$svn checkout http://lv-k@10.1.29.125/svn/sony/trunk/Source/

检出之后，本地就有了一个工作拷贝，可以在此基础上进行修改,修改之后可以把本地修改得部分提交(见后面)到svn服务器上供他人更新(见后面)使用。

检出之时，会提示两次输入用户密码，一次是输入你本地机器的用户名密码，一次是服务器得。

## 2,提交之前检测：

$svn status

提交之前，可以先检测一下，看自己修改,删除(svn delete)，添加(svn add)了哪些文件，哪些文件没有添加到版本控制，以及做了什么，这个命令输出得含义参考"svn help status",便于在提交之前确认本次提交操作得正确性。

## 3,提交修改：

$svn commit -m "<你要添加得说明>"

这里，"<你要添加得说明>"将会在后面提到得"svn log"命令输出。

## 4,查看从前提交的log：

$svn log

这样，会输出每次提交得日志信息。

## 5,最新版本取出：

$svn update

这一步一般在每次工作（即修改本地工作拷贝）之前，保证自己得修改在最新得版本基础上进行。

## 6,取出以前版本：

$svn update -r <version>

这里，将会取出以前第"<version>"版得工作拷贝，具体<version>是多少，需要查看"svn log"来确定。

## 7,添加/删除文件：

$svn add <filename>

$svn delete <filename>

如果我们想要让svn管理新建立得文件，需用"svn add",反之用"svn delete",这两个命令执行完毕之后，需要"svn commit"才能反应到服务器上。

这里，没有add得文件，不会在"svn commit"得时候被提交，可以通过'svn status'查看这个文件是否被add过。

## 8.查看版本库管理得文件：

$svn ls

## 9,查看当前路径工作拷贝得全局信息：

$svn info

这条命令可以显示当前路径得工作拷贝在服务器上的检出路径，以及最新修改等信息.

## 10,撤消修改：

$svn revert

如果提交之前，发现当前修改错了，或者添加/删除文件错了，可以运行这个命令，重新对本地工作拷贝进行修改。

## 注意

以上命令一般都是递归的在当前目录进行，不影响父目录其它文件和目录。以上命令一般可以再添加一个文件/目录参数，对指定得文件/目录进行管理。

使用svn，一般得工作周期是：

1)更新本地工作拷贝

$svn update

2)修改代码

3)添加新增文件(如果需要得话)

$svn add/delete

4)检测修改状态

$svn status

5)提交修改

$svn commit -m ""

更多svn的信息参见http://i18n-zh.googlecode.com/svn-history/r734/www/svnbook-1.4/index.html

# 三、编译相关

## 1,环境搭建：

### 1.1)centeros 安装.

（省略）

### 1.2)yum配置：

$sudo vi /etc/yum.conf

添加如下行：

Proxy=http://user:passwd@dl-proxy.neusoft.com:8080/

$sudo yum update

### 1.3)安装编译环境：

#### 1.3.1)安装交叉编译工具链

smb://10.1.29.125/share/ALL/SONY-CE-Linux-5.0.4.1-ARM.iso

其中install\_en.txt里面有相关得文档。

#### 1.3.2)编译内核

为了编译内核添加如下路径到环境变量：

/usr/local/arm-sony-linux-gnueabi/devel/bin/

然后下载smb://10.1.29.125/SonyFS/Documents/TechnologyDoc/CELinux/CELinux/diablo/20100326/linux-5\_0\_DIABLO\_20100326-src-kernel.tar.gz

解压之后，

依次运行：

./setup-avp13

make

make kernel\_headers

sudo make kernel\_headers\_install

这样就可以编译应用程序了。

为了编译应用程序，应该添加如下路径到环境变量：

/usr/local/arm-sony-linux-gnueabi/bin

注意：编译内核相关得环境变量添加“/usr/local/arm-sony-linux-gnueabi/devel/bin/“，编译app的环境变量添加“/usr/local/arm-sony-linux-gnueabi/bin”。

为了方便开发，不用每次启动shell时手动添加，可以添加如下到/etc/bashrc中：

#编译内核时：

export PATH=/usr/local/arm-sony-linux-gnueabi/devel/bin/:$PATH

#编译 app时：

export PATH=/usr/local/arm-sony-linux-gnueabi/bin:$PATH

## 2,代码编译：

### 2.1)开发时：

每次修改完代码，编译过程如下：

$cd build

$./build.sh

这样会根据时间戳编译最新得内容。在第一次编译前或者增删文件时需要运行./build.sh depend，参照"四、添加一个新项目得方法"

编译得全部log信息存放在build\_Diablo.log

编译如果出错，那么错误信息存放在error.log

上次得编译log以及编译错误分别存放在：build\_Diablo.old.log和error.old.log

清除之前的编译中间文件：

./build.sh clean

### 2.2)release时：

\*编译内核：

$svn checkout http://lv-k@10.1.29.125/svn/sony/trunk/Source/Kernel

$cd Avp13\_Kernel

$ make avp13\_defconfig

$ make

注意，

\*编译内核之前，环境变量设置为"export PATH=/usr/local/arm-sony-linux-gnueabi/devel/bin/:$PATH"

\*生成的内核avp13Build/vmlinux.bin

\*编译app以及生成文件系统镜像：

$svn checkout http://lv-k@10.1.29.125/svn/sony/trunk/Source/Rootfs

$cd For\_Release

$./neu\_release.sh all

注意，

\*运行之前，环境变量设置为"export PATH=/usr/local/arm-sony-linux-gnueabi/bin:$PATH"

\*生成的镜像名称为target/\*.img

\*对于该脚本具体信息，最好参考下For\_Release/readme

## 3,运行方法：

开发时最好用nfs来挂接文件系统，然后进行运行测试，在其它文档中会有详细说明。

大致如下：

编译好的文件在build/diablo.bin下，

需要拷贝diablo.bin到文件系统下，

然后再把osWrapper中得相关库文件拷贝到文件系统下，

然后把新拷贝得osWrapper库路径添加到环境变量LD\_LIBRARY\_PATH中。

运行./diablo.bin

# 四、添加一个新项目得方法

举例如下：

假设我要添加一个User\_test目录，其中包含test.h test.c,那么：

1)编辑build/incpaths.mk文件

1.1)找到"# Main DIR"注释处，添加定义新目录对应变量的语句如下：

USER\_TEST= $(SRCROOT)/User\_test

1.2)找到"ALL\_PATHS"变量得定义，在其最后添加刚才得新变量如下：

############

$(USER\_TEST) \

2)新建User\_test/usertest.mk文件，内容如下：

USER\_TEST\_SOURCES= \

test.c

USER\_TEST\_OBJS := $(addprefix $(TARGETDIR)/, $(USER\_TEST\_SOURCES:%.c=%.o))

USER\_TEST\_DEPS := $(addprefix $(TARGETDIR)/, $(USER\_TEST\_SOURCES:%.c=%.dep))

vpath %.c $(USER\_TEST)

############

3)编辑build/diablo\_objects.mk文件，在最后添加如下：

include $(USER\_TEST)/usertest.mk

4)编辑Diablo.make,在OBJECTS变量最后追加一条：

$(USER\_TEST\_OBJS) \

5)重新生成依赖：

./build.sh clean

./build.sh depend

6)编译：

./build.sh

可参见上面。

具体参见build.sh脚本。

# 五、补充

能够进行x86版本编译和调试的diabloMainCpu代码已经更新到SVN，

使用方法如下：

编辑diabloMainCpu/build/build.sh

PLATFORMPARAM="USE\_ARM=yes" //编译ARM版本

PLATFORMPARAM="USE\_ARM=no" //x86 debug版本

这两个版本编译的结果都是diablo.bin文件。

以上,有问题请联系:

Email:lv-k@neusoft.com