OpenWrt Wiki

创建软件包

我们已经尝试做的一件事情，是让移植软件到OpenWrt的操作变得非常容易。如果打开OpenWrt里的一个软件包的目录(OpenWrt/Package/* 或 OpenWrt/feeds/packages/*/*)，通常会发现几样东西：

- package/Makefile [必备]
- package/patches/ [可选]
- package/files/ [可选]

patches目录和files目录都是可选的，pactches目录通常包括bug修复和对可执行文件体积的优化，files目录通常包括配置文件。你也可能看到其它目录，因为只要在Makefile文件中指明，目录名字是可以任取的。前面两个是约定俗成的做法，强烈建议你也这么做。

Makefile文件最关键，一般来说它提供了下载、编译、安装这个软件包的步骤。

当我们打开这里的Makefile文件，很难认出这是一个Makefile。它的格式跟一般的Makefile不一样，因为它的功能跟普通Makefile就是不一样的。它是一种编写方便的模板。

这里，以package/bridge/Makefile文件为例：

```
include $(TOPDIR)/rules.mk

PKG_NAME:=bridge
PKG_VERSION:=1.0.6
PKG_RELEASE:=1

PKG_BUILD_DIR:=$(BUILD_DIR)/bridge-utils-$(PKG_VERSION)
PKG_SOURCE:=bridge-utils-$(PKG_VERSION).tar.gz
PKG_SOURCE_URL:=@SF/bridge
PKG_MD5SUM:=9b7dc52656f5cbec846a7ba3299f73bd
PKG_CAT:=zcat

include $(INCLUDE_DIR)/package.mk

define Package/bridge
  SECTION:=base
  CATEGORY:=Network
  TITLE:=Ethernet bridging configuration utility
  #DESCRIPTION:=This variable is obsolete. use the Package/name/description define instead!
  URL:=http://bridge.sourceforge.net/
endef

define Package/bridge/description
 Ethernet bridging configuration utility
 Manage ethernet bridging; a way to connect networks together to
 form a larger network.
endef

define Build/Configure
  $(call Build/Configure/Default,--with-linux-headers=$(LINUX_DIR))
endef

define Package/bridge/install
  $(INSTALL_DIR) $(1)/usr/sbin
  $(INSTALL_BIN) $(PKG_BUILD_DIR)/brctl/brctl $(1)/usr/sbin/
endef

$(eval $(call BuildPackage,bridge))
```

软件包变量

建立一个软件包不需要太多工作；大部分工作都隐藏在其它的 makefiles 中，编写工作被抽象成对几个变量的赋值。

- PKG_NAME -软件包的名字, 在 menuconfig 和 ipkg 显示
- PKG_VERSION -软件包的版本，主干分支的版本正是我们要下载的
- PKG_RELEASE -这个 makefile 的版本
- PKG_BUILD_DIR -编译软件包的目录
- PKG_SOURCE -要下载的软件包的名字，一般是由 PKG_NAME 和 PKG_VERSION 组成
- PKG_SOURCE_URL -下载这个软件包的链接

- PKG_MD5SUM -软件包的 MD5 值
- PKG_CAT -解压软件包的方法 (zcat, bzcat, unzip)
- PKG_BUILD_DEPENDS -需要预先构建的软件包，但只是在构建本软件包时，而不是运行的时候。它的语法和下面的DEPENDS一样。

PKG_*变量定义了从何处下载这个软件包；@SF是表示从sourceforge网站下载的一个特殊关键字。md5sum用来检查从网上下载的软件包是否完好无损。PKG_BUILD_DIR定义了软件包源代码的解压路径。

注意到上面示例文件底部的最后一行吗？这是最为关键的BuildPackage宏。它是在$(INCLUDE_DIR)/package.mk文件里定义的。BuildPackage宏只要求一个参数，即要编译的软件包名，在本例中是"bridge"。所有其他信息都通过宏来获得，这提供了一种内在的简洁性。比如BuildPackage需要软件包的一大串描述信息，我们并不要向它传递冗长的参数，因为我们已经约定描述信息定义在DESCRIPTION宏，BuildPackage从里面读取就可以了。

BuildPackage相关的宏

Package/

描述软件包在menuconfig和ipkg中的信息，可以定义如下变量：

- SECTION - 软件包类型 (尚未使用)
- CATEGORY - menuconfig中软件包所属的一级目录，如Network
- SUBMENU - menuconfig中软件包所属的二级目录，如dial-in/up
- TITLE - 软件包标题
- DESCRIPTION - 软件包的详细说明
- URL - 软件的原始位置，一般是软件作者的主页
- MAINTAINER - (optional) 软件包维护人员
- DEPENDS - (optional) 依赖项，运行本软件依赖的其他包

Package/conffiles (可选)

软件包需要复制的配置文件列表，一个文件占一行

Build/Prepare (可选)

一组解包源代码和打补丁的命令，一般不需要。

Build/Configure (可选)

如果源代码编译前需要configure且指定一些参数，就把这些参数放在这儿。否则可以不定义。

Build/Compile (可选)

编译源代码命令。

Package/install

软件安装命令，主要是把相关文件拷贝到指定目录，如配置文件。

Package/preinst

软件安装之前被执行的脚本，别忘了在第一句加上#!/bin/sh。如果脚本执行完毕要取消安装过程，直接让它返回false即可。

Package/postinst

软件安装之后被执行的脚本，别忘了在第一句加上#!/bin/sh。

Package/prerm

软件删除之前被执行的脚本，别忘了在第一句加上#!/bin/sh。如果脚本执行完毕要取消删除过程，直接让它返回false即可。

Package/postrm

软件删除之后被执行的脚本，别忘了在第一句加上#!/bin/sh。

为什么一些定义是"Package/"前缀，另一些定义却是"Build"前缀？这是因为我们支持一个特性：从单个源代码构建多个软件包。OpenWrt工作在一个Makefile对应一个源代码的假设之上，但是你可以把编译生成的程序分割成任意多个软件包。因为编译只要一次，所以使用全局的"Build"定义是最合适的。然后你可以增加很多"Package/"定义，为各软件包分别指定安装方法。建议你去看看dropbear包，这是一个很好的示范。

提示：对于所有在pre/post, install/removal脚本中使用的变量，都应该使用"$$"代替"$"。这是告诉make暂时不要解析这个变量，而是把它当成普通字符串以及用"$"代替"$$"。 – 更多信息 [https://forum.openwrt.org/viewtopic.php?pid=85197#p85197]

在编辑好Makefile文件，并放到指定目录后，这个新的软件包将在下次执行make menuconfig时出现，你可以选择这个软件包，保存退出，用make编译。现在就把一个软件成功移植到OpenWrt中了！

创建内核模块软件包

内核模块 [http://www.digitalhermit.com/linux/Kernel-Build-HOWTO.html]是一类扩展Linux内核功能的可安装程序。内核模块的加载发生在内核加载之后，（比如使用insmod命令）。

Linux源代码包含了很多内核应用程序。在menuconfig时有3种选择，

1. 编译进内核；
2. 编译成可加载的内核模块；
3. 不编译。

参看***FIX:Customizingthekerneloptions customizing the kernel options***

要包含这些内核模块，只要make menuconfig选择相应的内核模块选项。（参看Build Configuration）。如果在make menuconfig时没有发现想要的内核模块，必须添加一个stanza到package/kernel/modules目录的一个文件中。下面是一个从package/kernel/modules/other.mk提取出来的例子。

```
define KernelPackage/loop
  TITLE:=Loopback device support
  DESCRIPTION:=Kernel module for loopback device support
  KCONFIG:=$(CONFIG_BLK_DEV_LOOP)
  SUBMENU:=$(EMENU)
  AUTOLOAD:=$(call AutoLoad,30,loop)
  FILES:=$(MODULES_DIR)/kernel/drivers/block/loop.$(LINUX_KMOD_SUFFIX)
endef
$(eval $(call KernelPackage,loop))
```

此外，你还可以添加不属于Linux源码包的内核模块。在这种情况下，内核模块放在package/目录，跟通常的软件包是一样的。package/Makefile文件使用KernelPackage/xxx定义代替Package/xxx。

这是package/madwifi/Makefile的例子：

```
#
# Copyright (C) 2006 OpenWrt.org
#
# This is free software, licensed under the GNU General Public License v2.
# See /LICENSE for more information.
#
# $Id$

include $(TOPDIR)/rules.mk
include $(INCLUDE_DIR)/kernel.mk

PKG_NAME:=madwifi
PKG_VERSION:=0.9.2
PKG_RELEASE:=1

PKG_SOURCE:=$(PKG_NAME)-$(PKG_VERSION).tar.bz2
PKG_SOURCE_URL:=@SF/$(PKG_NAME)
PKG_MD5SUM:=a75baacbe07085ddc5cb28e1fb43edbb
PKG_CAT:=bzcat

PKG_BUILD_DIR:=$(KERNEL_BUILD_DIR)/$(PKG_NAME)-$(PKG_VERSION)

include $(INCLUDE_DIR)/package.mk

RATE_CONTROL:=sample

ifeq ($(ARCH),mips)
  HAL_TARGET:=mips-be-elf
endif
ifeq ($(ARCH),mipsel)
  HAL_TARGET:=mips-le-elf
endif
ifeq ($(ARCH),i386)
  HAL_TARGET:=i386-elf
endif
ifeq ($(ARCH),armeb)
  HAL_TARGET:=xscale-be-elf
endif
ifeq ($(ARCH),powerpc)
  HAL_TARGET:=powerpc-be-elf
endif

BUS:=PCI
ifneq ($(CONFIG_LINUX_2_4_AR531X),)
  BUS:=AHB
endif
ifneq ($(CONFIG_LINUX_2_6_ARUBA),)
  BUS:=PCI AHB  # no suitable HAL for AHB yet.
endif

BUS_MODULES:=
ifeq ($(findstring AHB,$(BUS)),AHB)
```

```
  BUS_MODULES+=$(PKG_BUILD_DIR)/ath/ath_ahb.$(LINUX_KMOD_SUFFIX)
endif
ifeq ($(findstring PCI,$(BUS)),PCI)
  BUS_MODULES+=$(PKG_BUILD_DIR)/ath/ath_pci.$(LINUX_KMOD_SUFFIX)
endif

MADWIFI_AUTOLOAD:= \
        wlan \
        wlan_scan_ap \
        wlan_scan_sta \
        ath_hal \
        ath_rate_$(RATE_CONTROL) \
        wlan_acl \
        wlan_ccmp \
        wlan_tkip \
        wlan_wep \
        wlan_xauth

ifeq ($(findstring AHB,$(BUS)),AHB)
        MADWIFI_AUTOLOAD += ath_ahb
endif
ifeq ($(findstring PCI,$(BUS)),PCI)
        MADWIFI_AUTOLOAD += ath_pci
endif

define KernelPackage/madwifi
  SUBMENU:=Wireless Drivers
  DEFAULT:=y if LINUX_2_6_BRCM |  LINUX_2_6_ARUBA |  LINUX_2_4_AR531X |  LINUX_2_6_XSCALE, m if ALL
  TITLE:=Driver for Atheros wireless chipsets
  DESCRIPTION:=\
        This package contains a driver for Atheros 802.11a/b/g chipsets.
  URL:=http://madwifi.org/
  VERSION:=$(LINUX_VERSION)+$(PKG_VERSION)-$(BOARD)-$(PKG_RELEASE)
  FILES:= \
                $(PKG_BUILD_DIR)/ath/ath_hal.$(LINUX_KMOD_SUFFIX) \
                $(BUS_MODULES) \
                $(PKG_BUILD_DIR)/ath_rate/$(RATE_CONTROL)/ath_rate_$(RATE_CONTROL).$(LINUX_KMOD_SUFFIX) \
                $(PKG_BUILD_DIR)/net80211/wlan*.$(LINUX_KMOD_SUFFIX)
  AUTOLOAD:=$(call AutoLoad,50,$(MADWIFI_AUTOLOAD))
endef

MADWIFI_MAKEOPTS= -C $(PKG_BUILD_DIR) \
                PATH="$(TARGET_PATH)" \
                ARCH="$(LINUX_KARCH)" \
                CROSS_COMPILE="$(TARGET_CROSS)" \
                TARGET="$(HAL_TARGET)" \
                TOOLPREFIX="$(KERNEL_CROSS)" \
                TOOLPATH="$(KERNEL_CROSS)" \
                KERNELPATH="$(LINUX_DIR)" \
                LDOPTS=" " \
                ATH_RATE="ath_rate/$(RATE_CONTROL)" \
                DOMULTI=1

ifeq ($(findstring AHB,$(BUS)),AHB)
  define Build/Compile/ahb
        $(MAKE) $(MADWIFI_MAKEOPTS) BUS="AHB" all
  endef
endif

ifeq ($(findstring PCI,$(BUS)),PCI)
  define Build/Compile/pci
        $(MAKE) $(MADWIFI_MAKEOPTS) BUS="PCI" all
  endef
endif

define Build/Compile
        $(call Build/Compile/ahb)
        $(call Build/Compile/pci)
endef

define Build/InstallDev
        $(INSTALL_DIR) $(STAGING_DIR)/usr/include/madwifi
        $(CP) $(PKG_BUILD_DIR)/include $(STAGING_DIR)/usr/include/madwifi/
        $(INSTALL_DIR) $(STAGING_DIR)/usr/include/madwifi/net80211
        $(CP) $(PKG_BUILD_DIR)/net80211/*.h $(STAGING_DIR)/usr/include/madwifi/net80211/
endef

define KernelPackage/madwifi/install
        $(INSTALL_DIR) $(1)/etc/init.d
        $(INSTALL_DIR) $(1)/lib/modules/$(LINUX_VERSION)
        $(INSTALL_DIR) $(1)/usr/sbin
        $(INSTALL_BIN) ./files/madwifi.init $(1)/etc/init.d/madwifi
        $(CP) $(PKG_BUILD_DIR)/tools/{madwifi_multi,80211debug,80211stats,athchans,athctrl,athdebug,athkey,athstats,wlanconfig} $(1)/usr/sbin/
endef

$(eval $(call KernelPackage,madwifi))
```

zh-cn/doc/devel/packages.txt · Last modified: 2015/05/18 08:34 by paul.cpp