**u-boot 2010.03 mkconfig文件分析**

[u-boot](http://linchunai1212.blog.163.com/blog/#m=0&t=1&c=fks_084067087084087068080080094095087083080066086087082064) 2010-07-09 13:54:35 阅读94 评论1  字号：大**中**小 订阅

**在u-boot执行make之前，会完成配置过程，make    xxx\_config   这个目标是调用uboot根目录下的mkconfig文件。同时给他传递六个变量，以make smdk2410\_config为例，这六个变量为smdk2410 arm arm920t smdk2410 NULL s3c24x0**  
  
   #!/bin/sh -e  
**\*\*\*\*此处-e  不知道什么意思？**  
# Script to create header files and links to configure  
# U-Boot for a specific board.  
#  
# Parameters:  Target  Architecture  CPU  Board [VENDOR] [SOC]  
#**\*\*\*\*此处指出了脚本的参数**  
# (C) 2002-2006 DENX Software Engineering, Wolfgang Denk <wd@denx.de>  
#  
  
APPEND=no    # Default: Create new config file  
BOARD\_NAME=""    # Name to print in make output  
TARGETS=""  
**\*\*\*\*定义了三个变量**  
while [ $# -gt 0 ] ; do  
**\*\*\*\*$#表示输入进来的参数个数，-gt（great than）表示左边大于右边则返回真  
SHELL常用内部参数：　　  
 $# ----传递给程序的总的参数数目  
　　 $? ----上一个代码或者shell程序在shell中退出的情况，如果正常退出则返回0，反之为非0值。  
　　 $\* ----传递给程序的所有参数组成的字符串。  
      $n ----表示第几个参数，$1 表示第一个参数，$2 表示第二个参数 ...   
      $0 ----当前程序的名称  
      $@----以"参数1" "参数2" ... 形式保存所有参数  
      $$ ----本程序的(进程ID号)PID  
      $!  ----上一个命令的PID**   
    case "$1" in  
    --) shift ; break ;;  
    -a) shift ; APPEND=yes ;;  
    -n) shift ; BOARD\_NAME="${1%%\_config}" ; shift ;;  
    -t) shift ; TARGETS="`echo $1 | sed 's:\_: :g'` ${TARGETS}" ; shift ;;  
    \*)  break ;;  
    esac  
done  
**\*\*\*\*\*由于mkconfig没有这些参数，所以上面语句不执行**  
[ "${BOARD\_NAME}" ] || BOARD\_NAME="$1"  
**\*\*\*\*\*此处很重要，他给BOARD\_NAME赋值为$1，即smdk2410**  
[ $# -lt 4 ] && exit 1  
**\*\*\*\* -lt是小于的意思，-le是小于等于的意思，此处用于判断参数的个数是否符合要求**  
[ $# -gt 6 ] && exit 1  
**\*\*\*\* @上面是处理第一个参数的过程最后BOARD\_NAME=smdk2410**  
if [ "${ARCH}" -a "${ARCH}" != "$2" ]; then  
    echo "Failed: \$ARCH=${ARCH}, should be '$2' for ${BOARD\_NAME}" 1>&2  
    exit 1  
fi  
**\*\*\*\* 判断ARCH是否等于参数2:arm.不是就 推出，并提示失败。此处-a不知道什么意思？**  
  
echo "Configuring for ${BOARD\_NAME} board..."  
  
#  
# Create link to architecture specific headers  
#  
if [ "$SRCTREE" != "$OBJTREE" ] ; then  
**\*\*\*\*判断是不是目标文件和源文件一致，可以选择在别的文件建立编译过程**  
    mkdir -p ${OBJTREE}/include  
    mkdir -p ${OBJTREE}/include2  
    cd ${OBJTREE}/include2  
    rm -f asm  
**\*\*\*\* 删除上次配置产生的链接文件**  
    ln -s ${SRCTREE}/include/asm-$2  asm  
**\*\*\*\* 再次建立链接文件，并让它指向asm-$2，即asm --> asm-arm**  
    LNPREFIX="../../include2/asm/"  
    cd ../include  
    rm -rf asm-$2  
    rm -f asm  
    mkdir asm-$2  
    ln -s asm-$2 asm  
else  
    cd ./include  
    rm -f asm  
    ln -s asm-$2 asm  
fi  
**\*\*\*\* 再次建立链接文件，并让它指向asm-$2， 即asm --> asm-arm**  
rm -f asm-$2/arch  
**\*\*\*\*删除asm-$2/arch，即asm-arm/arch**  
if [ -z "$6" -o "$6" = "NULL" ] ; then  
**\*\*\*\*判断$6是否为空，或等于NULL，此处为假，所以执行else**  
    ln -s ${LNPREFIX}arch-$3 asm-$2/arch  
else  
    ln -s ${LNPREFIX}arch-$6 asm-$2/arch  
**\*\*\*\*由前面定义可知，如果在源文件中编译，则LNPREFIX=“”，所以此处实质为ln -s arch-s3c24x0 asm-arm/arch**  
fi  
  
if [ "$2" = "arm" ] ; then  
    rm -f asm-$2/proc  
    ln -s ${LNPREFIX}proc-armv asm-$2/proc  
fi  
**\*\*\*\*重建sam-arm/proc文件，并让它连接到proc-armv**  
#  
# Create include file for Make  
#  
echo "ARCH   = $2" >  config.mk  
echo "CPU    = $3" >> config.mk  
echo "BOARD  = $4" >> config.mk  
  
[ "$5" ] && [ "$5" != "NULL" ] && echo "VENDOR = $5" >> config.mk  
  
[ "$6" ] && [ "$6" != "NULL" ] && echo "SOC    = $6" >> config.mk  
**\*\*\*\*创建顶层包含的include/config.mk文件。其内容很简单。此处>是重定向，>>也是重定向，只是追加在文件尾部。往config.mk里面写入：ARCH = arm，CPU = arm920t，BOARD = smdk2410,SOC = s3c24x0  
你可以在顶层目录输入make smdk2410\_config,在include目录里面多了一个config.mk文 件，里面内容就是刚刚写入的内容**  
# Assign board directory to BOARDIR variable  
if [ -z "$5" -o "$5" = "NULL" ] ; then  
    BOARDDIR=$4  
else  
    BOARDDIR=$5/$4  
fi  
**\*\*\*\*BOARDDIR的定义，这在主控makefile是用到的  
#**  
# Create board specific header file  
#  
if [ "$APPEND" = "yes" ]    # Append to existing config file  
then  
    echo >> config.h  
else  
    > config.h        # Create new config file  
**\*\*\*\*这是创建空文档的一种方法**  
fi  
echo "/\* Automatically generated - do not edit \*/" >>config.h  
**\*\*\*\*创建开发板相 关的头文件  
/board/samsung/smdk2410/lowlevel\_init.S  
开头有条 语句：#include <config.h>**  
for i in ${TARGETS} ; do  
    echo "#define CONFIG\_MK\_${i} 1" >>config.h ;  
done  
**\*\*\*\*此处TARGETS为空，不执行**  
cat << EOF >> config.h  
**\*\*\*\*EOF只是一个分界符，其实你完全可以用abcde替换，也一样的功能，只是大家都习惯用EOF来表示。当s h e l l看到< <的时候，它就会知道下一个词是一个分界符。在该分界符以后的内容都被当作输入，直到s h e l l又看到该分界符(位于单独的一行)。这个分界符可以是你所定义的任何字符串。**  
#define CONFIG\_BOARDDIR board/$BOARDDIR  
#include <config\_defaults.h>  
#include <configs/$1.h>  
#include <asm/config.h>  
EOF  
**\*\*\*\*写入四个文件，不同版本的uboot的mkconfig的细节是有区别的。此处就是一例。**  
exit 0  
  
**总结mkconfig做的工作  
  a. 25 [ "${BOARD\_NAME}" ] || BOARD\_NAME = "$1",结果    BOARD\_NAME=smdk2410.  
  b. 在include目录下，建立链接：asm --> asm-arm，include/asm-arm目录下，arch --> arch-s3c24x0,  
proc --> proc-armv  
  c.在include目录下，建立config.mk文件，往里面写入：  
     ARCH  = arm  
     CPU   = arm920t  
     BOARD = smdk2410  
     SOC   = s3c24x0  
  d.在include目录下，建立头文件config.h.往里面写入：  
     /\* Automatically generated - do not edit \*/  
     #define CONFIG\_BOARDDIR board/$BOARDDIR  
    #include <config\_defaults.h>  
    #include <configs/$1.h>  
    #include <asm/config.h>**