Software Development Kit

Switch Product

Developer Guide

(CONFIDENTIAL: Development Partners Only)

Rev. 2.1.4.53590
25 December 2014

Realtek Semiconductor Corp.

No. 2, Innovation Road II, Hsinchu Science Park, Hsinchu 300, Taiwan

Tel.: +886-3-578-0211 Fax: +886-3-577-6047

www.realtek.com

USING THIS DOCUMENT

This document is intended for use by the system engineer when integrating with Realtek Switch Software SDK. Though every effort has been made to assure that this document is current and accurate, more information may have become available subsequent to the production of this guide. In that event, please contact your Realtek representative for additional information that may help in the development process.

REVISION HISTORY

Revision	Release Date	Summary
2.0.0	2010/6/25	1. Formal version supporting RTL8389M & RTL8328M series chips, the version only supporting Linux only. 2. Move original chapter 4 "Installing Tool Chains" to Installation Guide document.
2.0.1	2010/8/13	1. Change revision
2.0.2	2010/10/19	1. Revision for SDK update
2.0.3	2010/10/27	1. Supported RTL8214B/ RTL8214FB PHY chip 2. Add new symbols in menuconfig for above new PHYs: **`CONFIG_SDK_RTL8214B and CONFIG_SDK_RTL8214B`**
2.0.4	2011/2/25	1. Supported 8212B chip 2. Supported RTL8231 driver
2.0.5	2011/4/18	1. Supported 8328L chip and demo board
2.0.6	2011/7/4	1. Identify 8212B and 8214B chip in U-Boot and SDK

Revision	Release Date	Summary
2.0.6.20571	2011/8/5	1. Add the new symbol in SDK menuconfig for quad SPI IO: **`CONFIG_SDK_BSP_MTD_SPI_QUAD_IO`**
2.0.7.24969	2011/11/04	1. Supported 8328M, 8328S and 8328L R0C chip. 2. Supported **`CONFIG_SDK_WA_EEE_COMPATIBLE`** option 3. Supported PIE/ACL Input Data Endian Select in following two options: **`CONFIG_SDK_PIE_DATA_ENDIAN_BIG`** and **`CONFIG_SDK_PIE_DATA_ENDIAN_LITTLE`** 4. Supported customer RTK mechanism by **`CONFIG_SDK_DRIVER_RTK_CUSTOMER`** option. 5. Supported NIC Rx thread by **`CONFIG_SDK_RX_THREAD`** option 6. Supported LinkMon thread mode selection by **`CONFIG_SDK_LINKMON_POLLING_MODE`** `and` **`CONFIG_SDK_LINKMON_ISR_MODE`** option. 7. Supported **CONFIG_SDK_WA_RTL8231_RESET** option
2.0.7.26226	2012/01/09	1. Supported **CONFIG_SDK_WA_88E6063_COMPATIBLE** option
2.0.7.29481	2012/6/1	1. Supported **CONFIG_SDK_WA_PKTBUF_WATCHDOG** option 2. Supported CONFIG_SDK_LINKMON_MIXED_MODE option 3. Add 8328L serial mode board model in u-boot.
Pre2.1.0.27873	2012/5/19	1. The SDK v2.1.0 pre-package document (svn revision: 27873) 2. The SDK v2.1.0 kernel package based is changed from linux to uClinux.
Pre2.1.0.30160	2012/6/29	1. The 2.1.0 and late SDK package is not supported the RTL8329M, RTL8389M and RTL8389LM chip. 2. Supported the cypress and maple series MAC chips and RTL8218B PHY chip.
Pre2.1.0.31102	2012/7/19	1. The SDK supported the RTL8352M chip in this revision. 2. The SDK supported the uClinux 2.6.32.58 kernel in OSAL APIs. 3. The uClinux 2.6.32.58 porting guide, please reference document: RTK_MS_SDK_Linux_Kernel_2.6.32.58_Porting_Guide.doc 4. The detail menuconfig option select, please reference the document: RTK_MS_SDK_DemoBoard_UserGuide_2.1.0.31102.doc
Pre2.1.0.32476	2012/9/7	1. Update the menuconfig for package name.
2.1.0.33152	2012/10/3	1. Update the menuconfig pictures and document description. 2. Add new symbols in menuconfig for following new PHYs: **`CONFIG_SDK_RTL8218FB and CONFIG_SDK_RTL8214FC`** 3. Supported **`CONFIG_SDK_SOFTWARE_CONTROL_LED`** option

Revision	Release Date	Summary
2.1.0.33894	2012/10/30	1. package name update
2.1.0.36032	2013/01/11	1. package name update 2. Supported **`CONFIG_SDK_UART1`** option. 3. Supported OSAL atomic, wait and workqueue function
2.1.0.36693	2013/01/30	1. Supported RTL8218B/RTL8218FB/RTL8214FC MP chips 2. Supported **CONFIG_SDK_WA_RTL833X_COMBO_LED** option 3. package name update
2.1.1.38074	2013/03/29	1. package name update
2.1.1.39239	2013/05/16	1. package name update
2.1.2.41872	2013/08/09	1. package name update
2.1.3.pre45842	2014/01/09	1. package name update
2.1.3.46351	2014/02/06	1. package name updated.
2.1.3.46351-next	2014/04/16	1. Remove ECOS item.
2.1.4.53590	2014/12/25	1. package name updated.

Table of Contents

Chapter 1 Introduction

The guideline document is providing for Realtek customer to understand Realtek SDK architecture and guide them to port the SDK to their environment.

1.1 Supported Platforms

- 8328M/8328S/8328L demo board
- 8330M/8332M/8380M/8382M demo board
- 8352M/8353M/8392M/8393M/8396M demo board

1.2 Supported Chip, PHY, OS

- Switch Chip
  - RTL8328M: 24*10/100M + 4*1000M-PORT MANAGED SWITCH
  - RTL8330M: 16*10/100M + 4*1000M-PORT MANAGED SWITCH
  - RTL8332M: 24*10/100M + 4*1000M-PORT MANAGED SWITCH
  - RTL8380M: 20*1000M-PORT MANAGED SWITCH
  - RTL8382M: 28*1000M-PORT MANAGED SWITCH
  - RTL8352M: 24*10/100M + 4*1000M-PORT MANAGED SWITCH
  - RTL8353M: 48*10/100M + 4*1000M-PORT MANAGED SWITCH
  - RTL8392M: 28*1000M-PORT MANAGED SWITCH
  - RTL8393M: 52*1000M-PORT MANAGED SWITCH
  - RTL8396M: 24*1000M + 2*10G-PORT MANAGED SWITCH

- PHY Chip
  - RTL8201 : 1*10/100M-PORT PHY
  - RTL8208 : 8*10/100M-PORT PHY
  - RTL8214 : 4*10/100/1000M-PORT PHY
  - RTL8214F: 4*10/100/1000M-PORT COMBO PHY
  - RTL8218 : 8*10/100/1000M-PORT PHY
  - RTL8212F: 2*10/100/1000M-PORT COMBO PHY
  - RTL8212B/RTL8214B: 2/4*10/100/1000M-PORT COMBO PHY
  - RTL8214FB: 4*10/100/1000M-PORT COMBO PHY
  - RTL8218B : 8*10/100/1000M-PORT PHY
  - RTL8218FB: 4*10/100/1000M-PORT +

4*10/100/1000M-PORT COMBO PHY
- RTL8214FC: 4*10/100/1000M-PORT COMBO PHY

- OS
  - uClinux Kernel 2.6.19
  - uClinux Kernel 2.6.32.58

1.3 New in This Release

- This and late revision is not supported the RTL8329M, RTL8389M and RTL8389LM MAC chip.
- Supported RTL8330M, RTL8332M, RTL8380M and RTL8382M MAC chip.
- Supported RTL8352M, RTL8353M, RTL8392M, RTL8393M and RTL8396M MAC chip.
- Supported RTL8218B, RTL8218FB and RTL8214FC PHY chip.
- The OS supported change from linux 2.6.19 to uClinux 2.6.19.
- Supported the menuconfig option for select the old or new diag syntax.
  - old diag syntax in diag directory for RTL8328 series
  - new diag syntax in diag_v2 for RTL8328, RTL833x, RTL835x, RTL838x and RTL839x series.
- Supported the new uClinux 2.6.32.58, the uClibc 0.9.30.3 and the tool chain G.C.C 4.3.6
- Supported the menuconfig option for select the uClinux 2.6.19 or 2.6.32.58
- Supported the menuconfig option for select the uClibc 0.9.28 or 0.9.30.3
- The new kernel, uClibc and toolchain installs please guide the document: RTK_MS_SDK_Linux_Kernel_2.6.32.58_PortingGuide_v1.0.6.pdf
- Supported the RTL8218B/RTL8218FB/RTL8214FC MP chips
- Supported **CONFIG_SDK_WA_RTL833X_COMBO_LED** option

Chapter 2 Terms Definition

Term	Description
SDK	Software Development Kit
RTK	Realtek^{TM} Public Service API Layer
RTL	Realtek^{TM} LAN Product Line
BSP	Board Support Package
DAL	Device Abstraction Layer
HAL	Hardware Abstraction Layer
IOAL	Input/Output Abstraction Layer
OSAL	Operation System Abstraction Layer
RTUSR	Realtek^{TM} Linux User Mode Layer
RTDRV	Realtek^{TM} Linux Kernel Mode Layer
DiagShell	Realtek^{TM} Diagnostic Shell User Interface
RTNIC	Realtek^{TM} Linux NIC Module

Chapter 3 SDK Components

3.1. Tree View

```
[sdk]
   [build]     Configure and makefiles
   [config]    SDK menuconfig files
   [example]   SDK example code files
   [include]   SDK header file
   [src]       SDK source code
   [system]    OS dependent layers
```

3.2. SDK Directories

```
[include]             SDK public API include files
[src]
   [app]
      [diag]          SDK diagnostic shell V1
      [diag_v2]       SDK diagnostic shell V2
   [common]
      [util]          SDK utility
   [dal]
      [cypress]       Cypress switch product line
      [esw]           Ethernet switch product line
      [maple]         Maple switch product line
      [ssw]           Smart switch product line
   [hal]
      [chipdef]       Chip information APIs.
      [common]        Control and common APIs, like init
                      sequence and probe mechanism
      [mac]           Switch MAC driver
      [phy]           Switch PHY driver
   [rtk]              Realtek SDK public API layer
      [customer]      Customer RTK API layer
```

```
[system]
   [common]            Lower layer common code
      [debug]          Debug module
      [rtcore]         Rtcore common code
   [drv]               Lower layer driver
      [gpio]           GPIO driver
      [intr]           Interrupt control driver
      [nic]            NIC driver
      [rtl8231]        RTL8231 driver
      [smi]            SMI driver
      [swcore]         Swcore driver
      [swled]          Software LED driver
      [uart]           uart driver
      [watchdog]       Watchdog driver
   [include]           System dependent include files
   [ioal]              I/O layer to different H/W interface
   [linux]
      [linux-2.6.32.x] Linux 2.6.32.58 package
      [linux-2.6.x]  Linux 2.6.19 package
      [rtcore]         Linux Rtcore module
      [rtdrv]          SDK Linux kernel mode
      [rtk]
         [rtusr]       SDK Linux user mode
      [rtnic]          SDK Linux NIC module
   [osal]
       [linux]         Linux OS arbitration layer
```

3.3. Component Description

OSAL: A layer between Realtek SDK and operation system let the Realtek SDK become OS independent. OSAL will take care the different OS implementation. You can easy to change to different OS, only need to implement the OSAL body for your specified OS.

COMMON: common utility and debug module for internal SDK used. The header file in common directory is public for upper application and customer who use the SDK.

IOAL: HW interface I/O layer, we support the Lexra bus directly access for realtek switch chip right now.

HAL: A layer to provide all basic switch and PHY access, include the switch table, register, table field, register field and PHY register read/write and additional operation. The layer can dynamic probe supported switch and PHY (see 1.2) and associate the correct driver. Support the foreword compatible in switch and PHY revision. Customer don't need to update the SDK version when use the future revision switch and PHY chip.

DAL: A layer to service RTK layer requirement in all supported chips. It have do the re-mapping to different product line depend on chip architecture. And design some common structure, shadow and database for hidden the chip behavior. Protection mechanism, like semaphore also implement in this layer.

RTK: A public service APIs layer for customer in specified device view. Only permit customer to use the layer APIs. It is one physical chip view layer for upper application usage. There is one sub-directory **CUSTOMER** that permit customer to do specified customization implementation by themselves.

RTDRV: a layer in Linux kernel mode to translate the user mode request to RTK layer. The layer is not need in non Linux platform.

RTUSR: a layer in application layer to service upper layer (example: diagnostic shell). In Linux platform, the layer receives the upper layer request and use one kind of Linux user/kernel mode to translate the request into the kernel mode RTDRV module.

Diagnostic shell: It is a realtek style command line diagnostic shell. It is design some basic CLI command for verification and debug the chip detail behavior quickly. There are two diag command styles and can select from menuconfig option.

RTNIC: one Linux loadable NIC module to handle the NIC Tx/Rx mechanism, it have handle the NIC Tx/Rx callback function, include register one net_device to Linux. It is one layer between Linux OS and chip NIC driver.

DRV: some lower layer drivers and separate by function based, like GPIO, INTR, NIC, RTL8231, SMI, SWCORE, SWLED and

WATCHDOG components.

3.4. Component from Linux Loadable Module View

You can select to build module or non-module style from following configure (See **Fig.1**). If you select to build out the module style, there are 4 Linux loadable modules are generated and each module involves component list as following:

- Application - SDK demo shell
  - diagnostic shell
  - rtusr library

- RTDRV module - SDK user/kernel interface module
  - rtdrv

- SDK module - SDK major driver module
  - rtk
  - dal
  - hal

- RTCORE module - System dependent driver module
  - drv (gpio, intr, nic, rtl8231, smi, swcore, swled, uart and watchdog)
  - ioal
  - osal
  - debug module

NIC module – SDK NIC module
  - rtnic

3.5. Public Directory and File Introduction

Following directories and files are public for SDK user:

- system/include/common/*.h

error.h	System error symbol definition
rt_autoconf.h	SDK compiler flags (auto generation)

type.h	System common type definition
util.h	System utility definition

➢ include/common/*.h

rt_error.h	Error symbols definition
rt_type.h	SDK common type definition
rt_version.h	SDK version definition

➢ system/include/drv/nic/*.h | system/drv/nic/*.c

common[.h]	NIC symbol and structure definition
nic[.h|.c]	NIC driver (CPU tag, TX/RX) APIs

➢ include/rtk/*.h | src/rtk/*.c

acl[.h|.c]	ACL APIs (RTL833xM, RTL835xM, RTL8380M, RTL8382M and RTL839xM)
customer_hook[.h]	Customer RTK hook APIs
default[.h]	SDK default configure
diag[.h|.c]	Diagnostic APIs
dot1x[.h|.c]	802.1x port/mac based APIs
eee[.h|.c]	EEE APIs
filter[.h|.c]	Flow table/ACL APIs (RTL8389M)
flowctrl[.h|.c]	Ingress/Egress threshold APIs
init[.h|.c]	System initialize APIs
l2[.h|.c]	L2 address APIs
l3[.h|.c]	L3 routing APIs
led[.h|.c]	LED APIs
mirror[.h|.c]	Mirror APIs
mpls[.h|.c]	MPLS APIs
oam[.h|.c]	OAM(802.3ah) & CFM(802.1ag) APIs
pie[.h|.c]	PIE APIs (RTL8328M)
port[.h|.c]	Port APIs
qos[.h|.c]	Priority decision, ingress remap, Egress remark and queue scheduling
rate[.h|.c]	Storm and ingress/egress rate APIs
sec[.h|.c]	Security APIs
stat[.h|.c]	Statistic counter APIs
stp[.h|.c]	Spanning tree (802.1D/1w/1s) APIs
svlan[.h|.c]	VLAN stacking (802.1ad) APIs
switch[.h|.c]	Other global wise APIs

time[.h|.c]	IEEE 1588 (Time and PTP) APIs
trap[.h|.c]	Packet2CPU and RMA APIs
trunk[.h|.c]	TRUNK APIs
vlan[.h|.c]	VLAN (802.1Q) APIs

- include/rtk/customer/*.h | src/rtk/customer/*.c

customer_api[.h|.c]	Customer RTK APIs
customer_demo[.h|.c]	Customer RTK demo APIs

- include/osal/*.h

cache.h	Cache relative APIs
inet.h	INET relative APIs
isr.h	Interrupt relative APIs
lib.h	Library APIs
memory.h	Memory APIs
print.h	Print APIs
sem.h	Semaphore APIs
spl.h	Spin lock APIs
thread.h	Thread APIs
time.h	Timer APIs
atomic.h	Atomic APIs
wait.h	Wait APIs
workqueue.h	Workqueue APIs

Chapter 4 Installing SDK

4.1. Packages Definition

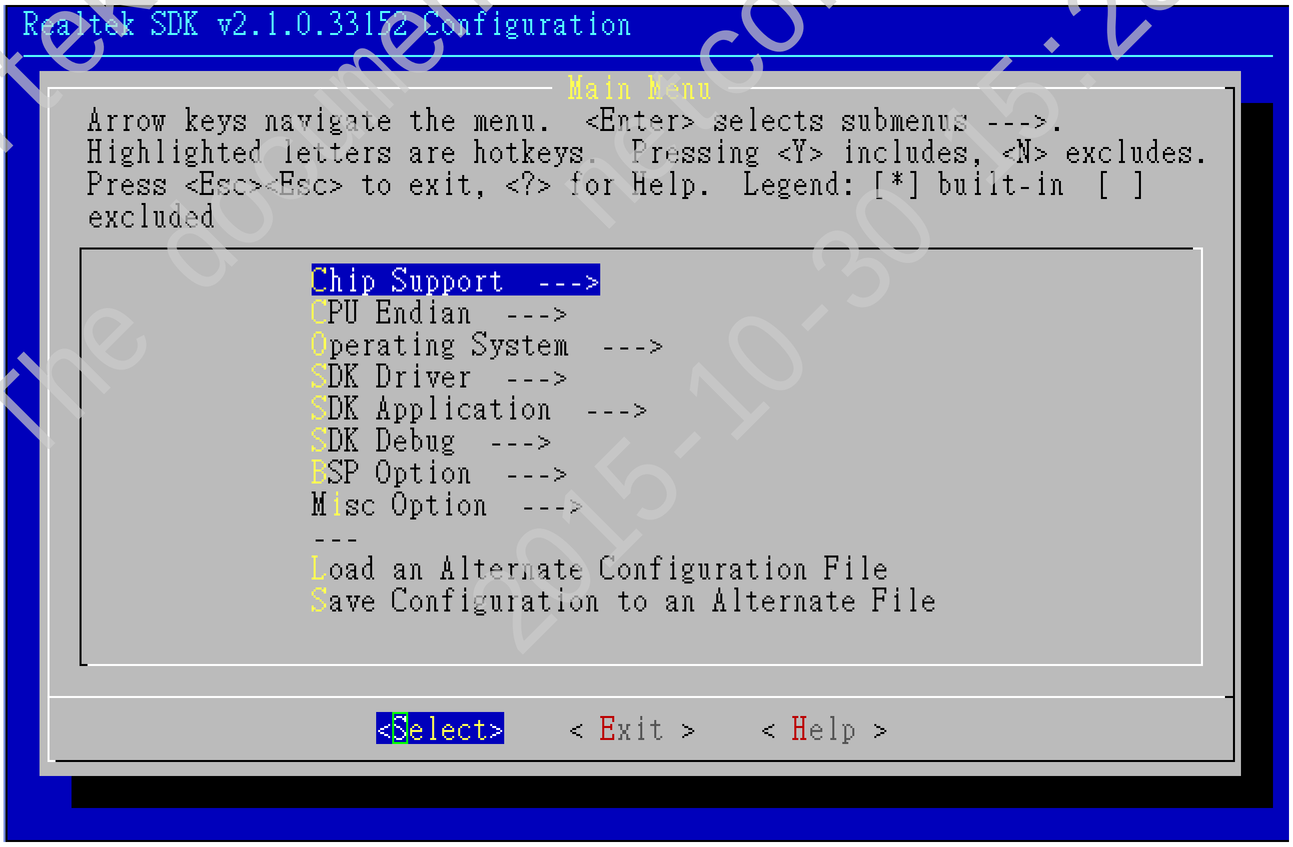
- **rtk-ms-uClinux-src-2.6.19-2.6.32.58-svn53590.tar.gz**
  - uClinux kernel 2.6.19 and 2.6.32.58 distribution
- **rtk-ms-uboot-src-1.3.0.53590.tar.gz**
  - old loader package: u-boot v1.3.0
- **rtk-ms-sdk-src-2.1.4.53590.tar.gz**
  - Realtek draft SDK package
- **rtk-ms-uboot-2011.12-src-svn53590.tar.gz**
  - new u-boot-2011.12 package: u-boot-2011.12

4.2. Create Development Environment

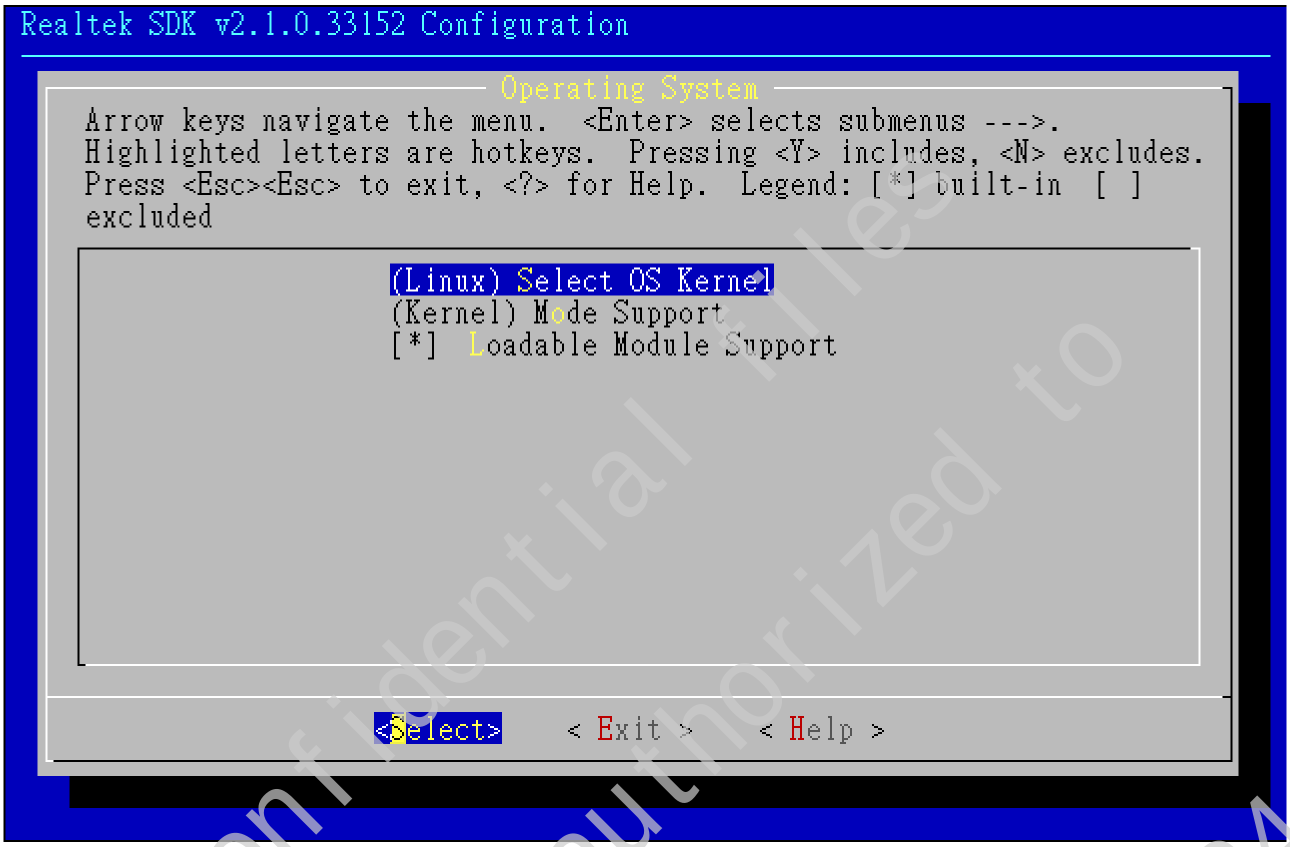
- Create your working directory, example "working".
- Select to copy above packages to "working" directory.
  - If have use the new u-boot-2011.12, need to copy the **rtk-ms-uboot-2011.12-src-svn53590.tar.gz** package.
  - If have use the old loader, need to copy the **rtk-ms-uboot-src-1.3.0.53590.tar.gz** package.
- Extract **rtk-ms-ulinux-src-2.6.19-2.6.32.58-svn53590.tar.gz** and you will see 'kernel' directory in your working directory.
- Extract **rtk-ms-uboot-src-1.3.0.53590.tar.gz** and you will see 'u-boot' directory in your working directory.
- Extract **rtk-ms-sdk-src-2.1.4.53590.tar.gz** and you will see 'sdk' directory in your working directory.
- Extract **rtk-ms-uboot-2011.12-src-svn53590.tar.gz** and you will see 'u-boot-2011.12' directory in your working directory.

4.3. Menuconfig

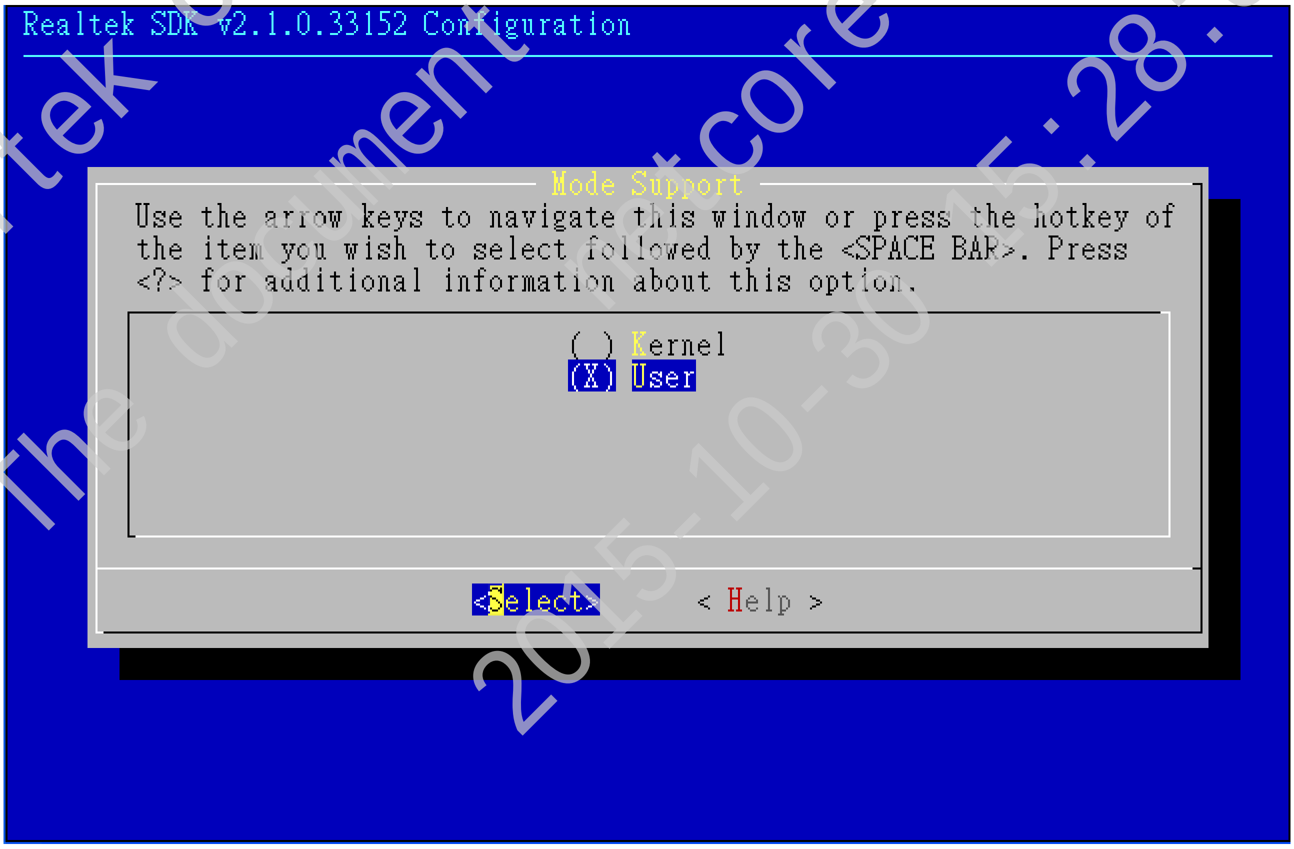
- In SDK 2.1.4.53590 package, the default menuconfig is for RTL8393M/RTL8353M QA board with SDK driver and NIC driver in kernel mode. You need to change the configure for your specified board if the default menuconfig is not meet your expected.
- You can type 'make menuconfig' to configure your kernel and SDK settings.
- After you exit the configuration tool with save option, all necessary dynamic paths and files will be automatic created in your working directory.
- You can refer to building files in sdk/build, and modify them to your style.
- For Linux solution, SDK support in both kernel mode and user mode. We recommend run SDK driver in kernel mode (Default option). But if you have specified condition and try to run SDK driver in Linux user mode, please change "Mode Support" option to "User".



```
Realtek SDK v2.1.0.33152 Configuration

                         Main Menu
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes.
  Press <Esc><Esc> to exit, <?> for Help.  Legend: [*] built-in  [ ]
  excluded

              Chip Support  --->
              CPU Endian  --->
              Operating System  --->
              SDK Driver  --->
              SDK Application  --->
              SDK Debug  --->
              BSP Option  --->
              Misc Option  --->
              ---
              Load an Alternate Configuration File
              Save Configuration to an Alternate File

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



```
Realtek SDK v2.1.0.33152 Configuration
                         Operating System
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes.
  Press <Esc><Esc> to exit, <?> for Help.  Legend: [*] built-in  [ ]
  excluded

                 (Linux) Select OS Kernel
                 (Kernel) Mode Support
                 [*]  Loadable Module Support

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



```
Realtek SDK v2.1.0.33152 Configuration
                          Mode Support
  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.

                        ( ) Kernel
                        (X) User

                  <Select>    < Help >
```

4.4. SDK Compiler Options

There are two files for SDK compiler options:

- **sdk/include/common/rt_autoconf.h**: compiler options for SDK source files.
- **sdk/config/.config**: compiler options for building procedures, like Makefile.

Above two files is automatic generating by 'make menuconfig'. For definition and description of each symbol, you can select 'Help' in your configuration screen or please refer to the help file, which is located at **sdk/config/Configure.help**

Some major options list as following:

- Chip select options:
  - **CONFIG_SDK_RTL8328**: include RTL8328M, RTL8328S and RTL8328L MAC chips
  - **CONFIG_SDK_RTL8390**: include RTL8352M, RTL8353M, RTL8392M, RTL8393M and RTL8396M MAC chips
  - **CONFIG_SDK_RTL8380**: include RTL8330M, RTL8332M, RTL8380M and RTL8382M MAC chips
  - **CONFIG_SDK_RTL8208**: include RTL8208 PHY chip
  - **CONFIG_SDK_RTL8214**: include RTL8214 PHY chip
  - **CONFIG_SDK_RTL8214F**: include RTL8214F PHY chip
  - **CONFIG_SDK_RTL8201**: include RTL8201 PHY chip
  - **CONFIG_SDK_RTL8218**: include RTL8218 PHY chip
  - **CONFIG_SDK_RTL8218B**: include RTL8218B PHY chip
  - **CONFIG_SDK_RTL8218FB**: include RTL8218FB PHY chip
  - **CONFIG_SDK_RTL8212F**: include RTL8212F PHY chip
  - **CONFIG_SDK_RTL8212B**: include RTL8212B PHY chip
  - **CONFIG_SDK_RTL8214B**: include RTL8214B PHY chip
  - **CONFIG_SDK_RTL8214FB**: include RTL8214FB PHY chip
  - **CONFIG_SDK_RTL8214FC**: include RTL8214FC PHY chip
  - **CONFIG_SDK_RTL8231**: include RTL8231 LED/GPIO chip
- OS select options:
  - **CONFIG_SDK_KERNEL_LINUX**: SDK for Linux platform
  - **CONFIG_SDK_KERNEL_LINUX_MODULE_LOADABLE**: Compile SDK component to become a loadable Linux module

- Component select options:
  - **CONFIG_SDK_DRIVER_RTCORE:** Select the option to compile the feature code for SDK Kernel Core Driver
  - **CONFIG_SDK_DRIVER_COMPILE:** Compile all drivers below RTK Layer as built-in or module
  - **CONFIG_SDK_DRIVER_RTK:** Select the option to compile the feature code for RTK kernel driver or RTK user library
  - **CONFIG_SDK_DRIVER_RTK_CUSTOMER:** Select the option to compile the customer API code to SDK driver
  - **CONFIG_SDK_DRIVER_NIC:** Select the option to compile the NIC code for SDK kernel driver
  - **CONFIG_SDK_DRIVER_RTDRV:** SDK Linux user/kernel common driver
  - **CONFIG_SDK_DRIVER_RTNIC:** SDK Linux ethernet driver
  - **CONFIG_SDK_DEBUG:** Enable or disable SDK debug module and debug messages. Reduce the code size when you do not select this.
  - **CONFIG_SDK_APP_DIAG:** Compile SDK diagnostic shell.
  - **CONFIG_TRUNK_FAILOVER_HANDLING:** Trunk failover mechanism
  - **CONFIG_SDK_DRIVER_NIC_USER_MODE:** User mode NIC driver
  - **CONFIG_SDK_WA_EEE_COMPATIBLE:** Compiler EEE compatible mechanism.
  - **CONFIG_SDK_PIE_DATA_ENDIAN_BIG:** Select Big Endian input data format for PIE/ACL.
  - **CONFIG_SDK_PIE_DATA_ENDIAN_LITTLE:** Select Little Endian input data format for PIE/ACL.
  - **CONFIG_SDK_WA_RTL8231_RESET:** Compiler RTL8231 reset mechanism.
  - **CONFIG_SDK_RX_THREAD:** Compiler NIC Rx thread mechanism.
  - **CONFIG_SDK_LINKMON_POLLING_MODE:** Compiler LinkMon thread as polling mode.
  - **CONFIG_SDK_LINKMON_ISR_MODE:** Compiler LinkMon thread as interrupt mode.
  - **CONFIG_SDK_LINKMON_MIXED_MODE:** Select the option to enable the LinkMon thread by polling and interrupt

mixed mode

- **CONFIG_SDK_AUTO_COMBO_MEDIA_BY_GPIO**: Detect the GPIO signal and auto configure media of combo port by software thread
- **CONFIG_SDK_WA_88E6063_COMPATIBLE**: Select the option to workaround link up issue with Marvell 88E6063 when connect in Nway 10M
- **CONFIG_SDK_WA_LINKDOWN_PWR_SAVING**: Select the option to workaround link up issue in internal port of RTL8328M/RTL8328S when power saving mode is enabled
- **CONFIG_SDK_WA_BACK_PRESSURE**: Select the option to workaround the back pressure issue of RTL8328M/RTL8328S when duplex is half
- **CONFIG_SDK_WA_INTRALINK_DELAY**: Select the option to workaround intralink delay argument of RTL8328M/RTL8328S when connect to RTL8212F PHY
- **CONFIG_SDK_WA_PKTBUF_WATCHDOG**: Select the option to monitor packet buffer status and trigger packet buffer
- **CONFIG_SDK_WA_RTL833X_COMBO_LED**: Select the option to workaround the combo port led display in MAC RTL833X chip.
- **CONFIG_SDK_SOFTWARE_CONTROL_LED**: Select the option to controls RTL8231 by software driver, and disable hardware control capability.
- **CONFIG_SDK_UART1**: Select the option to enable uart1 function.
- **CONFIG_SDK_PORT_VIEW_PHYSICAL_PORT**: Select the RTK API port view option to physical port
- **CONFIG_SDK_PORT_VIEW_ZERO_BASE_PORT**: Select the RTK API port view option to zero-based port for 838x/833x 8-port/16-port model

Note: Please refer to above symbols and define appropriate symbol in your building scripts. You can use these compiler options to select what you want to build into your runtime image.

Chapter 5 Build SDK Image

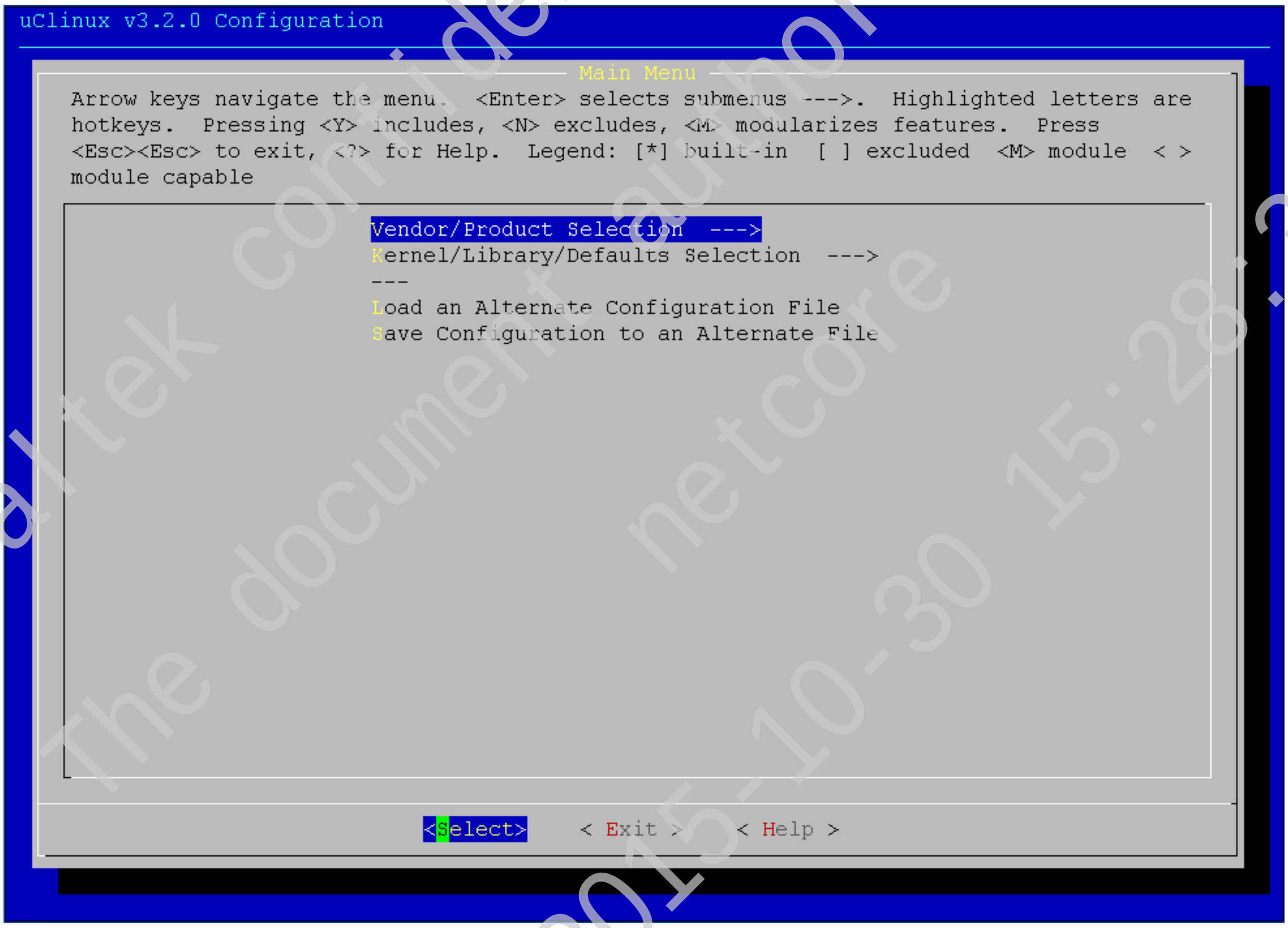
5.1. How to Build a Demo Shell in Linux

- How to build out a runtime image

  Step 1: Goto your working directory.

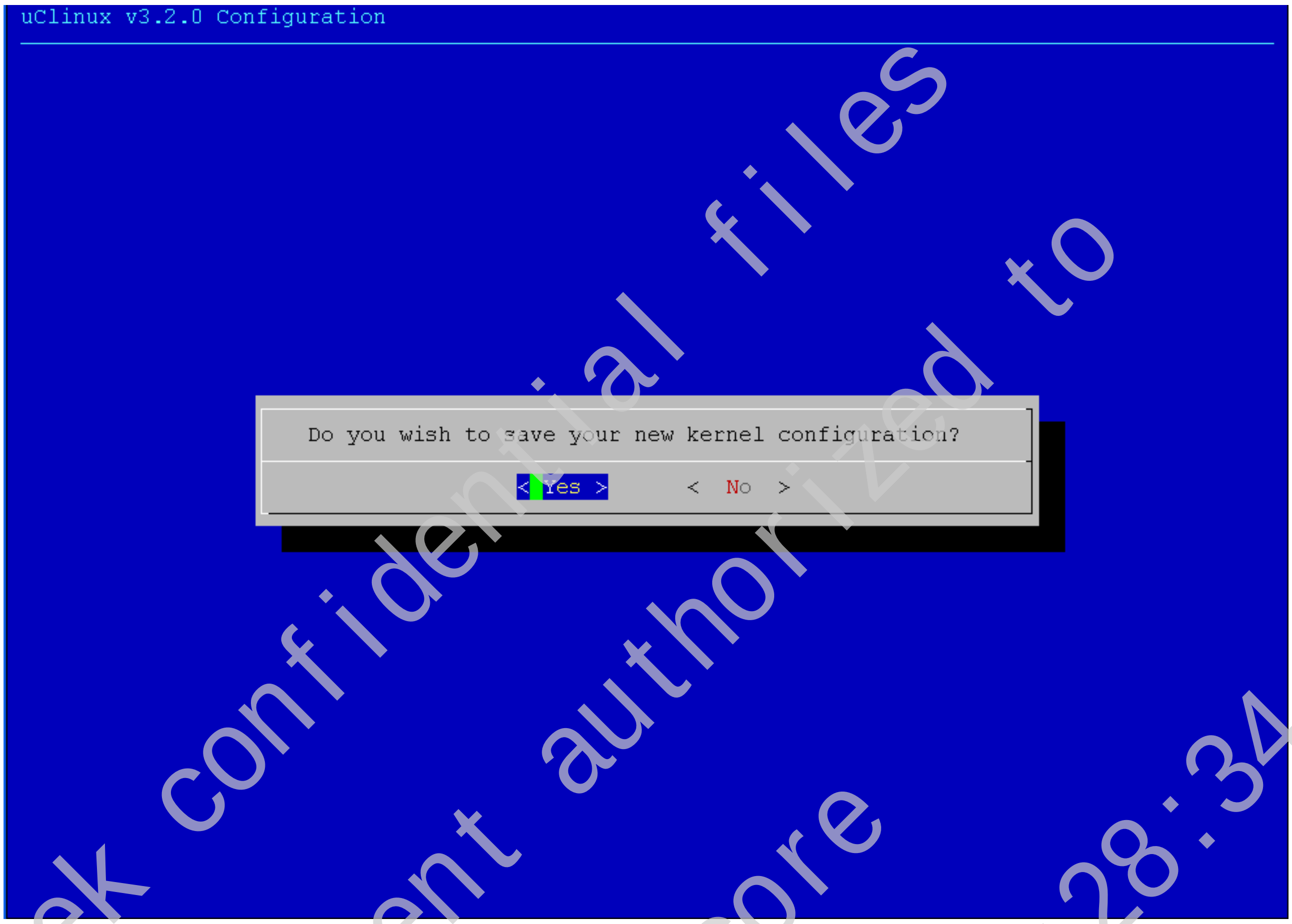
  Step 2: type 'make distclean' to clear all configuration settings.

  Step 3: Type 'make menuconfig' to enter the configuration menu of the distribution.



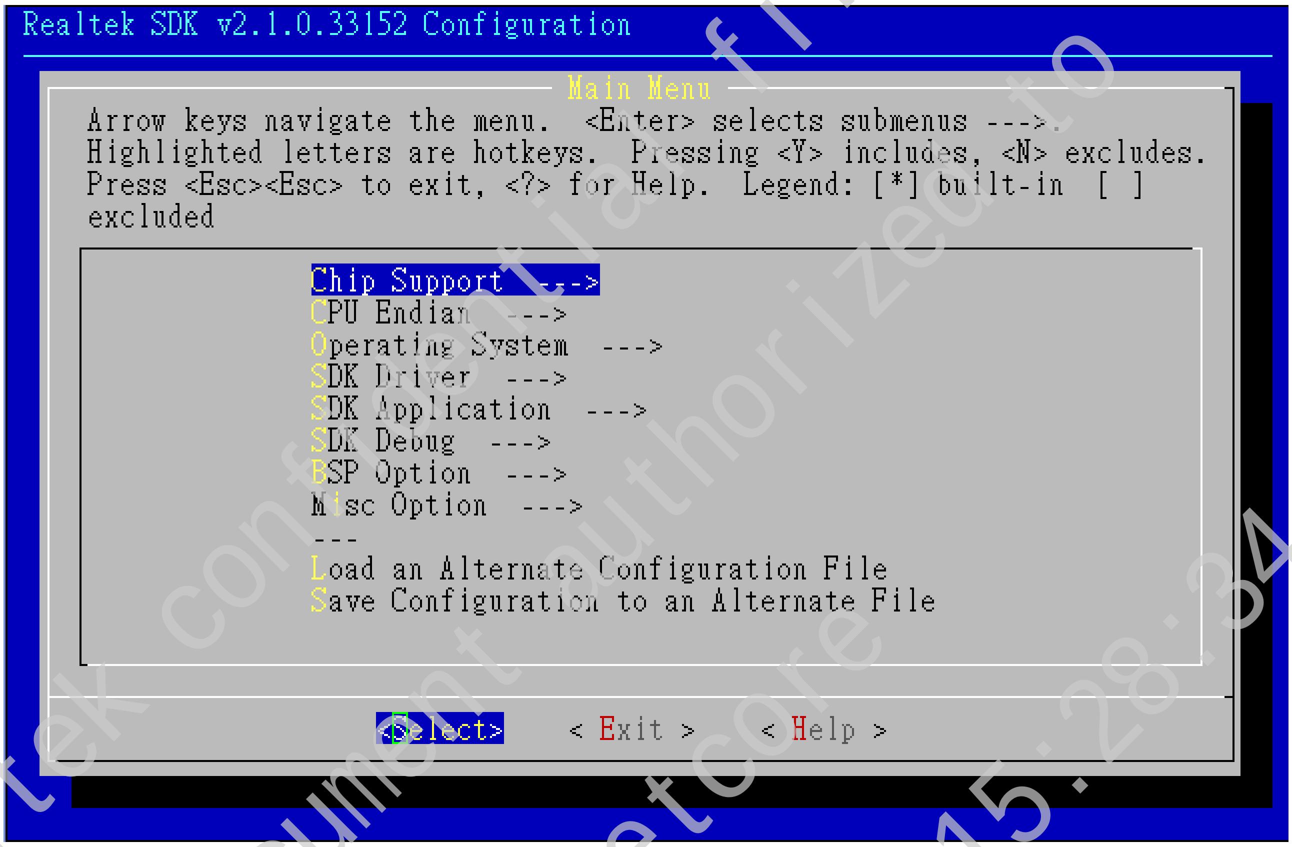
```
uClinux v3.2.0 Configuration
                                Main Menu
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are
  hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press
  <Esc><Esc> to exit, <?> for Help.  Legend: [*] built-in  [ ] excluded  <M> module  < >
  module capable

                       Vendor/Product Selection  --->
                       Kernel/Library/Defaults Selection  --->
                       ---
                       Load an Alternate Configuration File
                       Save Configuration to an Alternate File

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

Step 4: If you want to use default settings, just simply select <Exit> and save the configuration.



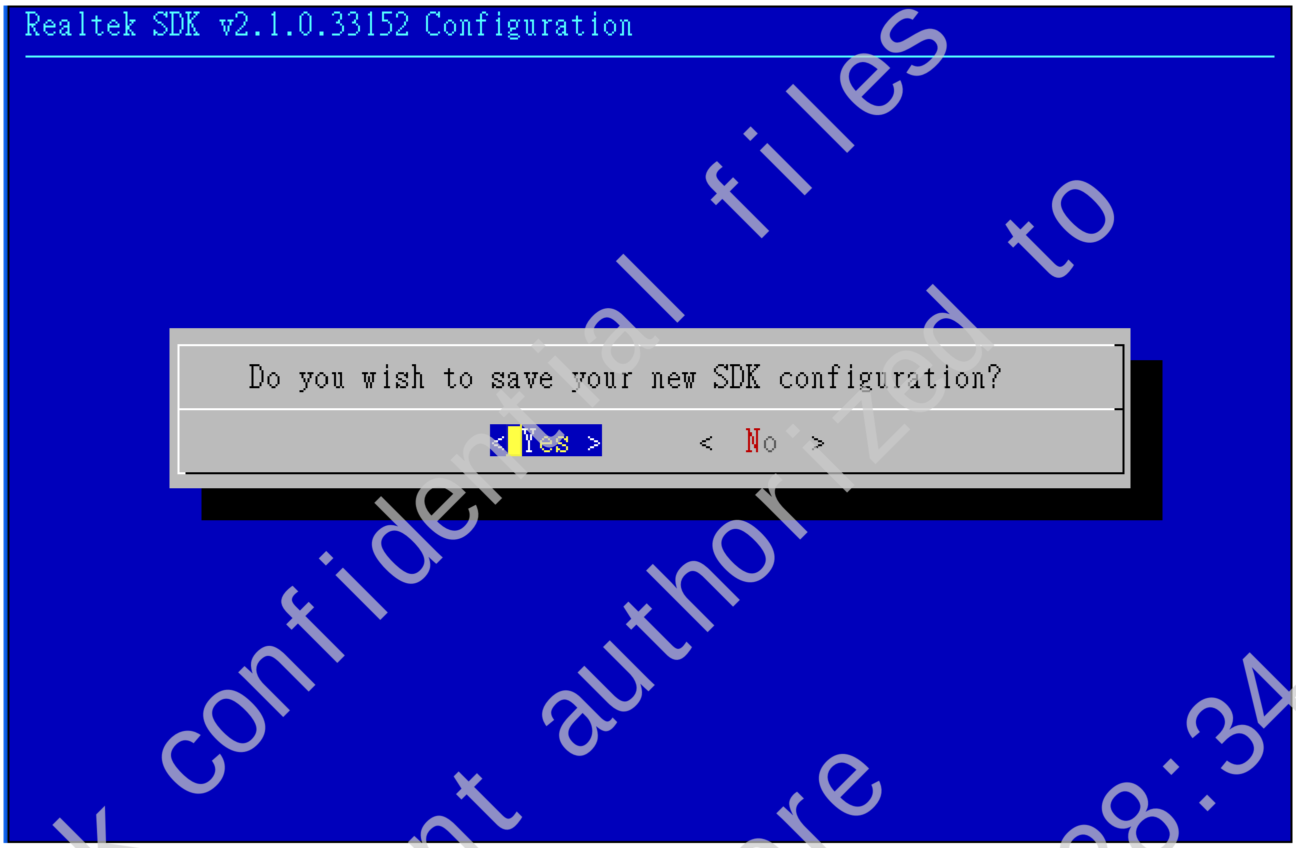
```
uClinux v3.2.0 Configuration

        Do you wish to save your new kernel configuration?

                  < Yes >          <  No  >
```

Step 5: After init procedure of the configuration tool, you will see the SDK menu configuration. In your development stage, you can quickly type 'make sdkconfig' to enter this menu and change any SDK configuration as you wish. Be aware that you always need to type 'make menuconfig' in the first time while you have done 'make distclean'.



```
Realtek SDK v2.1.0.33152 Configuration

                         Main Menu
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes.
  Press <Esc><Esc> to exit, <?> for Help.  Legend: [*] built-in  [ ]
  excluded

              Chip Support  --->
              CPU Endian  --->
              Operating System  --->
              SDK Driver  --->
              SDK Application  --->
              SDK Debug  --->
              BSP Option  --->
              Misc Option  --->
              ---
              Load an Alternate Configuration File
              Save Configuration to an Alternate File

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

Step 6: If you want to use the default settings, select <Exit> and save the configuration.



```
Realtek SDK v2.1.0.33152 Configuration

Do you wish to save your new SDK configuration?

< Yes >        <  No  >
```

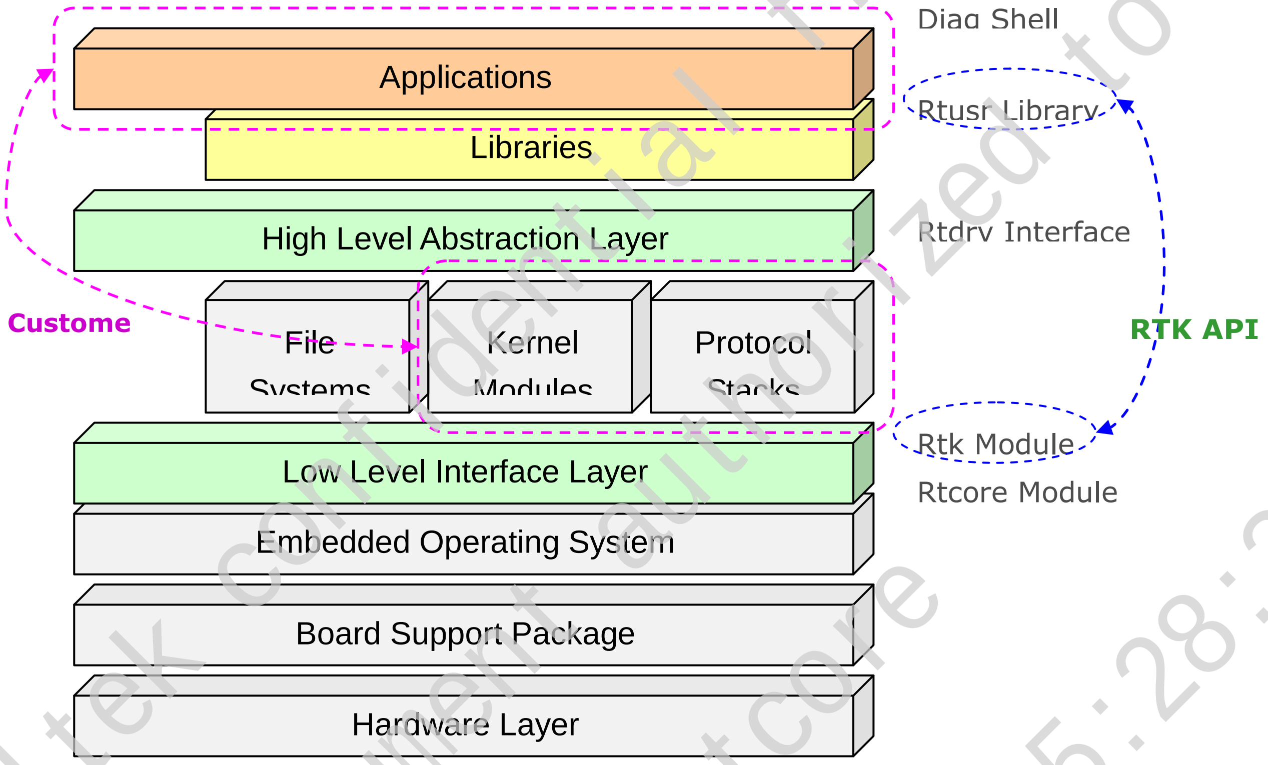
Step 7: After exit from configuration menu, type ‘make’ to build out the runtime image.

Step 8: If build is success, you will see following message from your console. The image size will be different depend on your configuration settings.

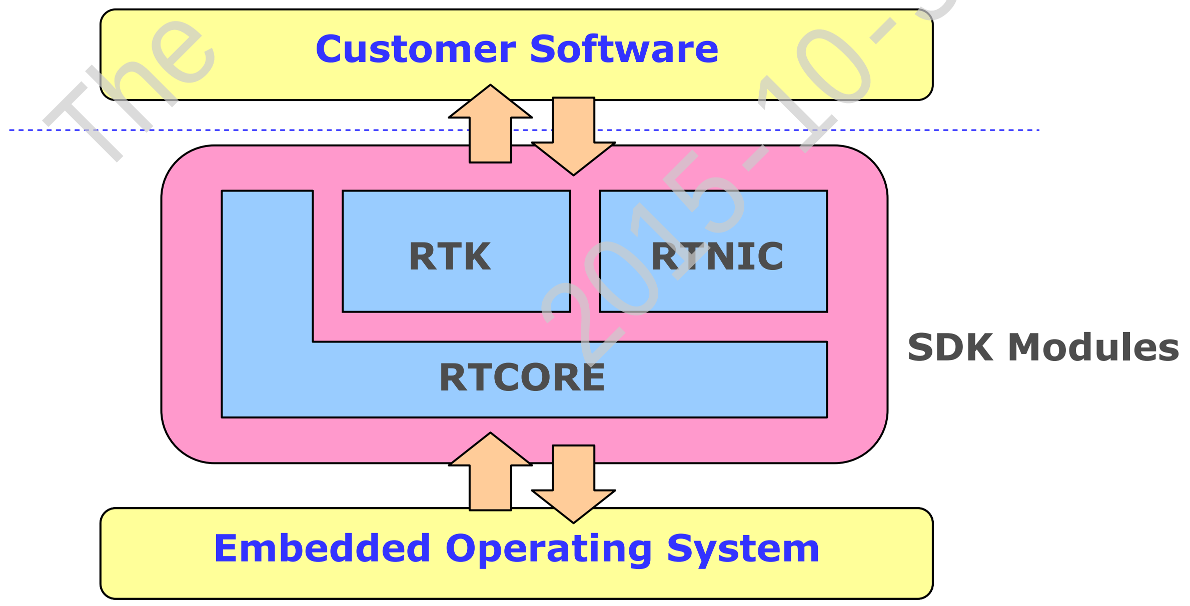
```
Image Name:   Linux Kernel Image
Created:      Wed Oct 14 12:15:10 2009
Image Type:   MIPS Linux Kernel Image (gzip compressed)
Data Size:    1636633 Bytes = 1598.27 kB = 1.56 MB
Load Address: 0x80000000
Entry Point:  0x801FA000
```

Chapter 6 Porting Guideline

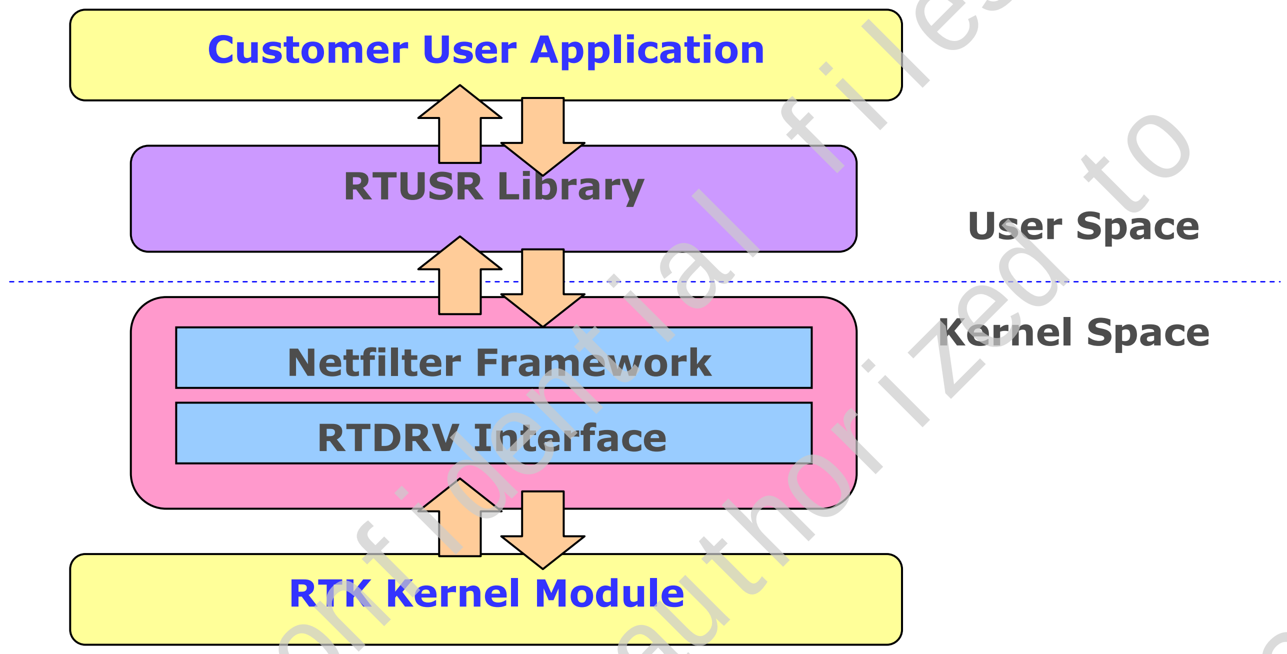
6.1. SDK Architecture



6.2. SDK Common Modules



6.3. SDK Modules for Linux Solution



6.4. Which Directories Needs to be Porting

__For non Linux OS Customer__:

- Implement OSAL body for your OS.

Note: You can refer to source code in sdk/system/osal/linux or sdk/system/osal/ecos.

Following function are used in this SDK now:

- osal/print.h
  - osal_printf

- osal/memory.h
  - osal_alloc
  - osal_free
  - osal_mmap
  - osal_munmap

- osal/isr.h
  - osal_isr_register

  - osal_isr_unregister

- osal/cache.h
  - osal_cache_memory_flush

- osal/sem.h
  - osal_sem_mutex_create
  - osal_sem_mutex_destroy
  - osal_sem_mutex_take
  - osal_sem_mutex_give
  - osal_sem_create
  - osal_sem_destroy
  - osal_sem_take
  - osal_sem_give

- osal/lib.h
  - osal_strlen
  - osal_strcmp
  - osal_strcpy
  - osal_memset
  - osal_memcpy
  - osal_memcmp

- osal/thread.h
  - osal_thread_create
  - osal_thread_destroy
  - osal_thread_self
  - osal_thread_name
  - osal_thread_exit

- osal/time.h
  - osal_time_usec2Ticks_get
  - osal_time_usecs_get
  - osal_time_seconds_get
  - osal_time_usleep
  - osal_time_sleep
  - osal_time_udelay
  - osal_time_mdelay

- osal/spl.h
  - osal_spl_ spin_lock
  - osal_spl_ spin_unlock

- osal/inet.h
  - osal_ntohs
  - osal_ntohl
  - osal_htons
  - osal_htonl

- osal/atomic.h
  - osal_atomic_inc
  - osal_atomic_set
  - osal_atomic_dec_return
  - osal_atomic_read

- osal/wait.h
  - osal_init_waitqueue_head
  - osal_wait_event_interruptible
  - osal_wake_up_interruptible

- osal/workqueue.h
  - osal_list_del
  - osal_list_add_tail
  - osal_schedule_work

For All Customers:

- Implement your upper layer driver code to call RTK layer APIs

Ref.: RTK_MS_SDK_API_Document_2.1.4.53590.pdf

- Prepare the nic packet size, allocate, free function when call drv_nic_init
- Prepare your receive callback function: drv_nic_rx_cb_f with cookie.
- Prepare your transmit callback function: drv_nic_tx_cb_f with cookie.

Ref.: sdk/system/linux/rtnic/rtnic_drv.c
sdk/system/linux/rtnic/rtnic_drv.h

6.5. SDK Initial Sequence

- Initialize your OS and BSP.
- Initialize RTCORE module – OS depend driver module
  For non Linux os
  - ■ Call rtcore_init (uint32 unit) function to initialize RTCORE module.

  For Linux os
  - ■ Load the RTCORE module
- Initialize SDK module – SDK kernel driver module
  For non Linux os
  - Call rtk_init (uint32 unit) function to initialize SDK.

  For Linux os
  - Load the SDK kernel driver module
- Initialize NIC module Part
  For non Linux os
  - Call drv_nic_init (uint32 unit, drv_nic_initCfg_t *pInitCfg) function
  - Call drv_nic_rx_register function.
  - Call drv_nic_rx_start function.

  For Linux os
  - Load the NIC module - reference rtnic_drv.c.

Ref.: RTK_MS_SDK_API_Document_2.1.4.53590.pdf

- Initialize application module - SDK user/kernel interface module

  For non Linux os

  - Not support yet.

  For Linux os

  - Load the SDK user/kernel interface module

Realtek Semiconductor Corp.

Headquarters

No. 2, Innovation Road II

Hsinchu Science Park, Hsinchu 300, Taiwan

Tel.: +886-3-578-0211. Fax: +886-3-577-6047

www.realtek.com