MediaTek Linux SDK Developer Guide

Version: **1.0**
Release date: **2011/12/22**

Document Revision History

Revision	Date	Author	Description
Draft	*2010/9/15*	*xyyou*	*Initial draft*
	2011/12/22	*fredli*	*Release SDK*
	2015/4/15	*Guanhu.zou*	*Release SDK*

Table of Contents

1 Introduction

This SDK is the Linux platform for MediaTek CPE and it supports TC3162u, TC3182,RT63260,RT65168 and RT63365 concurrently .

This document is provided for RD that develop the advanced features with MediaTek Linux SDK. Developers can learn the detail of Build Framework architecture and how to choose the profile&feature to build a correct image in different platform. They can also easily add a new application according to the following step.

2 The Architecture of Build Framework

The whole source code structure is shown as below. And the framework contains kernel, driver, user application and configuration files, which locates different directories.

Releasebsp

Makefile		Top-level
linux		Linux Kernel code
bootrom		bootloader source and binary
modules		public and private driver module files.
	private	Linux Kernel code
	public	GPL licensed driver
apps		User application code
	private	Application public source code
	public	Application private source cod
Project		profile and config file for Project build
	config	Config related files
	customer_release	For customer release files
	image	Hold the built images
	profile	Files with configuration
	lib	Libs of toolchain
tools		Tools for buildimage
filesystem		files and folders for rootfs of tclinux
doc		Development document and manual

3 Prepare the profile

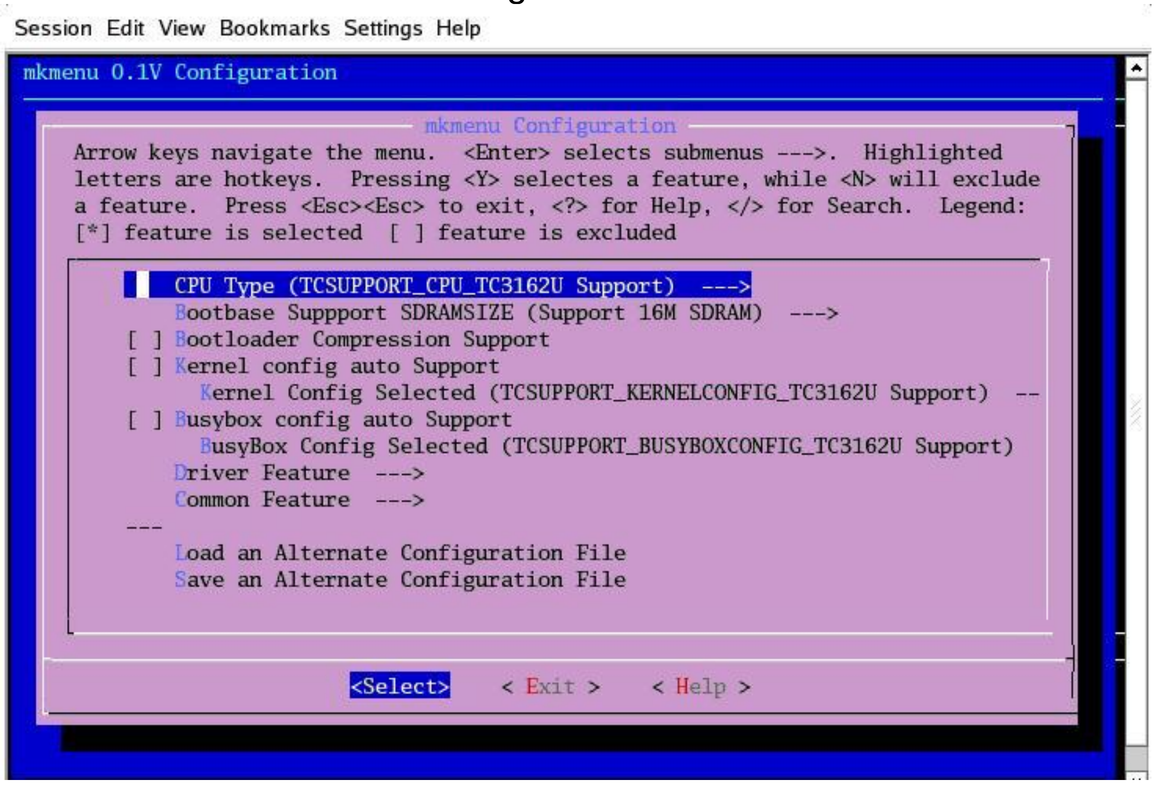
Uncompress SDK with command “tar -zxvf releasebsp_profilename_releasedate.tgz” into “releasebsp” folder. (the “releasebsp” represents location that you extract the SDK package to, the following the same)

3.1 Create a new profile

Change to dir“releasebsp/Project/profile” , create a folder and empty file under this folder with the same name. For instance, “mkdir test & touch test/test.profile ”

3.2 Configure a profile

Step1: Change to dir “releasebsp/” , configure your target profile with “make PROFILE=[profile to be configure] menuconfig”. For instance, “make PROFILE=test menuconfig”



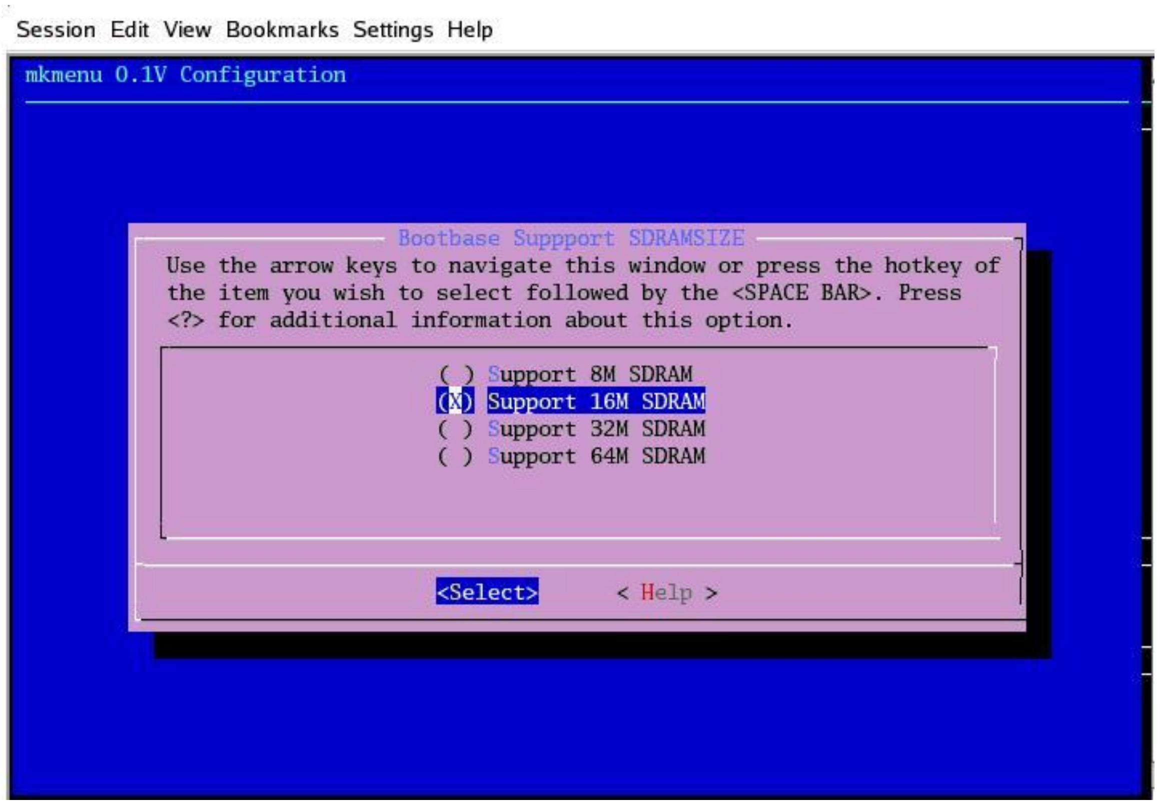
```
mkmenu 0.1V Configuration

                         mkmenu Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted
  letters are hotkeys.  Pressing <Y> selectes a feature, while <N> will exclude
  a feature.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend:
  [*] feature is selected  [ ] feature is excluded

        CPU Type (TCSUPPORT_CPU_TC3162U Support)  --->
        Bootbase Suppport SDRAMSIZE (Support 16M SDRAM)  --->
    [ ] Bootloader Compression Support
    [ ] Kernel config auto Support
          Kernel Config Selected (TCSUPPORT_KERNELCONFIG_TC3162U Support)  --
    [ ] Busybox config auto Support
          BusyBox Config Selected (TCSUPPORT_BUSYBOXCONFIG_TC3162U Support)
        Driver Feature  --->
        Common Feature  --->
    ---
        Load an Alternate Configuration File
        Save an Alternate Configuration File

              <Select>    < Exit >    < Help >
```

Step 2:Select the item which you want to configure, for example, choose the Bootbase Support SDRAMSIZE, and configure it as Support 16M SDRAM



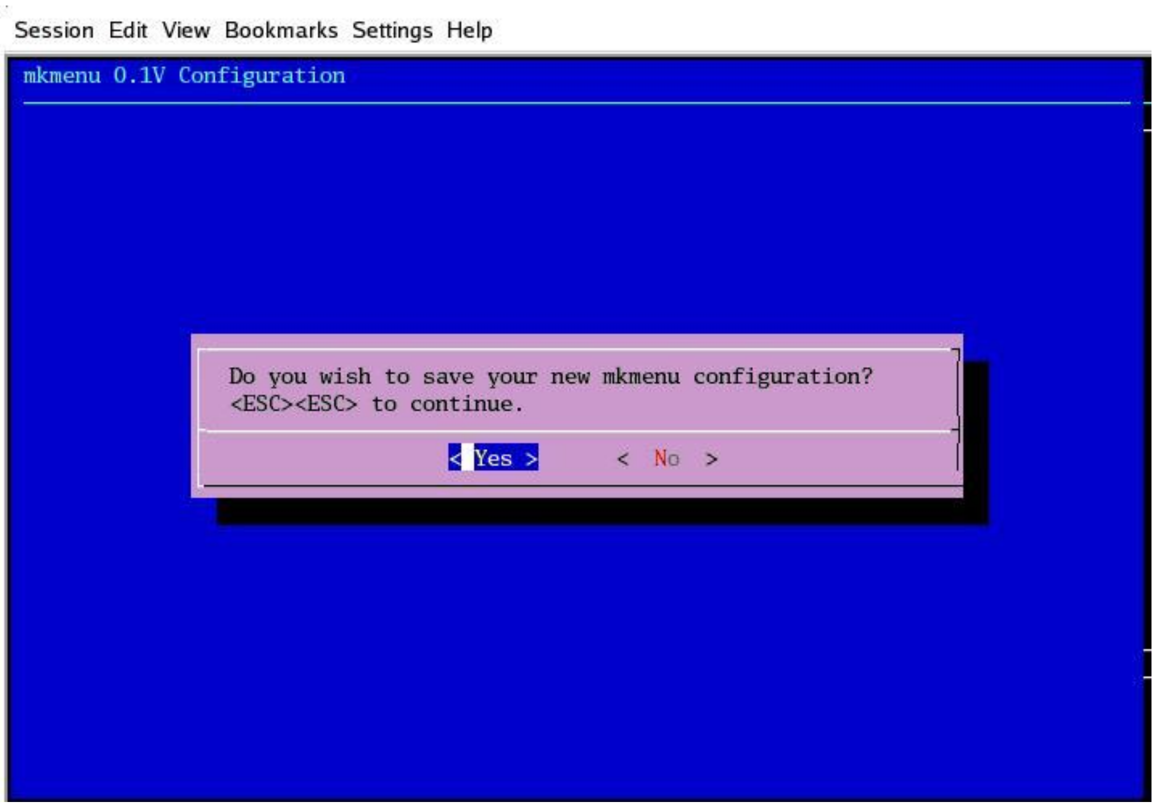
```
mkmenu 0.1V Configuration

Bootbase Suppport SDRAMSIZE
Use the arrow keys to navigate this window or press the hotkey of
the item you wish to select followed by the <SPACE BAR>. Press
<?> for additional information about this option.

          ( ) Support 8M SDRAM
          (X) Support 16M SDRAM
          ( ) Support 32M SDRAM
          ( ) Support 64M SDRAM

          <Select>      < Help >
```

Step 3:When you finish the configuration, please exit and save it, then your profile “test” will be configured as you want.



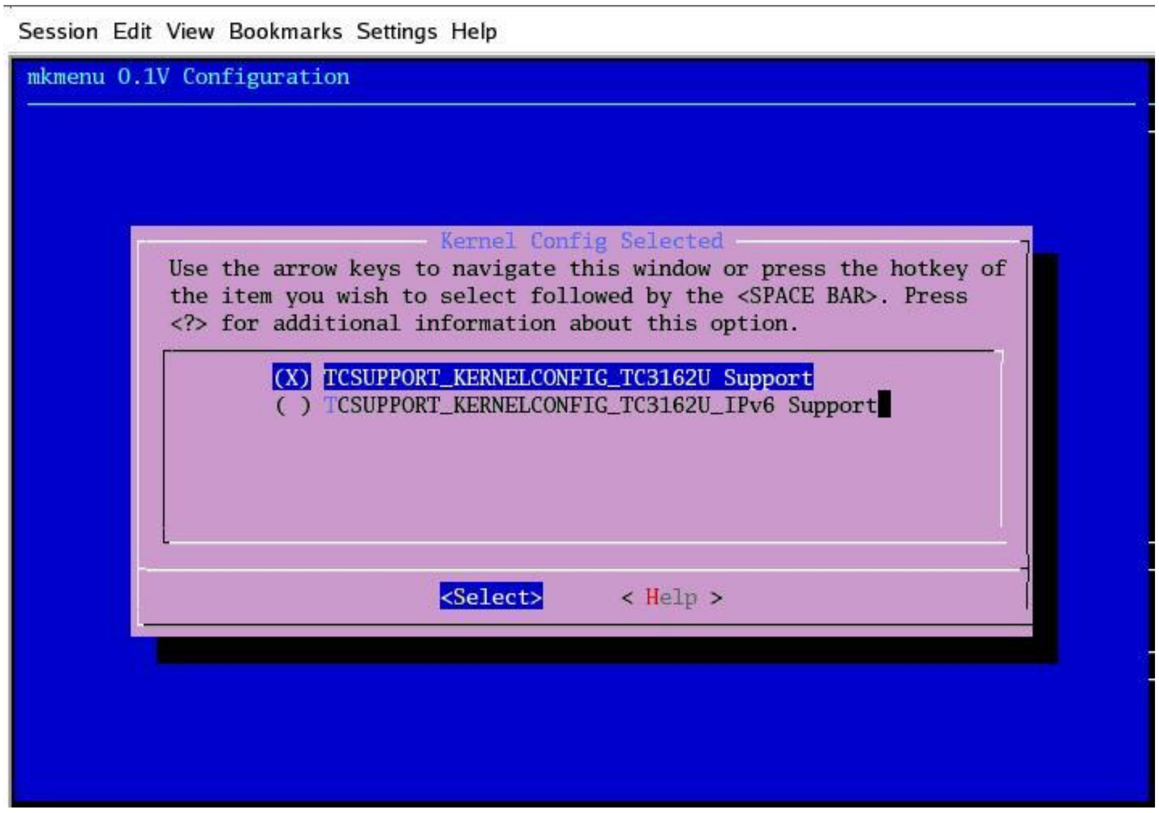
```
mkmenu 0.1V Configuration

Do you wish to save your new mkmenu configuration?
<ESC><ESC> to continue.
```

3.3 Configure Busybox or kernel

We provide two ways to configure Busybox or kernel.

(1) Enter ““releasebsp”” folder, Enter command “make PROFILE=test menuconfig”, and choose kernel config selected, then you can select which config file you want.



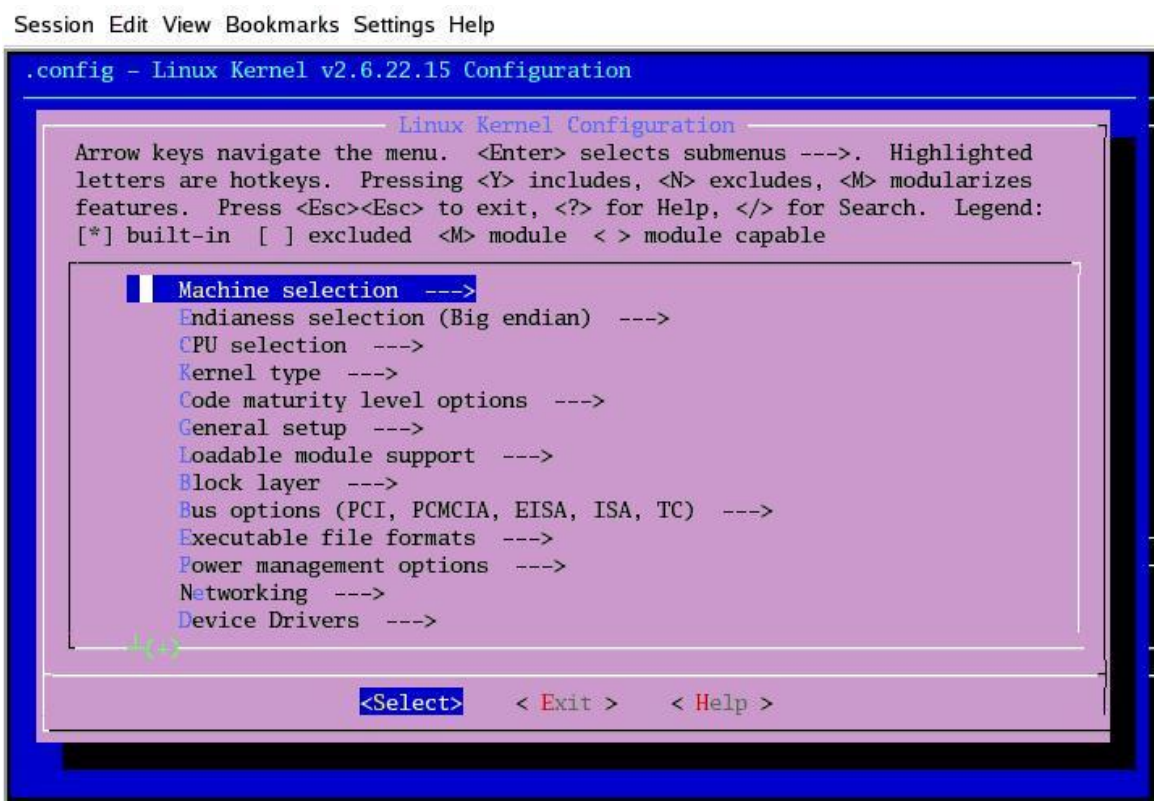
```
mkmenu 0.1V Configuration

Kernel Config Selected
Use the arrow keys to navigate this window or press the hotkey of
the item you wish to select followed by the <SPACE BAR>. Press
<?> for additional information about this option.

(X) TCSUPPORT_KERNELCONFIG_TC3162U Support
( ) TCSUPPORT_KERNELCONFIG_TC3162U_IPv6 Support

<Select>      < Help >
```

(2) You can also configure the kernel's detail. Enter command "make PROFILE=test kernel_menuconfig", after configure new kernel config file, will automatically copy to your profile folder as file "kernel.config". When build kernel will use this kernel config first.



```
Session  Edit  View  Bookmarks  Settings  Help
.config - Linux Kernel v2.6.22.15 Configuration
                     Linux Kernel Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted
  letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes
  features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend:
  [*] built-in  [ ] excluded  <M> module  < > module capable

          Machine selection  --->
          Endianess selection (Big endian)  --->
          CPU selection  --->
          Kernel type  --->
          Code maturity level options  --->
          General setup  --->
          Loadable module support  --->
          Block layer  --->
          Bus options (PCI, PCMCIA, EISA, ISA, TC)  --->
          Executable file formats  --->
          Power management options  --->
          Networking  --->
          Device Drivers  --->
      v(+)

                  <Select>    < Exit >    < Help >
```

Note:

(1) We have introduced the way of kernel configuration, and the Busybox configuration is in the same way while use busybox instead of kernel.For example, “make PROFILE=test busybox_menuconfig”

(2) Once you use the second way to configure the detail of kernel or busybox, it will always use this configuration, except you delete the Project/profile/test/kernel.config and Project/profile/test/busybox.config

4 Building images

4.1 Build image

1. Enter ““releasebsp”” folder, Enter command “make PROFILE=test clean” & “make PROFILE=test”

2. You can get “tcboot.bin”, “tclinux.bin” , “tclinux_allinone” and “tclinux_allinone.swap” files in “Project/images” folder

4.2 Bootloader

1. Enter ““releasebsp”” folder, Enter command “make PROFILE=test bootbase_clean” & “make PROFILE=test bootbase”, you can get tcboot.bin files in “Project/images” folder.
   Tips:You can select SDRAM size support of bootbase when you configure the profile.

2. Manufacture Related Data(MRD)
   The MRD data file named mi.conf includes the following information
   - MAC address
   - vendor name
   - product name
   - GPIO number for Ethernet switch reset in bootloader

   You can use any text editor to modify the content of mi.conf, and rebuild Bootloader.

4.3 Linux Kernel

1. Enter ““releasebsp”” folder, Enter command “make PROFILE=test menuconfig” to select right kernel config file

2. You can configure kernel manually, Enter command “make PROFILE=test kernel_menuconfig”, after configure new kernel config file, will automatically copy to your profile folder as file “kernel.config”. When build kernel will use this kernel config first.

3. Enter ““releasebsp”” folder, Enter command “make PROFILE=test kernel_clean” & “make PROFILE=test kernel” to build kernel

4. You should see a binary file named linux.7z is created after building complete at tc3162/linux folder.

5. The linux.bin is the uncompressed linux kernel image at tc3162/linux folder.

6. USB 2.0 Device(Only TC3162U, TC3182,RT63260 support)
   - linux/drivers/usb/gadget/g_ether.ko

7. USB 2.0 Host(Only TC3162U, TC3182,RT65168 and RT63365 support)
   - linux/drivers/scsi/scsi_mod.ko
   - linux/drivers/scsi/scsi_wait_scan.ko
   - linux/drivers/scsi/sd_mod.ko
   - linux/drivers/usb/core/usbcore.ko
   - linux/drivers/usb/host/ehci-hcd.ko
   - linux/drivers/usb/host/ohci-hcd.ko
   - linux/drivers/usb/storage/usb-storage.ko
   - linux/fs/fat/fat.ko
   - linux/fs/msdos/msdos.ko
   - linux/fs/vfat/vfat.ko
   - linux/fs/nls/nls_ascii.ko
   - linux/fs/nls/nls_base.ko
   - linux/fs/nls/nls_cp437.ko

4.4 Busybox

1. Change directory to ""releasebsp"" folder, Enter command "make PROFILE=test menuconfig" to select right busybox config file
2. You can configure busybox manually, Enter command "make PROFILE=test busybox_menuconfig", after configure new kernel config file, will automatically copy to your profile folder as file "busybox.config". When build busybox will use this busybox config first.
3. Change directory to ""releasebsp"" folder, Enter command "make PROFILE=test busybox_clean" & "make PROFILE=test busybox" to build busybox
4. Busybox apps will automatically installed in filesystem folder

.

4.5 Modules

Change directory to ""releasebsp"" folder, Enter command "make PROFILE=test modules_clean" & "make PROFILE=test modules"

4.6 Applications

Change directory to ""releasebsp"" folder, Enter command "make PROFILE=test apps_clean" & "make PROFILE=test apps"

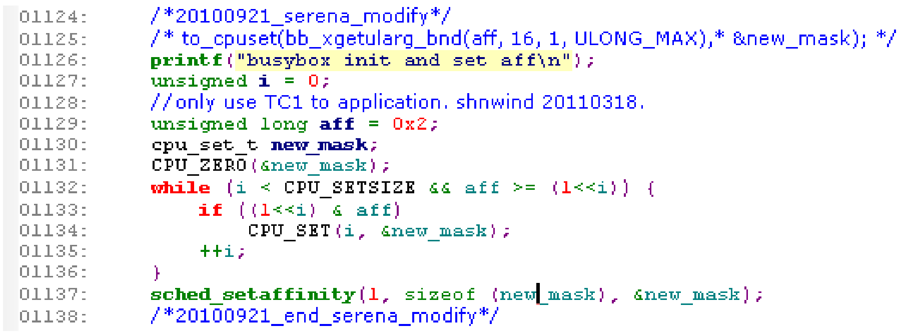
4.7 Buildimage only

Change directory to ""releasebsp"" folder, Enter command "make PROFILE=test buildimage"

5 Task and interrupt setting for user space, kernel and VoIP for CPU binding

1. User space task should be bound to CPU1.

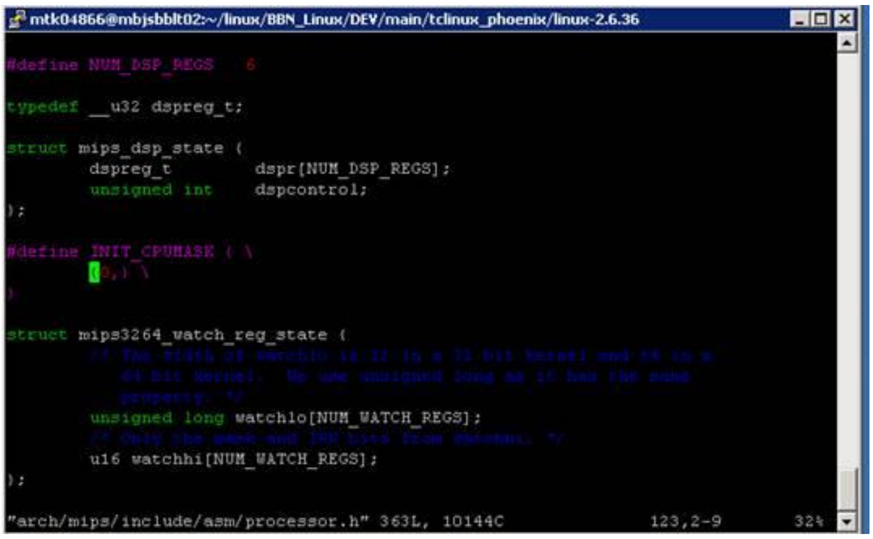
   file：tclinux_phoenix\apps\public\busybox-1.00\init.c



```
/*20100921_serena_modify*/
/* to_cpuset(bb_xgetularg_bnd(aff, 16, 1, ULONG_MAX),* &new_mask); */
printf("busybox init and set aff\n");
unsigned i = 0;
//only use TC1 to application. shnwind 20110318.
unsigned long aff = 0x2;
cpu_set_t new_mask;
CPU_ZERO(&new_mask);
while (i < CPU_SETSIZE && aff >= (1<<i)) {
    if ((1<<i) & aff)
        CPU_SET(i, &new_mask);
    ++i;
}
sched_setaffinity(1, sizeof (new_mask), &new_mask);
/*20100921_end_serena_modify*/
```

2. Kernel task should be bound to CPU0,CPU1,CPU2.

   file: linux-2.6.36/arch/mips/include/asm/processor.h
   INIT_CPUMASK set 0 to 7 (bound to CPU0/CPU1/CPU2)



```
mtk04866@mbjsbblt02:~/linux/BBN_Linux/DEV/main/tclinux_phoenix/linux-2.6.36

#define NUM_DSP_REGS   6

typedef __u32 dspreg_t;

struct mips_dsp_state {
        dspreg_t        dspr[NUM_DSP_REGS];
        unsigned int    dspcontrol;
};

#define INIT_CPUMASK { \
        {0,} \
}

struct mips3264_watch_reg_state {
        /* The width of watchlo is 32 in a 32 bit kernel and 64 in a
           64 bit kernel.  We use unsigned long as it has the same
           property. */
        unsigned long watchlo[NUM_WATCH_REGS];
        /* Only the mask and IRW bits from watchhi. */
        u16 watchhi[NUM_WATCH_REGS];
};
"arch/mips/include/asm/processor.h" 363L, 10144C              123,2-9        32%
```

3. CPU3 is only assigned to use for VoIP, other interrupt & user-space task can't be bound to CPU3 except PCM and timer interrupt. For example,qdma interrupt should not be bound to CPU3,and it is recommended to be bound to CPU0,CPU1,CPU2.

6 How to add a new application

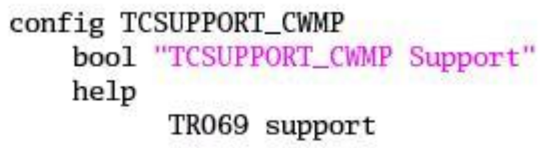
Here’s an example of how to add TR069 feature:

Step 1: Add source code into apps/TR69_64

Step 2: Define a new compile option: TCSUPPORT_CWMP

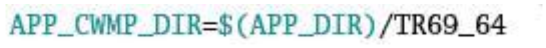
Step 3: Add “TCSUPPORT_CWMP” into profile, for example,you can add “TCSUPPORT_CWMP=y” to test/test.profile

Step 4: Modify the Project/config/menuconfig/Config.in to support menu configuration



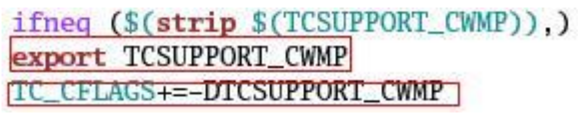
```
config TCSUPPORT_CWMP
    bool "TCSUPPORT_CWMP Support"
    help
          TR069 support
```

Step 5: Add your application path in Project/dir.mak



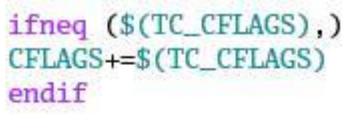
```
APP_CWMP_DIR=$(APP_DIR)/TR69_64
```

Step 6: Export your compile option value and modify the TC_CFLAGS in Project/rule.mak



```
ifneq ($(strip $(TCSUPPORT_CWMP)),)
export TCSUPPORT_CWMP
TC_CFLAGS+=-DTCSUPPORT_CWMP
```

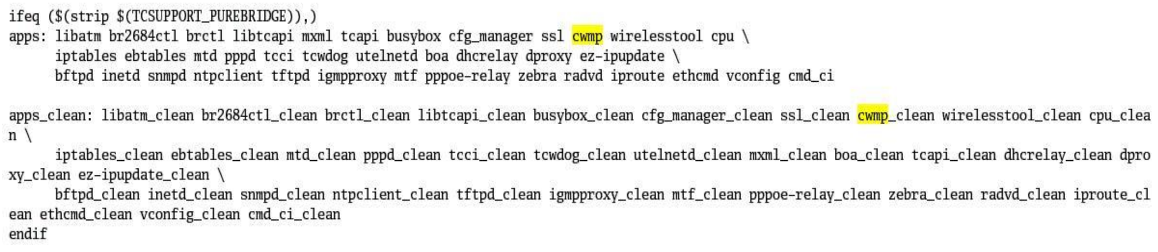
Step 7: Add the TC_CFLAGS into apps/TR69_64/Makefile



```
ifneq ($(TC_CFLAGS),)
CFLAGS+=$(TC_CFLAGS)
endif
```

Step 8: Modify the Project/MakeFile_Main

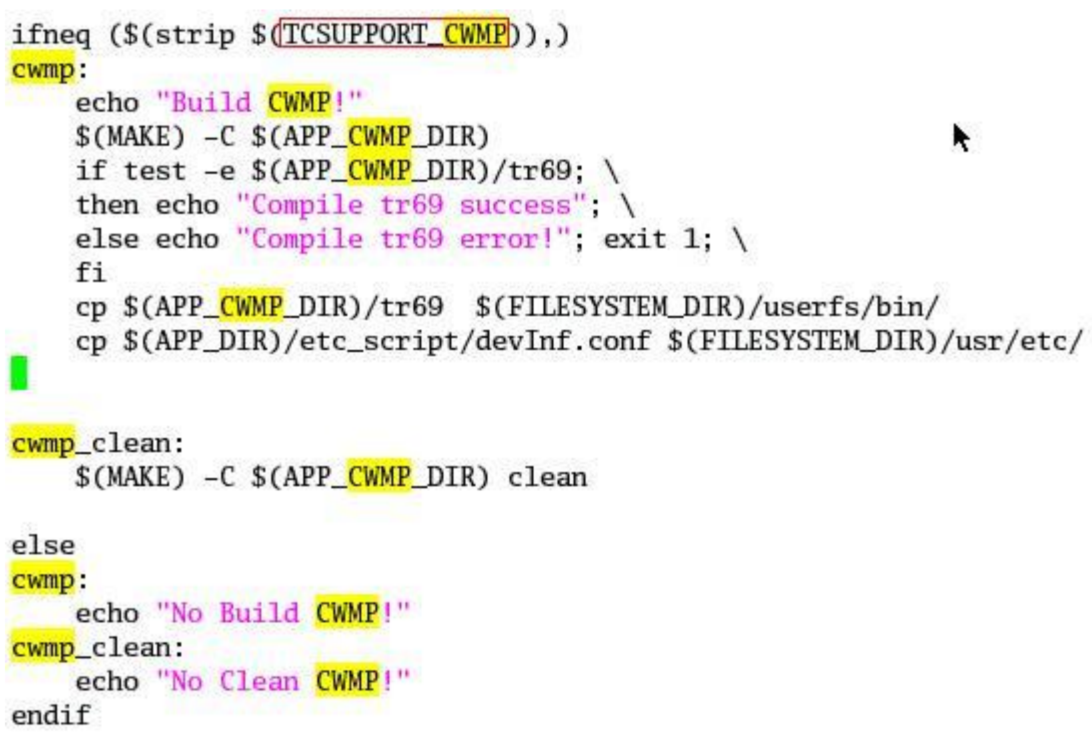
(1)add cwmp into apps:



```
ifeq ($(strip $(TCSUPPORT_PUREBRIDGE)),)
apps: libatm br2684ctl brctl libtcapi mxml tcapi busybox cfg_manager ssl cwmp wirelesstool cpu \
      iptables ebtables mtd pppd tcci tcwdog utelnetd boa dhcrelay dproxy ez-ipupdate \
      bftpd inetd snmpd ntpclient tftpd igmpproxy mtf pppoe-relay zebra radvd iproute ethcmd vconfig cmd_ci

apps_clean: libatm_clean br2684ctl_clean brctl_clean libtcapi_clean busybox_clean cfg_manager_clean ssl_clean cwmp_clean wirelesstool_clean cpu_clea
n \
      iptables_clean ebtables_clean mtd_clean pppd_clean tcci_clean tcwdog_clean utelnetd_clean mxml_clean boa_clean tcapi_clean dhcrelay_clean dpro
xy_clean ez-ipupdate_clean \
      bftpd_clean inetd_clean snmpd_clean ntpclient_clean tftpd_clean igmpproxy_clean mtf_clean pppoe-relay_clean zebra_clean radvd_clean iproute_cl
ean ethcmd_clean vconfig_clean cmd_ci_clean
endif
```

(2)add compile cwmp



```
ifneq ($(strip $(TCSUPPORT_CWMP)),)
cwmp:
	echo "Build CWMP!"
	$(MAKE) -C $(APP_CWMP_DIR)
	if test -e $(APP_CWMP_DIR)/tr69; \
	then echo "Compile tr69 success"; \
	else echo "Compile tr69 error!"; exit 1; \
	fi
	cp $(APP_CWMP_DIR)/tr69  $(FILESYSTEM_DIR)/userfs/bin/
	cp $(APP_DIR)/etc_script/devInf.conf $(FILESYSTEM_DIR)/usr/etc/

cwmp_clean:
	$(MAKE) -C $(APP_CWMP_DIR) clean

else
cwmp:
	echo "No Build CWMP!"
cwmp_clean:
	echo "No Clean CWMP!"
endif
```