



www.robustel.com

RobustOS SDK Developer Guide

Document Name: RobustOS Developer Guide

Version: 5.0.3

Date: 2025-01-20

Status: Confidential

Contents

- Chapter 1 Document Revision History 4
- Chapter 2 Overview 5
- Chapter 3 Basic Information 6
 - 3.1 Supported Hardware 6
 - 3.2 Supported Programming Languages 6
 - 3.3 Recommended Development Operating Systems 7
 - 3.4 Recommended IDE Tools 7
 - 3.5 RobustOS Software Architecture 7
 - 3.6 C Library 8
 - 3.7 Hardware Information for SDK 8
 - 3.8 Hardware Resources 9
 - 3.8.1 R2000 Lite 9
 - 3.8.2 R3000 Lite 9
 - 3.8.3 R3000 9
 - 3.8.4 R3000 (with 4xDI Version) 9
 - 3.8.5 R3000 Quad 10
 - 3.8.6 R3000 LG 10
 - 3.8.7 R3010 10
 - 3.8.8 R2110 & R5020 10
 - 3.8.9 M1200 10
 - 3.8.10 M1201 11
 - 3.8.11 R3020 11
 - 3.8.12 R1511PL 11
 - 3.8.13 R1500 11
 - 3.8.14 R1510 11
 - 3.8.15 R1511 11
 - 3.8.16 R1520 11
 - 3.8.17 R2000 Ent 12
 - 3.8.18 ET8012 12
 - 3.8.19 ET8013 12
 - 3.8.20 ET8015 12
 - 3.8.21 R2010 12
- Chapter 4 Preparation for Development 13
 - 4.1 SDK installation instructions 13
 - 4.1.1 Docker Desktop Direction 13
 - 4.1.2 Install SDK 14
 - 4.2 The Directory Structure 15
 - 4.3 RobustOS Makefile Introduction 16
 - 4.3.1 BuildPackage Variables 16
 - 4.3.2 BuildPackage Defines 16
 - 4.3.3 Other Important Variables 17
- Chapter 5 Create a RobustOS APP 18
 - 5.1 Create Helloworld Package 18
 - 5.2 Add Source Code 18

- 5.3 Add SDK Script 18
- 5.4 Add Makefile for SDK Building System20
- 5.5 Build Helloworld Package 21
- 5.6 Install APP 21
- 5.7 Check APP Running Status 22
- 5.8 APP Package Naming Rules 22
- Chapter 6 Integrate into RobustOS Web Manager 23
 - 6.1 Configuration Definition 23
 - 6.2 Status Definition 24
 - 6.3 Webpage Layout 25
 - 6.4 UCI Attributes 25
 - 6.5 UCI Tag 26
 - 6.6 How to Get Configuration Settings 27
 - 6.7 How to Change Configuration Settings 28
 - 6.8 How to Find UCI Tags 28
 - 6.9 USI Tag 29
 - 6.10 How to Update Status 29
 - 6.11 How to Get Status 29
 - 6.12 How to Find USI Tags 31
- Chapter 7 SDK C-Libraries 32
 - 7.1 librouter data type 32
 - 7.2 librouter Log API 32
 - 7.3 librouter UCI API 34
 - 7.4 librouter Environment Variable API 41
 - 7.5 librouter IPC API 43
 - 7.6 librouter Event API 45
- Chapter 8 Pre-installed Tools 47
 - 8.1 Open Source Libraries 47
 - 8.2 Open Source Commands 47
 - 8.3 RobustOS Private Commands 47
 - 8.3.1 uci 47
 - 8.3.2 usi 48
 - 8.3.3 renv 48
 - 8.3.4 sms 48
- Chapter 9 Programming Tutorials 49
 - 9.1 How to Debug the Application 49
 - 9.1.1 Use Syslog 49
 - 9.1.2 Use Shell with Root Privileges 49
 - 9.1.3 Use tftp 49
 - 9.1.4 Use gdbserver 49
 - 9.1.5 Debugging under x86/x64 52
 - 9.2 How to Get System Status 52
 - 9.3 How to Read/Write Data via Serial Port 52
 - 9.4 How to Get DI Status 52
 - 9.5 How to Trigger DO Changes 54
 - 9.6 How to Get AI Status 54

- 9.7 How to Subscribe Event 54
 - 9.7.1 Supported Events 54
 - 9.7.2 Code Example 56
- 9.8 How to Send SMS 57
- 9.9 How to Send Email 57
- 9.10 How to Update Firmware of Router 57
- 9.11 How to Import/Export XML Configuration File 58
- 9.12 How to Pack a C++ Application 58
- 9.13 How to Control BLE Module 58
- 9.14 How to Make RobustOS App Realize File Upload Feature 60
 - 9.14.1 Determine js File Path of App Configuration Page 60
 - 9.14.2 JS File Description 60
 - 9.14.3 add_area_border 61
 - 9.14.4 Modify JS File 61
 - 9.14.5 Modify uci.xml 64
 - 9.14.6 Replace Default JS file when Device Starts 64
 - 9.14.7 Add js File to Source Code 65
 - 9.14.8 Add js File to rpk Installer 65
 - 9.14.9 File Upload Process 66
- Chapter 10 Sample Packages 67
 - 10.1 RobustOS Sample Packages 67
 - 10.2 Common Libraries 68
 - 10.3 Cloud Platform SDK Libraries 69
 - 10.4 Package dependencies 70
- Chapter 11 RobustOS Open Source Components 71

Chapter 1 Document Revision History

Revision	Author	Date	Description
1.0.0	Zhangliang Huang	2016-08-01	Initial version.
1.2.0	Zhangliang Huang	2018-05-09	Update.
1.2.1	Zhifeng Liao	2019-03-25	Add a description of sample packages, device hardware resource updates, etc.
1.3.0	Zhifeng Liao	2019-03-27	Add a description about UCI and USI.
1.4.0	Zhifeng Liao	2019-05-23	Add Chapter3.5 about RobustOS software architecture. Update Chapter9.4 about how to get DI status. Add Chapter9.8 about how to send Emails.
1.5.0	Zhifeng Liao	2019-08-05	Add BLE module for R2110.
	Zhao Wei	2020-07-10	Add support for new hardware models.
1.6.0	Tang Zhen	2020-10-10	Add sample_ai for R1520.
5.0.0	Lv Liang	2023-10-25	Add hardware resource info: R2000 Ent, R3020, R1511PL, ET801x, and R201x. Add 4.1.1 Docker Desktop Direction. Add 9.14 How to Make RobustOS App Realize File Upload Feature. Remove hardware resource info of MEG5000. Remove Python.
5.0.1	Lv Liang	2024-01-11	Update docker operation instructions. Add 10.4 Package dependencies.
5.0.2	Jeff Zhou	2024-07-05	Add Chapter 11 RobustOS Open Source Components.
5.0.3	ZhiXiong Xie	2025-01-20	Add support for R5010a

Chapter 2 Overview

Robustel's series router allows 3rd party to develop their applications. We provide a Software Development Kit (SDK), which offers a simple and fast way to implement customer-specific functions and applications.

If you have any problems in the programming process, please email support@robustel.com. We will provide technical support in various aspects.

Chapter 3 Basic Information

3.1 Supported Hardware

- R3000 (Firmware version should be v2.0.0 or above)
- R3000 Lite
- R3000 Quad
- R3000 (with 4xDI version)
- R3000 LG
- R3010
- R3020
- R2000
- R2000 Dual
- R2000 Lite
- R2000 Ent
- R2110
- R2100
- R2010
- R2011
- M1200
- M1201
- ET8012
- ET8013
- ET8015
- R1500
- R1510
- R1510 Lite
- R1511
- R1511P
- R1511PL
- R1520
- R5020
- R5010a

3.2 Supported Programming Languages

- C/C++
- Linux Shell Scripts

3.3 Recommended Development Operating Systems

Before starting the application development, you need to install SDK in your development environment, and it should be a Linux system, below two Linux operating systems are recommended. For SDK installation, please refer to chapter 3.1.

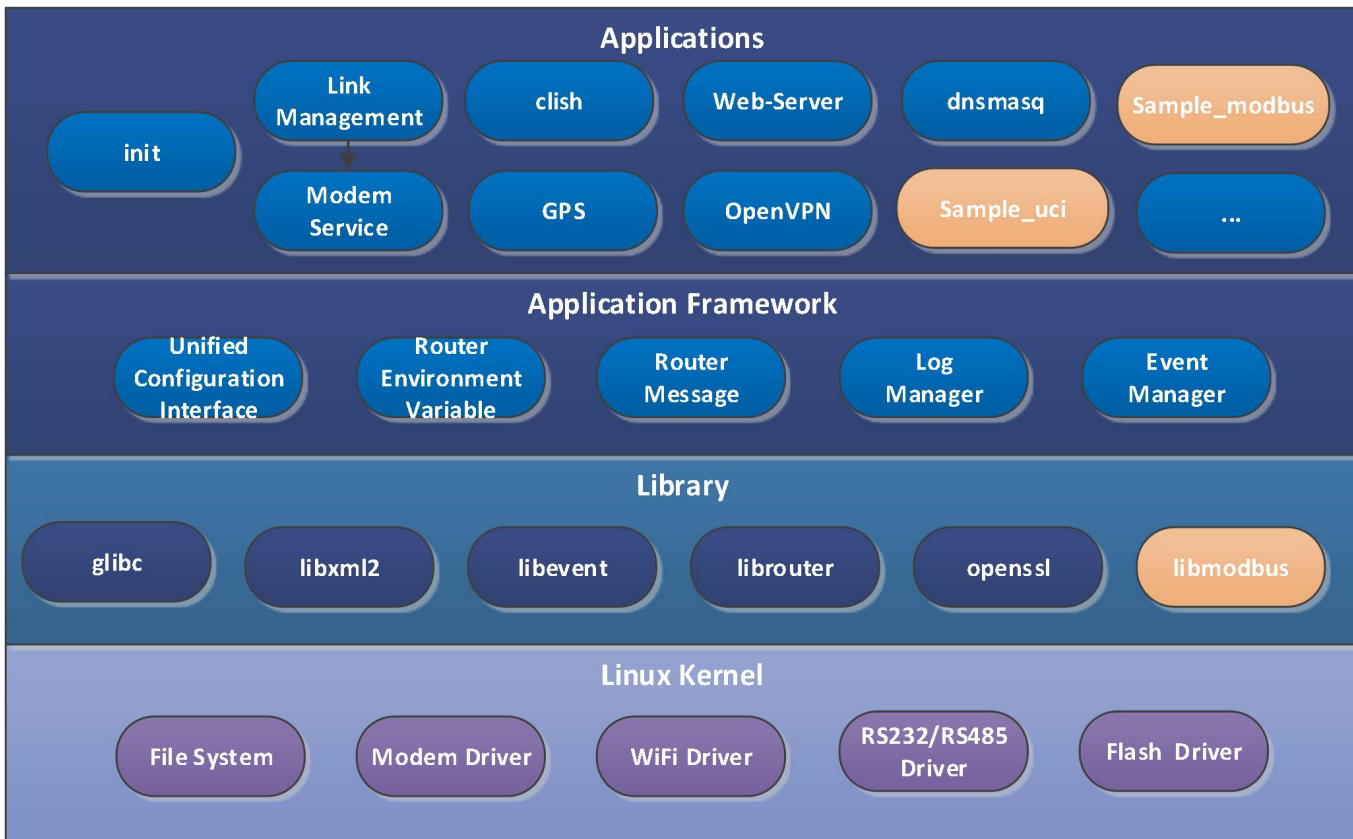
- CentOS 7
- Ubuntu 16.04 (change default shell to bash)

3.4 Recommended IDE Tools

- Vi
- Nodepad++

3.5 RobustOS Software Architecture

RobustOS is the software that runs on Robustel’s series router. As shown in the diagram below, RobustOS can be divided into four layers: Linux kernel, library, application framework, and applications. The SDK package provides the middle two layers (Library and Application Framework). Developers can implement their applications based on the SDK package.



3.6 C Library

- glibc

3.7 Hardware Information for SDK

	R3000 series	R2000 series	R3010	R2110	R2100	R2012
CPU	Atmel ARM9	Atheros SoC	Atmel ARM9	MTK SoC	MTK SoC	Qualcomm SoC
ARCH	ARM	MIPS	ARM	MIPS	MIPS	MIPS
Total Flash	256MB	16MB	256MB	32MB	32MB	32MB
Total RAM	128MB	64MB	128MB	512MB	512MB	128MB
Flash available for SDK	64MB	4MB	64MB	5MB	5MB	5MB
RAM available for SDK	64MB	16MB	64MB	256M	256M	256M

	M1200	M1201	ET8012	ET8013	ET8015
CPU	ARM9	ARM9	Qualcomm SoC	Qualcomm SoC	Qualcomm SoC
ARCH	ARM	ARM	MIPS	MIPS	MIPS
Total Flash	16MB	16MB	16MB	16MB	16MB
Total RAM	64MB	64MB	128MB	128MB	128MB
Flash available for SDK	3M	3M	4MB	4MB	4MB
RAM available for SDK	16MB	16MB	64M	64M	64M

	R1500	R1510	R1511	R1511PL	R5020
CPU	ARM9	Qualcomm SoC	Qualcomm SoC	Qualcomm SoC	MTK SoC
ARCH	ARM	MIPS	MIPS	MIPS	MIPS
Total Flash	16MB	16MB	16MB	16MB	32MB
Total RAM	64MB	128MB	128MB	128MB	512MB
Flash available for SDK	3MB	4MB	4MB	4MB	5MB
RAM available for SDK	16MB	64MB	64MB	64M	256M

	R3020	R2010	R2011	R1510 Lite	R1520
CPU	Atmel ARM9	Qualcomm SoC	Qualcomm SoC	Qualcomm SoC	Qualcomm SoC
ARCH	ARM	MIPS	MIPS	MIPS	MIPS
Total Flash	256MB	32MB	32MB	16MB	16MB
Total RAM	128MB	128MB	128MB	128MB	128MB
Flash available for SDK	64MB	4MB	4MB	4MB	4MB
RAM available for SDK	64MB	64M	64M	64MB	64M

	R5010a	
CPU	ARM Cortex-A7	
ARCH	ARM	
Total Flash	110MB	
Total RAM	192MB	
Flash available for SDK	10MB	
RAM available for SDK	45MB	

3.8 Hardware Resources

3.8.1 R2000 Lite

- `/dev/ttyCOM1` device file of serial port

3.8.2 R3000 Lite

- `/dev/ttyCOM1` device file of first serial port
- `/dev/ttyCOM2` device file of second serial port
- `/mnt/usb` USB disk mount point

3.8.3 R3000

- `/dev/ttyCOM1` device file of first serial port
- `/dev/ttyCOM2` device file of second serial port
- `/dev/ttyGPS` device file of GPS data port
- `/mnt/usb` USB disk mount point
- `/mnt/sd` SD card mount point
- `/dev/DI1` device file of first DI
- `/dev/DI2` file of second DI
- `/sys/class/leds/do1/` file path of first DO
- `/sys/class/leds/do2/` file path of second DO

3.8.4 R3000 (with 4xDI Version)

- `/dev/ttyCOM1` device file of first serial port
- `/dev/ttyCOM2` device file of second serial port
- `/dev/ttyGPS` device file of GPS data port
- `/mnt/usb` USB disk mount point
- `/mnt/sd` SD card mount point
- `/dev/DI1` device file of first DI
- `/dev/DI2` device file of second DI
- `/dev/DI3` device file of third DI

- `/dev/DI4` device file of fourth DI

3.8.5 R3000 Quad

- `/dev/ttyCOM1` device file of RS232 serial port
- `/dev/ttyCOM2` device file of RS485 serial port
- `/dev/ttyGPS` device file of GPS data port
- `/mnt/usb` USB disk mount point
- `/mnt/sd` SD card mount point

3.8.6 R3000 LG

- `/dev/ttyCOM1` device file of RS232/RS485 serial port
- `/dev/ttyGPS` device file of GPS data port
- `/dev/DI1` device file of DI
- `/mnt/usb` USB disk mount point
- `/mnt/sd` SD card mount point

3.8.7 R3010

- `/dev/ttyCOM1` device file of RS232 serial port
- `/dev/ttyCOM2` device file of RS485 serial port
- `/mnt/usb` USB disk mount point

3.8.8 R2110 & R5020

- `/dev/ttyCOM1` device file of RS232 serial port
- `/dev/ttyCOM2` device file of RS485 serial port
- `/dev/ttyGPS` device file of GPS data port
- `/dev/ttyXRUSB1` Serial port for BLE654
- `/dev/DI1` device file of DI
- `/sys/class/leds/do1/` file path of first DO
- `/mnt/usb` USB disk mount point
- `/mnt/sd` SD card mount point

3.8.9 M1200

- `/dev/ttyCOM1` device file of RS232 serial port
- `/dev/ttyCOM2` device file of RS485 serial port
- `/sys/class/leds/do1/` file path of first DO

3.8.10 M1201

- `/dev/ttyCOM1` device file of RS232 serial port
- `/dev/ttyCOM2` device file of RS232/RS422/RS485 serial port
- `/dev/ttyCOM3` device file of RS485 serial port (optional)

3.8.11 R3020

- `/dev/ttyCOM1` device file of RS485 serial port

3.8.12 R1511PL

- `/dev/ttyCOM1` device file of RS232 serial port

3.8.13 R1500

- `/dev/ttyCOM1` device file of RS232 serial port
- `/dev/ttyCOM2` device file of RS232 serial port

3.8.14 R1510

- `/dev/DI1` device file of first DI
- `/sys/class/leds/do1/` file path of first DO

3.8.15 R1511

- `/dev/ttyCOM1` device file of RS232/RS485 serial port

3.8.16 R1520

- `/dev/ttyCOM1` device file of RS232 serial port
- `/dev/ttyCOM2` device file of RS485 serial port
- `/dev/ttyGPS` device file of GPS data port (optional)
- `/dev/DI1` device file of DI
- `/sys/class/leds/do1/` file path of first DO

- `/dev/AI1` device file of AI
- `/mnt/usb` USB disk mount point

3.8.17 R2000 Ent

- `/dev/ttyCOM1` device file of RS232 serial port

3.8.18 ET8012

- `/dev/ttyCOM1` device file of RS485 serial port

3.8.19 ET8013

- `/dev/ttyCOM1` device file of RS485 serial port

3.8.20 ET8015

- `/dev/ttyCOM1` device file of RS485 serial port

3.8.21 R2010

- `/dev/ttyCOM1` device file of RS485 serial port

Chapter 4 Preparation for Development

4.1 SDK installation instructions

4.1.1 Docker Desktop Direction

- Docker introduce:

The concept of Docker image is similar to the mirror in a virtual machine (e. g. ISO files), is a read-only template, a separate file system that includes the data needed to run the container, and can be used to create new containers.

Docker Container is a running instance created by Docker image similar to VM virtual machine, supports start, stop, delete, etc. Each container is isolated from each other, and it runs specific applications, containing specific application code and required dependency files.

- Docker Desktop Import the Docker image

When starting the docker program, open the PowerShell command line window, switch to the directory where the ros-sdk compressed package file is located, and execute the following command:

```
# docker load -i ros-sdk.tar
```

For R5010a

```
# docker load -i ros-sdk-5.0.0.tar
```

- volume

Method 1:

```
# docker run -v ros-sdk:/root/ROS-SDK -v /e/tmp:/root/output --name ros-sdk -it ros-sdk
```

For R5010a

```
# docker run -v ros-sdk-r5010a:/root/ROS-SDK -v /e/tmp:/root/output --name ros-sdk-r5010 -it  
ros-sdk-r5010a:5.0.0
```

The first -v parameter ros-sdk:/root/ROS-SDK is a docker managed volume to ensure that the following modifications in the /root/ROS-SDK directory will not be lost at the next start, which can be viewed in the PowerShell command line window with the following command:

```
# docker volume ls
```

The second -v parameter /e/ tmp:/root/output is to mount the Windows E:\tmp directory to the /root/output directory of the docker container, so that the compiled installation package can be copied to the Windows system.

--name can specify the name of the container, without rebuilding, you can start the previous container directly the next time.

Method 2:

Copy the files in the specified container to the host

```
# docker cp <containerID>:root\ROS-SDK\xxx e:\tmp
```

Copy the files in the host to the specified container

```
# docker cp e:\tmp\xxx <containerID>:root\ROS-SDK\
```

- Docker commands

View all of the containers:

```
# docker ps -a
```

Delete container:

```
# docker rm <containerID>
```

Delete all of the containers:

```
# docker stop $(docker ps -a -q)
# docker rm $(docker ps -a -q)
```

Run the container:

```
# docker start <containerID>
# docker attach <containerID>
```

After entering the container, you can type `ctrl + Q + P` to exit the container, then the container continues after exit; when type `exit`, the container exits and stops.

Stop the container:

```
# docker stop <containerID>
```

View all images:

```
# docker images
```

Delete image:

before delete the image you should stop and delete corresponding container

```
# docker rmi <imageID>
```

4.1.2 Install SDK

The cross-compile toolchain is distributed with SDK library, and packed into a compressed file. You can download the `ros-dev.tar` compact package from the Cloud and import the docker image. We use the `REPOSITORY:TAG` to define the different images, the image of SDK is defined as `ros-dev:latest`, which includes other platforms that you can switch between in the docker container, for example:

```
# docker load -i ros-sdk-5.0.1.tar
# docker run --name ros-sdk -v ros-sdk:/root/ROS-SDK -it ros-sdk:5.0.1
# cd ROS-SDK
# make r3000
```

Example for R5010a device:

```
# docker load -i ros-sdk-5.0.0.tar
# docker run -v ros-sdk-r5010a:/root/ROS-SDK --name ros-sdk-r5010 -it ros-sdk-r5010a:5.0.0
# cd ROS-SDK
# make r5010a
```

There are many benefits to integrating the toolchain with the SDK. You will automatically get the same compile flags with the firmware. And you do not need to specify the included path and link path for the SDK header and library files, which dramatically saves your efforts.

There are some examples distributed with the SDK. You can build an SDK example for testing after installation, for example:

```
# cd ROS-SDK
```

```
# make package/sample_shell/install
```

The APP installation file will be put into bin/r3000/packages directory after successful building.
Use the next command in PowerShell to copy the rpkg to host.

```
# docker cp ros-sdk:/root/ROS-SDK/bin/r3000/packages/r3000-sample_shell-1.0.0.rpk .
```

4.2 The Directory Structure

```
ROS-SDK
├── bin
│   └── r3000
├── build_dir
│   └── target-r3000
├── include
│   ├── check_uci.py
│   ├── host-build.mk
│   ├── platform.mk
│   ├── cmake.mk
│   ├── image.mk
│   ├── configure.mk
│   ├── package.mk
│   ├── quilt.mk
│   ├── rpkg-build.sh
│   ├── rstrip.sh
│   ├── unpack.mk
│   └── web_language_js_gen
├── Makefile
├── package
│   ├── sample_shell
│   ├── sample_uci
│   └── Makefile
├── rules.mk
├── rules
│   ├── rules_m1200.mk
│   ├── rules_r1500.mk
│   ├── rules_r1510.mk
│   ├── rules_r2110.mk
│   ├── rules_r3000.mk
│   └── rules_r2000.mk
├── staging_dir
│   ├── host
│   └── target-r3000
└── tarball
```

- **bin** - Where the .rpk package files will be generated.
- **build_dir** - Where all packages will be cross-compiled. The subdirectory target- <platform> represents the

compiled directory for the target platform.

- **include** - Some common makefiles and scripts used for package management.
- **package** - The RobustOS Makefiles and patches for all the packages. The RobustOS Makefile has its own syntax, different from the conventional Makefile of Linux make tool. The RobustOS Make file defines the meta information of the package, how to compile, where to install the compiled binaries, etc. See XXX for more detail.
- **rules** - Where to store the Makefile that compile for different platforms.
- **staging_dir/host** - Where the host tools need for cross-compilation will be installed.
- **staging_dir/target-<platform>** - Where to install the cross-compilation tools, the executable binary files, the library files, and the header files. Cross-compilation will search headers and library files from this directory.
- **tarball** - Where to store the compressed source codes.

4.3 RobustOS Makefile Introduction

An application is called as a package in the SDK build system. The package Makefile is vital because it provides the necessary steps to compile the package. The Makefile has been transformed into an object-oriented template that simplifies the entire ordeal. Everything is hidden in other Makefiles and abstracted to the point where you only need to specify a few variables and definitions.

4.3.1 BuildPackage Variables

- **PKG_NAME** - The name of the package.
- **PKG_VERSION** - The version of the package.
- **PKG_SOURCE** - The filename of the sources, which was stored in the tarball directory.
- **PKG_BUILD_DIR** - Where to compile the package.
- **PKG_MAKE_INSTALL** - Setting it to "1" will call the package's original "make install" with the prefix set to PKG_INSTALL_DIR.
- **PKG_INSTALL_DIR** - (optional) Default to PKG_BUILD_DIR/install_dir.
- **PKG_BUILD_DEPENDS** - (optional) Packages that need to be built before this package.
- **PKG_UNPACK** - (optional) If you are working on local source code rather than archive file, set this variable to override the default unpack command.

4.3.2 BuildPackage Defines

- **Package/Install** - A set of commands to copy files into the rpkg, represented by the \$(1) directory.
- **Package/Install/Develop** - (optional) Copy libs and headers needed to compile packages against it. If you are building a library, this should be defined.
- **Package/Configure** - (optional) You can override the default definition if the source has a particular configuration script. Add your arguments to CONFIGURE_ARGS, which should be worked on in most cases.
- **Package/Compile** - (optional) In most cases, you should leave this undefined because then the default is used, which calls make.

4.3.3 Other Important Variables

- `CONFIGURE_ARGS` - Add extra arguments for the configuration script in this way “`CONFIGURE_ARGS += xxx`”
- `TARGET_CFLAGS` - Add extra compile flags for target compiling in this way “`TARGET_CFLAGS += xxx`”
- `TARGET_LDFLAGS` - Add extra link flags for target compiling in this way “`TARGET_LDFLAGS += xxx`”
- `TARGET_CPPFLAGS` - Add extra preprocessor flags for target compiling in this way “`TARGET_CPPFLAGS += xxx`”

Chapter 5 Create a RobustOS APP

Here is a step-by-step guide for building helloworld package with RobustOS SDK.

5.1 Create Helloworld Package

```
$ tar jxf ROS-SDK-r3000-XX.tar.bz2 /opt
$ cd /opt
$ mkdir package/helloworld
$ mkdir package/helloworld/src
```

5.2 Add Source Code

Edit helloworld.c and save it in /opt/package/helloworld/src.

```
#include <syslog.h>
int main()
{
    syslog(LOG_DEBUG, "hello world");
    return 0;
}
```

Edit Makefile and save it in /opt/package/helloworld/src.

```
helloworld:
    $(CC) helloworld.c -o helloworld
```

5.3 Add SDK Script

In addition, to compile the program, you should write an SDK script to tell the router system how to start and stop the program. The SDK provides some functions so you can more easily do the script, which is located in `usr/lib/functions`. There are several commands you should support in the script.

- **start** - basic command, start the program.
- **stop** - basic command, stop the program.
- **restart** - stop and restart the program.
- **status** - check whether the program is running or stopped.
- **on_startup** - run on system startup, usually run start command.
- **on_reboot** - run before system reboot, usually run stop command.
- **on_link_up** - run when the primary link is up, usually run start command.
- **on_link_down** - run when the primary link is down, usually run stop command.
- **status_sync** - (optional) run when someone query status, used for sync status from other file to use format.

Add a SDK script to tell the router how to start or stop the application.

```
$ mkdir package/helloworld/files
```

```
$ cp package/example2/files/sdk.sh package/helloworld/files/  
$ vi package/helloworld/files/sdk.sh
```

Edit `sdk.sh` based on the example, change the `PROG` variable is enough for this application.

```
#!/bin/sh  
  
. /usr/lib/functions  
  
RETVAL=0  
PROG=helloworld  
  
start()  
{  
    ${PROG}  
    return $?  
}  
  
stop()  
{  
    killproc ${PROG}  
    RETVAL=$?  
    return $RETVAL  
}  
  
restart()  
{  
    stop  
    start  
}  
  
status()  
{  
    pidof -o %PPID ${PROG} >/dev/null 2>&1  
    RETVAL=$?  
    [ $RETVAL -eq 0 ] && echo ${PROG} is running || echo ${PROG} is stopped  
    return $RETVAL  
}  
  
status_sync()  
{  
    return 0  
}  
  
case "$1" in  
    start)
```

```
    start
    ;;
stop)
    stop
    ;;
restart)
    restart
    ;;
status)
    status
    ;;
on_startup)
    start
    ;;
on_link_up)
    stop
    start
    ;;
on_link_down)
    stop
    ;;
on_reboot)
    stop
    ;;
status_sync)
    status_sync
    ;;
*)
    echo $"Usage: $0 {start|stop|restart|status|on_startup|on_link_up|on_link_down|on_reboot|status_sync}"
    exit 2
esac

exit $?
```

5.4 Add Makefile for SDK Building System

```
$ cp package/sample_uci/Makefile package/helloworld/
$ vi package/helloworld/Makefile
```

Edit Makefile based on the example.

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

PKG_NAME:=helloworld
PKG_VERSION:=1.0
PKG_DESC:=helloworld example
```

```

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

PKG_UNPACK:=cp -rf src/* $(PKG_BUILD_DIR)

PKG_BUILD_PARALLEL:=1
PKG_BUILD_DEPENDS:=

include $(INCLUDE_DIR)/package.mk

define Package/Install
    $(INSTALL_DIR) $(1)/usr/bin/
    $(INSTALL_BIN) $(PKG_BUILD_DIR)/helloworld $(1)/usr/bin/
    $(INSTALL_DIR) $(1)/etc/sdk/
    $(INSTALL_BIN) files/sdk.sh $(1)/etc/sdk/$(PKG_NAME)
endef

$(eval $(call BuildPackage))
    
```

5.5 Build Helloworld Package

```
$ make package/helloworld/install
```

Then you can find the rpkg file in bin/target-<platform>/packages.

5.6 Install APP

App Center

For more information about App, please refer to <http://www.robustel.com/products/app-center/>.

App Install

File r2000-hell...rld-1.0.rpk

App Usage 54.4MB Free/57.5MB Total

Installed Apps

Index	Name	Version	Status	Description
-------	------	---------	--------	-------------

5.7 Check APP Running Status

```

services
System
  Debug
  Update
  App Center
  Tools
  Profile
Jan 1 00:00:02 router user.notice init[1]: dnsmasq started for interface lan0
Jan 1 00:00:02 router user.notice link_manager[742]: link manager started
Jan 1 00:00:02 router user.debug link_manager[742]: racy action connect from link_manager
Jan 1 00:00:02 router user.debug link_manager[742]: target link WWAN1, state Disconnected
Jan 1 00:00:02 router user.info link_manager[742]: WWAN1 start connect
Jan 1 00:00:02 router user.notice link_manager[742]: modem is not ready now, wait...
Jan 1 00:00:02 router user.info link_manager[742]: WWAN1 is not ready, waiting for initialization
Jan 1 00:00:03 router user.notice init[1]: firewall started
Jan 1 00:00:03 router user.notice init[1]: telnet started
Jan 1 00:00:03 router user.notice modemend[778]: modem service started
Jan 1 00:00:03 router user.info modemend[778]: modem initialization start, using SIM1
Jan 1 00:00:03 router user.debug syslog: hello world
Jan 1 00:00:10 router user.debug modemend[778]: ATT
    
```

5.8 APP Package Naming Rules

App should be named according to the following naming rules:

- 1、 App package name should not include midline character “-”.
- 2、 The length of package name should be less than 18 characters.

Chapter 6 Integrate into RobustOS Web Manager

The operators need a user interface for a generic application to change configuration and show running status. The SDK provides an easy way to integrate your application's configuration and status into the router's Web Manager. You do not need to know any web programming languages. Add a simple file in a pre-defined format, and the web page will be generated automatically.

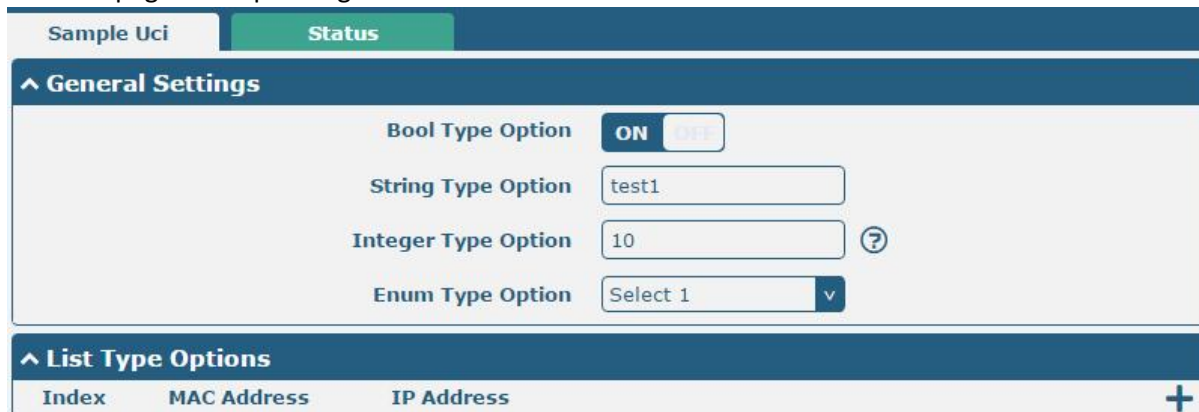
If the application you will develop does not have any configuration data, or you are planning to handle configurations in another way, you can skip this chapter.

If you would like to integrate your APP into the Web Manager of the router, just like the APP released officially, you should follow the UCI subsystem rules. The abbreviation UCI stands for Unified Configuration Interface and is intended to centralize the router's configuration. The UCI subsystem uses XML to describe data structure of configuration or items of status. Here we use an example to explain the UCI subsystem, and you can find the example in the package/sample_uci directory.

6.1 Configuration Definition

```
<sample_uci desc="configuration of SDK Sample Uci" group="Services" group_weight="40" menu="SDK Sample Uci"
  menu_weight="901" oid="1001">
  <option_bool type="bool" desc="Bool Type Option" default="false" tab="Sample Uci" tab_weight="0" area="General Settings"
    oid="1" />
  <option_string default="test" desc="String Type Option" oid="2" />
  <option_int type="int" desc="Integer Type Option" range="1..100" default="10" row_help="This is an integer type config
    option." oid="3" />
  <option_enum type="enum" desc="Enum Type Option" range="select1 select2" default="select1" range_desc="Select_1
    Select_2" oid="4" />
  <option_list desc="List Example" area="List Type Options" max_entry_num="20" oid="5" >
    <id type="int" range="1..20" desc="Index" col_width="80" area="Popup Window Title" oid="2" />
    <mac type="MAC_ADDR" unique="true" desc="MAC Address" col_width="140" oid="3" />
    <ip type="IP_ADDR" unique="true" desc="IP Address" col_width="140" oid="4" />
  </option_list>
</sample_uci>
```

The web page corresponding to the above XML is shown below:



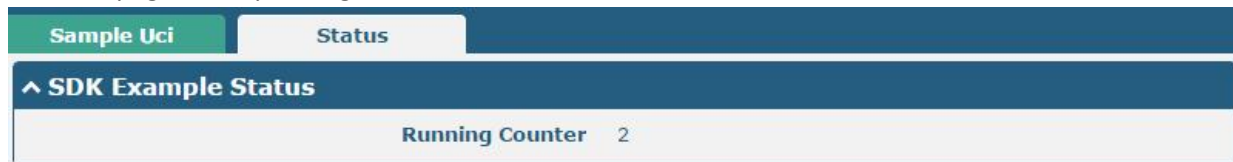
And you can also get it in CLI:

```
# show sample_uci all
option_bool = true
option_string = test
option_int = 10
option_enum = select1
```

6.2 Status Definition

```
<sample_uci cli_help="status of SDK Sample Uci" group="Services" group_weight="40" menu="SDK Sample Uci"
  menu_weight="901" oid="1000" >
  <counter desc="Running Counter" tab="Status" tab_weight="20" area="SDK Example Status" oid="1" />
</sample_uci>
```

The web page corresponding to the above XML is shown below:



And you can also get it in CLI:

```
# status sample_uci
counter = 2
```

6.3 Webpage Layout

The screenshot shows the Robustel SDK web interface. At the top, there is a navigation bar with the Robustel logo, 'Save & Apply', 'Reboot', and 'Logout' buttons. A yellow warning banner states: 'It is strongly recommended to change the default password.' Below this, the main content area is titled 'Example1' and contains two sections: 'General Settings' and 'List Type Options'. The 'General Settings' section includes:

- Bool Type Option: ON/OFF toggle
- String Type Option: text input field
- Integer Type Option: 10 input field with a help icon
- Enum Type Option: Select 1 dropdown menu

 The 'List Type Options' section contains a table with the following data:

Index	Member 1	Member 2	
1	value of member 1	one	list
2	value of member 2	two	
3	value of member 3	three	

 On the left side, there is a navigation menu with categories: Status, Interface, Network, VPN, Services, and System. The 'Services' category is expanded, showing options like Syslog, Event, NTP, SMS, Email, SSH, SNMP, Web Server, and Advanced. The 'SDK Example1' option is highlighted. The 'System' category is also visible with a 'group' label. At the bottom right, there are 'Submit' and 'Cancel' buttons. The footer contains the copyright notice: 'Copyright © 2015 Robustel Technologies. All rights reserved.'

6.4 UCI Attributes

- **group** - the group name in the navigation bar, please set it to 'Services'.
- **group_weight** - specifies the location of the group, please set it to 40.
- **menu** - the menu name in the navigation bar.
- **menu_weight** - specifies the location of the menu, the default is 1000.
- **tab** – the name of the tab on the page, you can define more than one tab if you need, e.g. one tab for configuration, one tab for status.
- **area** - defines an area used for grouping the configuration options.
- **area_help** - optional, you can add some helpful info for this area.
- **row_help** - optional, one config option takes up one row, you can add some helpful info for this option.
- **desc** - description of config option.

- **type** - config option, including string, int, enum, and bool. The default type is <string>. The string type and int type element will be displayed as a one-line text input field in the browser. The enum type element will be displayed as a drop-down list. The bool type element will be displayed as a radio button. Some internal types are also used for special string verification, including IP_ADDR, NETMASK, MAC_ADDR. You can use regular expressions for any other string verification.
- **range** - specifies the range of the config value. It can be the length range of string, numerical range of an integer, or the available options for an enum type.
- **range_desc** - optional, you can specify the description of enumerated options for the enum type.
- **default** - specifies the default value of this config option.
- **max_entry_num** - specifies the list's maximum rows, also used to indicate a list type configuration.
- **pattern** - specifies the regular expression used for verification.
- **unique** - the setting value of this option should be unique in all list entries.
- **col_width** - optional, all the config options in a list will be displayed in a pop-up window. You can add a col_width attribute to show this config option in the overview table. The col_width attribute also specifies the width of this row in a table. The width range is 10-200.
- **oid** - optional, specifies the OID used for SNMP management. The top-level OID should be greater than 1000. MIB file for this application will be generated automatically if the OID attribute is specified.

6.5 UCI Tag

It's vital to understand UCI tag before coding. Every configuration option can be referenced by a unique tag, mainly like a C-style variable reference. For example, use `syslog.level` to reference the debug level or use `ipsec.tunnel[1].mode` to reference the mode of the first IPsec tunnel. The list ID starts from 1, not 0. Please refer to the next chapter for more detailed information about UCI API. The tagged method for status is the same as the configuration.

There is an easy way to find the corresponding tag of a specific configuration option or status item. You can check the configuration tag with the `show` command in the CLI system and the `status` command for the tag of status.

```
// IP address of lan0, the UCI tag is lan.network[1].ip
# show lan all
network {
    id = 1
    interface = lan0
    ip = 192.168.0.1
    netmask = 255.255.255.0
    mtu = 1500
    dhcp {
        enable = true
        mode = server
        relay_server = ""
        pool_start = 192.168.0.2
        pool_end = 192.168.0.100
        netmask = 255.255.255.0
        gateway = ""
        primary_dns = ""
    }
}
```

```
    secondary_dns = ""
    wins_server = ""
    lease_time = 120
    static_lease = ""
    expert_options = ""
    debug_enable = false
}
}

// get MAC address from status, the UCI tag is lan.mac
// get RX bytes stats of lan0, the UCI tag is lan.interface_status[1].rx_bytes
# status lan
interface_status {
    id = 1
    interface = lan0
    mtu = 1500
    mac = 34:fa:40:10:26:0a
    ip = 192.168.0.1/255.255.255.0
    rx_bytes = 8354654
    rx_packets = 25224
    tx_bytes = 209611
    tx_packets = 247
}
mac = 34:fa:40:10:26:0a
ip = 192.168.0.1/255.255.255.0
connected_device {
    id = 1
    ip = 192.168.0.10
    mac = 00:13:3B:0C:22:67
    interface = lan0
    inactive_time = 176191s
}
#
```

6.6 How to Get Configuration Settings

```
int i;
int max_entry_num;
char *p;

# get string type config setting
p = uci_get("sample_uci.option_string");
# get integer type config setting
i = uci_get_int("sample_uci.option_int");
# check bool type config option
```

```

if (uci_check_bool("sample_uci.option_bool"))
{
    // do something
}
# check bool type config option
if (uci_match("sample_uci.option_enum", "select1"))
{
    // do something
}
# list traversal
max_entry_num = uci_list_get_max_entry_num("sample_uci.option_list");
for (i=1; i<= max_entry_num; i++)
{
    p = uci_list_get("sample_uci.option_list", i, "ip");
    printf("IP Address = %s\n", p);
}

```

6.7 How to Change Configuration Settings

Find tag of config option in the CLI first.

```

# show link_manager all
primary_link = wwan1
backup_link = none
backup_mode = cold_backup
revert_interval = 0
emergency_reboot = false
Link {
.....

```

```

uci_init();
# change current link to SIM2
uci_set("link_manager.primary_link", "wwan2");
# just apply the changes, do not save
system("uci apply");

```

6.8 How to Find UCI Tags

Using the administrator account, log in to the router by telnet, and it will enter CLI mode. CLI shell will automatically prompt the next valid menu in CLI mode by typing the TAB key. For UCI tag, type the UCI command "set", then TAB key, select the menu, and TAB again till it reach the final attribute. For example, try to set primary link to wwan2 in the link manager.

```
#
# set "set" and TAB
access_point    cellular    ddns            dido            email
ethernet        event      firewall        gre            ip_passthrough
ipsec           lan        link_manager    ntp            openvpn
reboot          robustlink3 route            serial_port    sms
ssh             syslog     system          usb            user_management
web_server      wifi
# set link_manager "set link_manager" and TAB
primary_link    backup_link    backup_mode     revert_interval
emergency_reboot link
# set link_manager primary_link
wwan1 wwan2 wan    wlan
# set link_manager primary_link wwan2 final attribute wwan2 then Enter
OK
#
```

Currently, UCI tag name for setting primary link is "link_manager.primary_link" and the value is "wwan2", see example code below.

```
uci_init();
# change current link to SIM2
uci_set("link_manager.primary_link", "wwan2");
# just apply the changes, do not save
system("uci apply");
```

6.9 USI Tag

USI tag is for the status information, similar to UCI tag, also defined by XML file, and the attribute is the same. Please refer to Chapter 6.5 UCI tag.

6.10 How to Update Status

```
int counter = 10;
usi_update("sample_uci.counter", "10");
usi_printf("sample_uci.counter", "%d", counter);
```

6.11 How to Get Status

Find tag of status in the CLI first.

```
# status system
hardware_version = 1.0
firmware_version = "2.0.7 (Rev 516)"
kernel_version = 3.10.49
device_model = "R2000"
serial_number = 01570316110002
uptime = "0 days, 00:01:07"
system_time = "Fri Jan 1 00:00:52 2016 (NTP not updated)"
# status cellular
```

```

total_sims = 1
status {
    id = 1
    modem_status = Ready
    current_sim = SIM1
    registration = "Registered to home network"
    cell_id = 517A,0000E602
    operator = "China Unicom"
    network_type = GSM/GPRS
    csq = "19 (-75dBm)"
    modem_model = MC7354
    imei = 359225050341551
    firmware_version = "SWI9X15C_05.05.58.00 r27038 carmd-fwbuild1 2015/03/04 21:30:23"
    imsi = 460065049045512
    iccid = 89860616090020638522
    phone_num = ""
    lai = 46001
    apn = internet
    username = ""
}
Link_uptime = "0 days, 00:00:46"

```

```

char *imei;
char *model;
int sim;

model = usi_get("system.device_model");
if (NULL != model)
{
    printf("Device Model = %s\n", model);
    free(model);
}
imei = usi_get("cellular.status[1].imei");
if (NULL != imei)
{
    printf("IMEI = %s\n", imei);
    free(imei);
}
sim = renv_get_int("cellular_current_sim");
imei = usi_list_get("cellular.status", sim, "imei");
if (NULL != imei)
{
    printf("IMEI = %s\n", imei);
    free(imei);
}

```

6.12 How to Find USI Tags

Similar to chapter “6.8 How to find UCI tags”, For USI tag, type the USI command “status”, then TAB key, select the menu, and TAB again till it reaches the final attribute. For example, try to get device_model shown on the web page.

```
#
# status status and TAB
access_point    cellular        ddns            dido            email
ethernet        event            firewall        gps             gre
ip_passthrough ipsec           lan             link_manager   ntp
openvpn         reboot          robustlink3    route          serial_port
sms             ssh             syslog         system         usb
user_management web_server     wifi

# status system status system and Enter
hardware_version = 1.0
ext_hardware_ver = 1.3
firmware_version = 2.0.0
firmware_version_full = "2.0.0 (Rev 2219)"
kernel_version = 3.18.92
device_model = R2100
serial_number = ""
uptime = "0 days, 00:05:40"
system_time = "Sun Jan 1 00:05:17 2017 (NTP not updated)"
ram_usage = "385M Free/448M Total"
#
#
```

It is the same as shown on web page.



Get the device_model in code:

```
char *imei;
char *model;
int sim;

model = usi_get("system.device_model");
if (NULL != model)
{
    printf("Device Model = %s\n", model);
    free(model);
}
```

Chapter 7 SDK C-Libraries

The SDK contains a private library named librouter, which is used for router functions. The librouter APIs give you programmatic access to UCI system, router environment variables, router IPC system, etc.

Some pre-compiled open libraries are also available in the SDK, such as OpenSSL, libcurl and libevent.

The header and libraries are located in the staging_dir/target-<platform>/usr/ directory. You can check the headers for detailed API descriptions or examples for instructions.

7.1 librouter data type

```
typedef signed char      int8_t;
typedef short int       int16_t;
typedef int             int32_t;
typedef long long int   int64_t;
typedef unsigned char   uint8_t;
typedef unsigned short int uint16_t;
typedef unsigned int    uint32_t;
typedef unsigned long long int uint64_t;
typedef int             intptr_t;

typedef int             status_t;
typedef unsigned int   bool_t;

#define TRUE           1
#define FALSE          0

#define ERROR          -1
#define OK              0

#ifndef NULL
#define NULL ((void *)0)
#endif
```

7.2 librouter Log API

Name:

```
log_open(n)
```

Description:

log_open() is similar to openlog(), it opens a connection to the system logger for a program. The string pointed to by n is prepended to every message, and is typically set to the program name. The use of log_open()

is optional; it will automatically be called by `syslog()` if necessary, in which case identity will default to `NULL`.

INPUT:

n

Output:

N/A

RETURN VALUE:

N/A

Name:

`log_close()`

Description:

closes the descriptor being used to write to the system logger. The use of `log_close()` is optional.

INPUT:

N/A

OUTPUT:

N/A

RETURN VALUE:

N/A

Name:

`log_debug(fmt, ...)`

Description:

Debug message log to syslog.

INPUT:

Message to log.

OUTPUT:

Output to syslog

RETURN VALUE:

N/A

Name:

`log_info(fmt, ...)`

Description:

Info message log to syslog.

INPUT:

Message to log.

OUTPUT:

Output to syslog

RETURN VALUE:

N/A

Name:

`log_notice(fmt, ...)`

Description:

Notice message log to syslog.

INPUT:
 Message to log.

OUTPUT:
 Output to syslog

RETURN VALUE:
 N/A

Name:
 log_warn(fmt, ...)

Description:
 Warning message log to syslog.

INPUT:
 Message to log.

OUTPUT:
 Output to syslog

RETURN VALUE:
 N/A

Name:
 log_error(fmt, ...)

Description:
 Error message log to syslog.

INPUT:
 Message to log.

OUTPUT:
 Output to syslog

RETURN VALUE:
 N/A

7.3 librouter UCI API

Name:
 status_t uci_init(void)

Description:
 If you want to modify the configuration file, you should call this init function firstly. There is no need to call uci_init if you just want to read configuration.

INPUT:
 N/A

OUTPUT:
 N/A

RETURN VALUE:
 OK means successful, or ERROR if an error occurred.

Name:

```
status_t uci_close(void)
```

Description:

User had better call this before exit if you had called uci_init.

INPUT:

N/A

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
const char *uci_get(const char *tag)
```

Description:

Get configuration by specified tag, it does not need to calling free() after get.

INPUT:

tag - tag with UCI format, eg: lan.ip means ip of lan.

OUTPUT:

N/A

RETURN VALUE:

The pointer of value, or NULL if an error occurred.

Name:

```
int32_t uci_get_int(const char *tag)
```

Description:

Get integer type configuration data.

INPUT:

tag - tag with UCI format

OUTPUT:

N/A

RETURN VALUE:

The integer value, or 0 if an error occurred.

Name:

```
bool_t uci_match(const char *tag, const char *value)
```

Description:

Check configuration data is matched or not.

INPUT:

tag - tag with UCI format

value - the value to compare

OUTPUT:

N/A

RETURN VALUE:

TRUE if matched, otherwise return FALSE.

Name:

```
bool_t uci_check_bool(const char *tag)
```

Description:

Check bool type configuration data is TRUE or FALSE.

INPUT:

tag - tag with UCI format

OUTPUT:

N/A

RETURN VALUE:

TRUE if the value is "true", otherwise return FALSE.

Name:

```
status_t uci_set(const char *tag, const char *value)
```

Description:

Set specified configuration item to specified value.

INPUT:

tag - tag with UCI format

value - string type value

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
int32_t uci_list_get_max_entry_num(const char *tag)
```

Description:

Get the maximum entry number of list, use to traversal the list.

INPUT:

tag - tag with UCI format

value - string type value

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
bool_t uci_list_entry_empty(const char *tag, uint32_t id)
```

Description:

Check list entry is empty or not.

INPUT:

tag - tag with UCI format

id - list ID

OUTPUT:

N/A

RETURN VALUE:

TRUE if the list entry is empty, otherwise return FALSE.

Name:

```
status_t uci_list_add(const char *tag, uint32_t id)
```

Description:

Setup the new list, you must to add a new entry to the list before.

INPUT:

```
tag - tag with UCI format
id - list ID
```

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
status_t uci_list_del(const char *tag, uint32_t id)
```

Description:

Delete a list entry with specified ID.

INPUT:

```
tag - tag with UCI format
id - list ID to delete
```

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
const char *uci_list_get(const char *list_tag, uint32_t id, const char *member_name)
```

Description:

Get the value of specified list member.

INPUT:

```
list_tag - tag with UCI format
id - list ID
member_name - member name of list
```

OUTPUT:

N/A

RETURN VALUE:

The pointer of value, or NULL if an error occurred.

Name:

```
status_t uci_list_set(const char *list_tag, uint32_t id, const char *member_name, const char *value)
```

Description:

Set the value of specified list member.

INPUT:

```
list_tag - tag with UCI format
id - list ID
member_name - member name of list
```

value - string type value

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

bool_t uci_list_match(const char *list_tag, uint32_t id, const char *member_name, const char *value)

Description:

Set the value of specified list member.

INPUT:

tag - tag with UCI format

id - list ID

member_name - member name of list

value - value to compare

OUTPUT:

N/A

RETURN VALUE:

TRUE if matched, otherwise return FALSE.

Name:

char *usi_get(const char *tag)

Description:

Get status by tag, it's up to the caller to free the memory.

INPUT:

tag - tag with USI format, be the same as UCI format

OUTPUT:

N/A

RETURN VALUE:

The pointer of value, or NULL if an error occurred.

Name:

bool_t usi_match(const char *tag, const char *value)

Description:

check status data is matched or not.

INPUT:

tag - tag with USI format

value - the value to compare

OUTPUT:

N/A

RETURN VALUE:

TRUE if matched, otherwise return FALSE.

Name:

status_t usi_update(const char *tag, const char *status)

Description:

Update specified status item to specified value, if tag is not define it will be created.

INPUT:

tag - tag with USI format
status- string type value

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
status_t usi_printf(const char *tag, const char *fmt, ...)
```

Description:

printf-style function to update status.

INPUT:

tag - tag with USI format
fmt - printf-style format

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
status_t usi_clear(const char *tag)
```

Description:

clear a status file, or clear all items of specified list, or clear an item.

INPUT:

tag - tag with USI format

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
status_t usi_list_set(const char *list_tag, uint32_t id, const char *member_name, const char *value)
```

Description:

update specified list member to specified value.

INPUT:

list_tag - tag with UCI format
id - list ID
member_name - list member
value - string type value

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

`status_t usi_list_printf(const char *list_tag, uint32_t id, const char *member_name, const char *fmt, ...)`

Description:

Update specified list member to specified value.

INPUT:

`list_tag` - tag with UCI format

`id` - list ID

`member_name` - list member

`fmt` - printf-style format

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

`char *usi_list_get(const char *list_tag, uint32_t id, const char *member_name)`

Description:

Get status of specified list member, it's up to the caller to free the memory.

INPUT:

`list_tag` - tag with UCI format

`id` - list ID

`member_name` - list member

OUTPUT:

N/A

RETURN VALUE:

The pointer of value, or NULL if an error occurred.

Name:

`status_t usi_list_delete(const char *list_tag, uint32_t id)`

Description:

Delete a list entry with specified ID.

INPUT:

`list_tag` - tag with UCI format

`id` - list ID to delete

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

`bool_t usi_list_match(const char *list_tag, uint32_t id, const char *member_name, const char *value)`

Description:

Check the status value of specified list member is matched or not.

INPUT:

list_tag - tag with UCI format
 id - list ID
 member_name - member name of list
 value - value to compare

OUTPUT:

N/A

RETURN VALUE:

TRUE if matched, otherwise return FALSE.

Name:

int32_t usi_list_get_max_entry_num(const char *tag)

Description:

Get the maximum entry number of list, use to traversal the list.

INPUT:

list_tag - tag with UCI format

OUTPUT:

N/A

RETURN VALUE:

maximum entry number of list, 0 if an error occurred.

Name:

bool_t usi_list_entry_empty(const char *tag, uint32_t id)

Description:

Check list entry is empty or not.

INPUT:

list_tag - tag with UCI format
 id - list ID

OUTPUT:

N/A

RETURN VALUE:

TRUE if the list entry is empty, otherwise return FALSE

7.4 librouter Environment Variable API

Name:

status_t renv_set(const char *name, const char *value)

Description:

Add or change a name/value pair to router environment variable

INPUT:

name - name of environment variable
 value - value of environment variable

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
status_t renv_printf(const char *name, const char *fmt, ...)
```

Description:

printf-style function to set environment variable

INPUT:

name - name of environmentvariable

fmt - printf-style format

OUTPUT:

N/A

RETURN VALUE:

OK means successful, or ERROR if an error occurred.

Name:

```
const char *renv_get(const char *name)
```

Description:

Get environment variable by name, it does not need to calling free() after get.

INPUT:

name - name of environment variable

OUTPUT:

N/A

RETURN VALUE:

The pointer of value, or NULL if an error occurred.

Name:

```
int32_t renv_get_int(const char *name)
```

Description:

Get integer type environment variable by name.

INPUT:

name - name of environment variable

OUTPUT:

N/A

RETURN VALUE:

The integer value, or 0 if an error occurred.

Name:

```
uint32_t renv_get_uint(const char *name)
```

Description:

Get unsigned integer type environment variable by name.

INPUT:

name - name of environment variable

OUTPUT:

N/A

RETURN VALUE:

The integer value, or 0 if an error occurred.

Name:

`status_t renv_del(const char *name)`

Description:

Delete environment variable by name.

INPUT:

name - name of environment variable

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

`bool_t renv_match(const char *name, const char *value)`

Description:

Check environment variable is matched or not.

INPUT:

name - name of environment variable

value- the value to compare

OUTPUT:

N/A

RETURN VALUE:

TRUE if matched, otherwise return FALSE.

7.5 librouter IPC API

Name:

`int rmsg_init(const char *sock_name)`

Description:

For IPC you can open a UNIX Domain Socket for message communication. You should call `rmsg_init` before `rmsg_send`.

INPUT:

sock_name - socket name, without path

OUTPUT:

N/A

RETURN VALUE:

Socket descriptor if success, or -1 if an error occurred.

Name:

`status_t rmsg_close(void)`

Description:

You had better call `rmsg_close` if you do not want to send any more messages.

INPUT:

N/A

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

status_t rmsg_send(dst, msgs, ...)

Description:

Send message parts to destination.

INPUT:

dst - destination socket name

msgs - key-value style message pairs

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

status_t rmsg_rcv(char **msg, char **from)

Description:

Receive message.

INPUT:

msg - pointer to the message buffer

from - pointer to the sender

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

char *rmsg_get(const char *key)

Description:

Get the corresponding value of specified key, it's up to the caller to free the memory.

INPUT:

key - key of message item

OUTPUT:

N/A

RETURN VALUE:

Pointer to the value if success, NULL if an error occurred.

7.6 librouter Event API

Name:

```
status_t event_subscribe(const char *event)
```

Description:

Send a message to event service to subscribe an event, `rmsg_init` should be called firstly.

INPUT:

event - event name

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

```
status_t event_unsubscribe_all(void)
```

Description:

unsubscribe all events, user had better call this before exit if you had called `event_subscribe`.

INPUT:

N/A

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

```
status_t event_report(const char *event, const char *data)
```

Description:

send a message to event service to report an event, `rmsg_init` should be called firstly.

INPUT:

event - event name

data - extra data of the event

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

```
status_t event_report_with_formatted_data(const char *event, const char *fmt, ...)
```

Description:

printf-style's `event_report`

INPUT:

event - event name

fmt - printf-style format

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

```
status_t event_report_full(const char *from, const char *event, const char *data)
```

Description:

printf-style's event_report

INPUT:

from - same as socket name in rmsg_init

event - event name

data - extra data of the event

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Name:

```
status_t event_report_full_with_formatted_data(const char *from, const char *event, const char *fmt, ...)
```

Description:

printf-style's event_report_full

INPUT:

from - same as socket name in rmsg_init

event - event name

fmt - printf-style format

OUTPUT:

N/A

RETURN VALUE:

OK if success, or ERROR if an error occurred.

Chapter 8 Pre-installed Tools

8.1 Open Source Libraries

- OpenSSL
- libevent
- libxml2
- curl
- ncurses

8.2 Open Source Commands

- curl
- OpenSSL
- mutt
- TFTP

8.3 RobustOS Private Commands

8.3.1 uci

```
$ uci -h
Usage: uci [<options>] <command> [<arguments>]

Commands:
  get      <tag> (get config by tag, e.g. get lan.ip)
  set      <tag> <value> (update config, e.g. set openvpn.tunnel[1].enable true)
  add      <tag> <id> (add a new list entry, e.g. add openvpn.tunnel 1)
  del      <tag> <id> (delete a list entry)
  load     (load config from xml)
  default  (generate default config)
  save     (save config changes)
  apply    (apply config changes)
  commit   (save and apply config changes)
  export   [<xmlfile>] (export config to xml, default to /tmp/config.xml)
  import   <xmlfile> (import config from xml)
  check    <category> (check config has changed or not)
  refresh  [<category>] (reload config from file to RAM)
  print    (show all uciconfig in the RAM)

Options:
  -c      when importing, change other config to default value
```

```
-d    when exporting, output detailed information
-s    when exporting, ignore some disabled feature to get a simpler xml file
-h    display this help and exit
```

8.3.2 usi

```
$ usi
Usage: usi<command> [<arguments>]

Commands:
  get <tag> (get status by tag, e.g. get system.firmware_version)
  set <tag> <value> (update status, e.g. set cellular.imei 123456789012345)
  clear <tag> (clear status, e.g. clear openvpn.tunnel_status)
  sync <category> (sync status to unified format)
```

8.3.3 renv

```
$ renv -h
Usage: renv [<command>] [<arguments>]

Print the current router environment if no command specified.

Commands:
  get<name>
  set<name><value>
  del<name>
```

8.3.4 sms

```
$ sms -h
sms: unknown option -h

usage: sms [options...] <phone_number> <sms_content>

Options:
  -s use the second modem
```

Chapter 9 Programming Tutorials

9.1 How to Debug the Application

9.1.1 Use Syslog

The RobustOS supports debugging via syslog. You can check syslog in the Web Manager, which located in System->Debug.

9.1.2 Use Shell with Root Privileges

A more convenient way is debugging with shell. The shell is not open unless you import a license file by default. You need to sign the NDA agreement to get the license file. Please contact your sales manager.

After importing the license on the update page, you can set the root password for shell login on the user management page. You can check syslog at `/var/log/messages` with root privileges.

9.1.3 Use tftp

For continuous debugging, you can download the program directly to the router through tftp. The target program is usually located in the `build_dir` directory.

```
$ tftp -g 192.168.0.10 -r example -l /usr/bin/example
$ chmod 755 /usr/bin/example
$ example
```

9.1.4 Use gdbserver

The gdbserver is also available for debugging and is included in the RobustOS SDK. You can build gdbserver as an APP and then install it to the router for remote debugging.

- **Install gdbserver**

To use the GDB debugger, you need to run the gdbserver program in the device. In the ROS5 source directory, use the following command to generate the gdbserver installation package

```
$ make package/gdbserver/install
```

- **Set gdbserver parameters**



Enable gdbserver, set the listening port for the gdbserver runtime, and specify the app name (executable file name) to debug.

Enable the Attach option to attach to a running process, otherwise a new process will be started for debugging.

Run the ps command to view the running status of gdbserver.

```

1269 root    0:00 {e2c_app_monitor} /bin/sh /usr/bin/e2c_app_monitor e2c_s_modb
1274 root    0:00 gdbserver_demo
1304 root    0:00 e2c_n_mqtt
1305 root    0:03 e2c_broker
1468 root    0:00 sshd: root@pts/0
1480 root    0:00 -sh
2681 root    0:00 /usr/bin/gdbserver 0.0.0.0:8888 /usr/bin/e2c_s_modbus
2684 root    0:00 /usr/bin/e2c_s_modbus
2981 root    0:00 [kworker/u8:0]
4367 root    0:00 [kworker/0:2]
4842 root    0:00 [kworker/u8:2]
5890 root    0:00 [kworker/0:0]
5927 root    0:00 modemd
6121 root    0:00 sleep 10
6124 root    0:00 sleep 10
    
```

● **Compile an executable program with debugging information**

GDB debugging requires an executable program with debugging information, and the -g parameter needs to be added when compiling the app. To compile ROS 5 app, use the RGDB_DEBUG=1 parameter.

```
make package/e2c_s_modbus/install RGDB_DEBUG=1
```

The resulting build_dir/target-r2110/edge2cloud/e2c_s_modbus/e2c_s_modbus is the program we will use.

You only need to run the gdb program on the PC to load the program with debugging information. The program installed on the device does not need to carry debugging information.

● **gdb file path**

The program running on the remote debugging router needs to use the gdb program provided by the cross-compilation toolchain. The file path is:

```
staging_dir/target-$PLATFORM/$GNU_TARGET_NAME/bin/$GNU_TARGET_NAME-gdb
```

For example:

```

staging_dir/target-r3000/arm-sam9x25-linux-gnueabi/bin/arm-sam9x25-linux-gnueabi-gdb
staging_dir/target-r2110/mipsel-74kc-linux-gnu/bin/mipsel-74kc-linux-gnu-gdb
staging_dir/target-r1510/mips-24k-linux-gnu/bin/mips-24k-linux-gnu-gdb
    
```

● **gdb debugging steps**

The following uses the R2110 device to debug the e2c_s_modbus program as an example to describe how to debug gdb remotely

Step1:Run the gdb program

```
./staging_dir/target-r2110/mipsel-74kc-linux-gnu/bin/mipsel-74kc-linux-gnu-gdb
build_dir/target-r2110/edge2cloud/e2c_s_modbus/e2c_s_modbus
```

```
→ ros git:(master) ./staging_dir/target-r2110/mipsel-74kc-linux-gnu/bin/mipsel-74kc-linux-gnu-gdb build_dir
/target-r2110/edge2cloud/e2c_s_modbus/e2c_s_modbus
GNU gdb (crosstool-NG 1.21.0) 7.12
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "--host=x86_64-build_unknown-linux-gnu --target=mipsel-74kc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from build_dir/target-r2110/edge2cloud/e2c_s_modbus/e2c_s_modbus...done.
(gdb)
```

Step2:Connect to remote gdbserver

```
(gdb) target remote 192.168.0.1:8888
```

```
(gdb) target remote 192.168.0.1:8888
Remote debugging using 192.168.0.1:8888
Reading symbols from /home/lvliang/git/ros/staging_dir/target-r2110/mipsel-74kc-linux-gnu/mipsel-74kc-linux-
gnu/sysroot/lib/ld.so.1...(no debugging symbols found)...done.
0x77fcd040 in __start ()
    from /home/lvliang/git/ros/staging_dir/target-r2110/mipsel-74kc-linux-gnu/mipsel-74kc-linux-gnu/sysroot/l
ib/ld.so.1
(gdb)
```

Step3:Set breakpoints

Suppose we need to set a breakpoint at line 873 of the modbus_plc.c file in the e2c_s_modbus source code to observe the value of recv_bytes

```
(gdb) break modbus_plc.c:873
```

```
(gdb) break modbus_plc.c:873
Breakpoint 1 at 0x403d48: file src/modbus_plc.c, line 873.
```

Step4:Run the program

Type continue/c to execute the program, which will stop when it reaches a breakpoint.

```
(gdb) c
Continuing.
warning: Could not load shared library symbols for 7 libraries, e.g. /usr/lib/librouter.so.
Use the "info sharedlibrary" command to see the complete listing.
Do you need "set solib-search-path" or "set sysroot"?
[New Thread 7854.13870]

Thread 1 "e2c_s_modbus" hit Breakpoint 1, mb_tcp_read (channel=channel@entry=0x4241e8)
    at src/modbus_plc.c:873
873         if (recv_bytes == 0) {
(gdb)
```

Step5:Check the variable

Enter print recv_bytes to view the value of the recv_bytes variable. You can also view the values of other variables.

```
(gdb) print recv_bytes
$1 = 2
(gdb) print channel->register_addr
$2 = 0
(gdb)
```

9.1.5 Debugging under x86/x64

It is recommended that most applications be developed and debugged in the x86 environment before being ported to the router, especially applications that are not related to the hardware interface. After the debugging is completed, it only needs to be cross-compiled and packaged to run flawlessly in the router.

9.2 How to Get System Status

Please refer to [Chapter 6](#).

9.3 How to Read/Write Data via Serial Port

Serial port programming in RobustOS is the same as in standard Linux systems. You should only notice the device file of serial port, which has been listed in Chapter 2.

9.4 How to Get DI Status

To get DI status, the application needs to subscribe to the DI ON/OFF event. An example is shown below.

```
#include "librouter.h"
#include "rlib_shell_utils.h"
#include "rlib_define.h"
#include "rlib_shared.h"
#include "rlib_string.h"
#include "event.h"

void di_event_msg_proc(int sock, short ev, void *arg)
{
    char *msg;
    char *from;
    char *cmd = NULL;
    char *event = NULL;

    if (OK != rmsg_rcv(&msg, &from))
    {
        log_err_location();
        return;
    }
}
```

```
}
log_debug("recv msg %s from %s", msg, from);

cmd = rmsg_get("cmd");
if (!cmd)
{
    log_err_location();
    goto END;
}
if (string_matched(cmd, "report"))
{
    event = rmsg_get("event");
    if (!event)
    {
        log_err_location();
        goto END;
    }
    if (string_matched(event, "di1_on"))
    {
        log_debug("DI1 on");
    }
    else if(string_matched(event, "di1_off"))
    {
        log_debug("DI1 off");
    }
}

END:
safe_free(cmd);
safe_free(event);

}

int main(int argc, char *argv[])
{

    int sd= -1;
    struct event event;

    daemon(0, 0);

    if(daemon_check_running("/var/run/sample_dido.pid"))
    {
        log_warn("sdk sample_dido already running!");
        return 0;
    }
}
```

```

if(!luci_check_bool("devcfg.dido_support"))
{
    log_error("Don't Support DIDO, sample_dido exit!");
    return -1;
}

log_notice("sdk sample_dido service started");

sd = rmsg_init(argv[0]);
event_init();

event_set(&event, sd, EV_READ|EV_PERSIST, di_event_msg_proc, NULL);
event_add(&event, NULL);

event_subscribe("di1_on");
event_subscribe("di1_off");
event_dispatch();

return 0;
}

```

9.5 How to Trigger DO Changes

RobustOS uses the Linux led subsystem for DO processing.

```

$ echo 0 > /sys/class/leds/do1/brightness
$ echo 1 > /sys/class/leds/do1/brightness

```

9.6 How to Get AI Status

There are two ways to get AI status.

- Through AI device file

AI value can be read from the AI device file `/dev/AI1`. It is used to get AI precision values.

- Subscribe AI events

If applications do not care about the precision values but whether they reach the voltage/current threshold, they can subscribe to the AI events listed below. Please take reference to how to get DI status.

9.7 How to Subscribe Event

9.7.1 Supported Events

- `system_startup` - system has been started
- `system_reboot` - system is going to reboot
- `system_time_updated` - system time has been updated

- system_login_success - someone has been logged in to the system successfully
- system_login_fail - someone failed to login the system
- config_changed - configuration has been changed
- cellular_data_stats_clear - cellular data stats has been cleared
- cellular_data_traffic_overflow – cellular data traffic reach threshold
- cellular_network_type_changed – cellular network type has been changed.
- poor_signal_quality - signal quality is bad
- link_switch - system is going to switch the link
- wan_up - data connection of WAN has been up
- wan_down - data connection of WAN has been down
- wlan_up - data connection of WLAN has been up
- wlan_down - data connection of WLAN has been down
- wwan_up - data connection of WWAN has been up
- wwan_down - data connection of WWAN has been down
- ipsec_up - IPSec tunnel up
- ipsec_down - IPSec tunnel down
- openvpn_up - OpenVPN tunnel up
- openvpn_down - OpenVPNTunnel down
- robustvpn_up - RobustVPN tunnel up
- robustvpn_down - RobustVPN tunnel down
- gps_up - GPS fixed
- gps_down - GPS fix fail
- lan_port_up - a certain LAN port link up
- lan_port_down - a certain LAN port link down
- usb_connected –USB device was inserted
- usb_removed – USB device was removed
- ddns_update_success - DDNS has been updated successfully
- ddns_update_fail - failed to update DDNS
- sms_received – receive a new message
- sms_sent – message was sent
- sms_cmd_executed – command in the message has been executed
- di1_on – di1 has been enabled
- di1_off - di1 has been disabled
- di1_counter_overflow – di1 counter reach threshold
- di2_on – di1 has been enabled
- di2_off - di1 has been disabled
- di2_counter_overflow – di1 counter reach threshold
- di3_on – di1 has been enabled
- di3_off - di1 has been disabled
- di3_counter_overflow – di1 counter reach threshold
- di4_on – di1 has been enabled
- di4_off - di1 has been disabled
- di4_counter_overflow – di1 counter reach threshold
- ai1_voltage_low – voltage reaches the minimum threshold
- ai1_voltage_high – voltage reaches the maximum threshold
- ai1_current_low – current reaches minimum threshold

- ai1_current_high – current reaches maximum threshold
- ai1_recovery – voltage or current returns to normal

9.7.2 Code Example

```
#include "librouter.h"
#include "event.h" //header file of libevent

void event_msg_proc(int sock, short ev, void *arg)
{
    char *msg;
    char *from;
    char *cmd = NULL;
    char *event = NULL;
    char *desc = NULL;
    char *date = NULL;

    if (OK != rmsg_rcv(&msg, &from))
    {
        return;
    }

    cmd = rmsg_get("cmd");
    if (NULL == cmd
        || strcmp(cmd, "report"))
    {
        return;
    }
    event = rmsg_get("event"); //you should check the pointer for real project
    desc = rmsg_get("desc");
    date = rmsg_get("date");

    // add some code for event handling
    log_info("receive event %s, desc = %s, date = %s", event, desc, date);

    free(cmd); //you should check the pointer for real project
    free(event);
    free(desc);
    free(date);
}

int main(int argc, char **argv)
{
    int sd;
    struct event event;
```

```
daemon(0, 0);

sd = rmsg_init(argv[0]);
event_init();
event_unsubscribe_all();

event_set(&event, sd, EV_READ|EV_PERSIST, event_msg_proc, NULL);
event_add(&event, NULL);

event_subscribe("wwan_up");
event_subscribe("config_changed");

event_dispatch();

return 0;
}
```

9.8 How to Send SMS

```
// Send SMS "hello world" to phone number 123456789:
system("sms 123456789 \"hello world\"");
```

9.9 How to Send Email

Users can use Email for notifications. The emaild daemon process implements the email function, and it can handle the send email request from another process. For user applications, send IPC messages to emaild daemon process with the target mail address and the content.

Please note that the user must first enable and configure the email feature (menu path: Service->Email).

```
// Send Email to test@gmail.com

char email_addresses[]="test@gmail.com";
char buf[]="from router";
rmsg_send("emaild", "cmd", "send", "addresses", email_addresses, "content", buf);
```

9.10 How to Update Firmware of Router

```
// download firmware to router, use curl tools or other methods
// add your code to download firmware

// use sysupgrade command for firmware updating, -q option means quiet
system("sysupgrade -q /path/to/firmware");
```

9.11 How to Import/Export XML Configuration File

```
// download XML configuration file to router
// add your code to download configuration file

// import configuration file, then save and apply
system("uci import /path/to/xmlfile");
system("uci save");
system("uci apply");

// export configuration to a XML file
system("uci export /path/to/xmlfile");

// upload the XML file, use curl/tftp command or other methods
// add your code to upload configuration file
```

9.12 How to Pack a C++ Application

The router does not have C++ library installed by default. You should install libstdc++ as a single APP or pack it with your APP.

```
$ make package/libstdc++/install
```

```
// package/sample_cpp/Makefile
PKG_INSTALL_DEPENDS:=libstdc++
```

9.13 How to Control BLE Module

Note: This chapter about BLE is for router R2110.

Router R2110 integrates with BLE 5.0 module that supports scanning and connection mode. For scanning mode, BLE module can catch the advertising data from the sensors nearby. CPU communicates with BLE module by serial port. User applications can interact with BLE module by a serial command.

Note: The serial commands described below are the low-layer serial commands. We have implement some APIs for this function in the R2110 SDK package. For more detail about the APIs, please refer to sample_ble package.

- **Steps of scanning mode:**

Step1: Open serial port /dev/ttyXRUSB1

Step2: cmd – set BLE module to enter serial command mode

Step3: scan start <timeout_ms > <white_list> // white_list always is 0, example: scan start 5000 0; Send this serial command to start scanning. Once started, the BLE module will continuously output sensor advertising data from the serial port until timeout. The advertising data format: ADV:<mac_addr> <data> <num_of_data_discarded> <RSSI>

For example:

ADV:01E9FAEC7DCA51 020106070941455F424C4511079ECADC240EE5A9E093F3A3B50100406E 0 -51

You also can stop the scanning manually before timeout by serial command “scan stop”.

● **Steps of connection mode:**

Step1: Open serial port /dev/ttyXRUSB1

Step2: cmd – set BLE module to enter serial command mode

Step3: connect <mac_addr> <timeout_ms> <minConnInt_us> <maxConnInt_us> <linkSup_us>

mac_addr: The device's Bluetooth address to connect must be properly formatted and is exactly seven bytes long.

timeout_ms: The length of time in milliseconds that the connection attempt lasts

minConnInt_us: The minimum connection interval in microseconds. The valid range is between 7500 and 4000000 microseconds.

maxConnInt_us: The maximum connection interval in microseconds. The valid range is between 7500 and 4000000 microseconds

linkSup_us: The link supervision timeout for the connection in microseconds.

Example: connect 01C1F755B9A7C2 5000 7500 20000 5000000

This command will return a handle id which indicates a connection session. The result is like below:

```
>connect 01F57C9C307DB1 5000 7500 20000 5000000
```

```
OK
```

```
>
```

```
--- Connect: (0001FF00) handle=1
```

Step4: gattc open 0 0 //open gatt

Step5: gattc tablemap <handle> // This command will return the mapping between service id and UUID

For example:

```
>gattc tablemap 1
```

```
S:1 ,(9) ,FE011800
```

```
C:3 ,00000002 ,FE012A00 ,0 // 3 is service id, 2A00 is UUID
```

Step6: gattc read <handle> <service ID> <data offset>

Example:

```
>gattc read 1 3 0
```

```
OK
```

```
>
```

```
EVATTRREAD(hConn=196352,handle=3,status=0)
```

```
>BleGattcReadData(data=4C4149524420424C363534,offset=0)
```

```
(data=LAIRD BL654)
```

Step7: disconnect <handle> //disconnect a session

9.14 How to Make RobustOS App Realize File Upload Feature

The following takes the sample_uci_upload app as an example to illustrate how to make the ROS app realize the feature of uploading files through the web page.

9.14.1 Determine js File Path of App Configuration Page

You need to log in to the device with root user using ssh or telnet and switch to the `/www/web/js` directory.

Each web page of ROS corresponds to a file in the js directory.

The js files corresponding to the two pages of sample_uci_upload app are

`services_sdk_sample_uci_upload_sample_uci.js` and `services_sdk_sample_uci_upload_status.js`

The rule for naming the file is: `group_menu_tab.js`.

In which all capital letters are converted to lowercase and spaces are replaced with underscores

```
<sample_uci_upload desc="configuration of SDK Sample Uci" group="Services" group_weight="40" menu="SDK Sample Uci Upload" menu_weight="901" oid="1001">
  <option_bool type="bool" desc="Bool Type Option" default="false" tab="Sample Uci" tab_weight="1" area="General Settings" oid="1" />
  <option_string default="test" desc="String Type Option" oid="2" />
  <option_int type="int" desc="Integer Type Option" range="1..100" default="10" row_help="This is an integer type config option." oid="3" />
  <option_enum type="enum" desc="Enum Type Option" range="select1 select2" default="select1" range_desc="Select_1 Select_2" oid="4" />
</sample_uci_upload>
```

As shown above, UCI file corresponding js file name is

`services_sdk_sample_uci_upload_sample_uci.js`

9.14.2 JS File Description

```
$(document).ready(function(){
    $("#rightbody_content").empty();
    add_area_border("services_sdk_sample_uci_upload_sample_uci_area_0", false, "false", false, false,
"false");
    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_bool",
"checkbox", "config", false, "false", "false", "false");
    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_string",
"input", "config", false, "false", "false", "false");
    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_int",
"input", "config", true, "false", "false", "false");
    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_enum",
"select", "config", false, "false", "false", "false");
    get_all_val();
    add_submit_btn_box();
});
```

9.14.3 add_area_border

Add an area that corresponds to the area property in uci.xml. Represents a set of UCI options on the Web page that can be collapsed and expanded.

```
function add_area_border(border_desc, en_help, help_action, rely_condition, rely_action, cmd_script)
```

border_desc: description of the area

en_help: whether to display the area help icon, set to false when not needed

help_action: help text, set to "false" when using the uci.xml area_help content

rely_condition: dependency condition

rely_action: action to be executed when the dependency condition is met, options: "hide", "show"

cmd_script: name of the script to run (special purpose), set to "false"

- **add_element**

Add a UCI option with the specific type defined in the uci.xml file.

```
function add_element(border_desc, top_struct_name, para_desc, type, e_type, en_help, help_action,
rely_condition, rely_action)
```

border_desc: description of the region

top_struct_name: name of the topmost node of uci.xml

para_desc: description of the parameter corresponding to the name of the child node in uci.xml file

type: type, there are: "input", "checkbox", "select"

e_type: element type, set to "config"

en_help: whether to show the area help icon, set to false if not needed

help_action: help text, set to "false" to use the area_help content in uci.xml

rely_condition: dependency condition

rely_action: action to be performed when the dependency condition is met, options: "hide", "show"

- **get_all_val**

Gets all the parameters on the page.

- **add_submit_btn_box**

Add "Submit" and "Cancel" buttons.

9.14.4 Modify JS File

- **Add form form**

Modify the services_sdk_sample_uci_upload_sample_uci.js file to insert the form control code for uploading files before get_all_val().

```
$(document).ready(function(){
    $("#rightbody_content").empty();
    add_area_border("services_sdk_sample_uci_upload_sample_uci_area_0", false, "false", false, false,
"false");
```

```

    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_bool",
"checkbox", "config", false, "false", "false", "false");
    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_string",
"input", "config", false, "false", "false", "false");
    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_int",
"input", "config", true, "false", "false", "false");
    add_element("services_sdk_sample_uci_upload_sample_uci_area_0", "sample_uci_upload", "option_enum",
"select", "config", false, "false", "false", "false");

    /* Add a form to upload file */

    /* Add a new area */
    add_area_border("services_sdk_sample_uci_upload_sample_uci_area_1", false, "false", false, false,
"false");

    /* Add a form */
    var html_str = "<form id=\"id_json_upload_form\" class=\"cs_upload_form\" method=\"post\"
entype=\"multipart/form-data\" target=\"upload_feedback\"> \
    <div class=\"cs_arearow\"> \
    <ul> \
    <li class=\"cs_coltip\" id=\"id_json_upload_tip\" title=\"JSON file\"> \
    <span>JSON</span> \
    </li> \
    <li class=\"cs_colval\"> \
    <span><input type=\"file\" name=\"file\" id=\"id_json_upload_val\"/></span> \
    <span><input type=\"button\" value=\"\" + lang.import_desc + \"\" onclick=\"upload_file();\"/></span> \
    \
    <span><input type=\"hidden\" name=\"hash\" id=\"id_json_upload_hash\"></span> \
    </li> \
    </ul> \
    </div> \
    </form>";
    $("#id_services_sdk_sample_uci_upload_sample_uci_area_1").append(html_str);

    /* Add form end */

    get_all_val();
    add_submit_btn_box();
});

```

The above code adds an area, and the area's content is a form.

The text of the form is "JSON" (span), and the hover message is JSON file (title).

The text of the submit button is "Import", and the upload_file function will be executed when the submit button is clicked.

● Add file upload function

```
function upload_file() {
```

```

// Get import file
var file = $("#id_json_upload_val")[0].files[0];

// Target file name
var uploadFileName = "sample_uci.json";

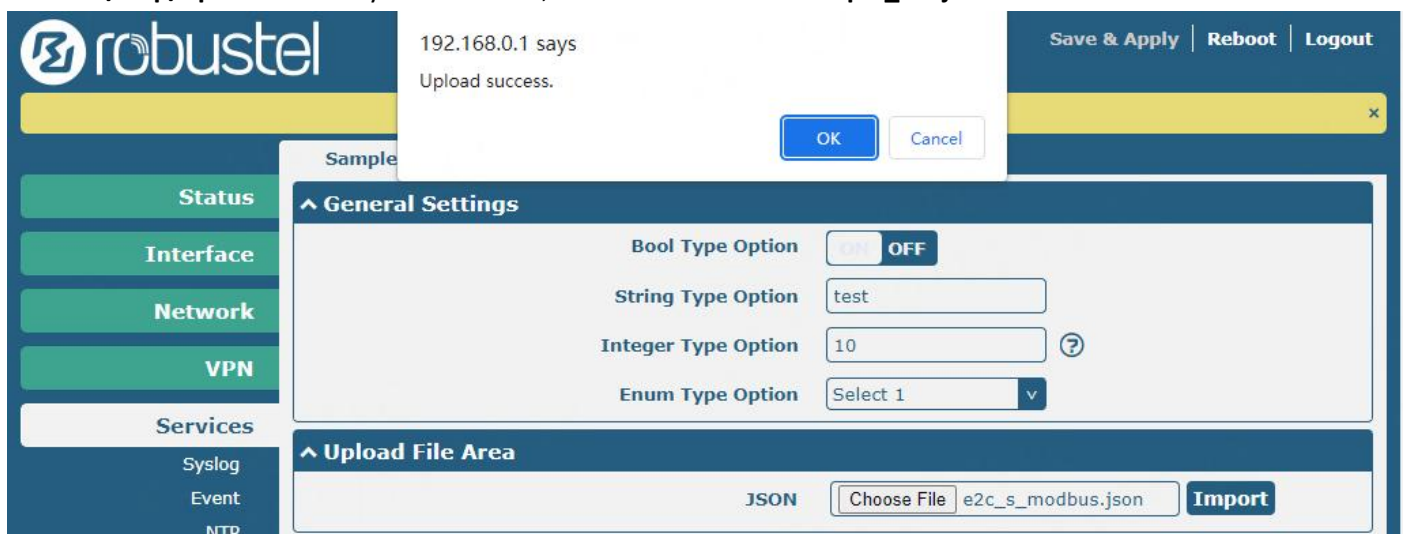
var formData = new FormData();
formData.append('file', file, uploadFileName);
formData.append('hash', ck_id);

$.ajax({
  url: "/action/import_file",
  data: formData,
  type: 'POST',
  cache: false,
  contentType: false,
  processData: false,
  success: function(_res) {
    confirm("Upload success.");
  },
  error: function(_e) {
    confirm("Upload failed.");
  }
});
}

```

Where the **file** variable is the file imported by the user and **uploadFileName** is the name of the file uploaded to the device.

After saving the js file and refreshing the web page, you can realize uploading the file to the router. The uploaded file is in the **/tmp/upload** directory of the device, and the file name is **sample_uci.json**.



9.14.5 Modify uci.xml

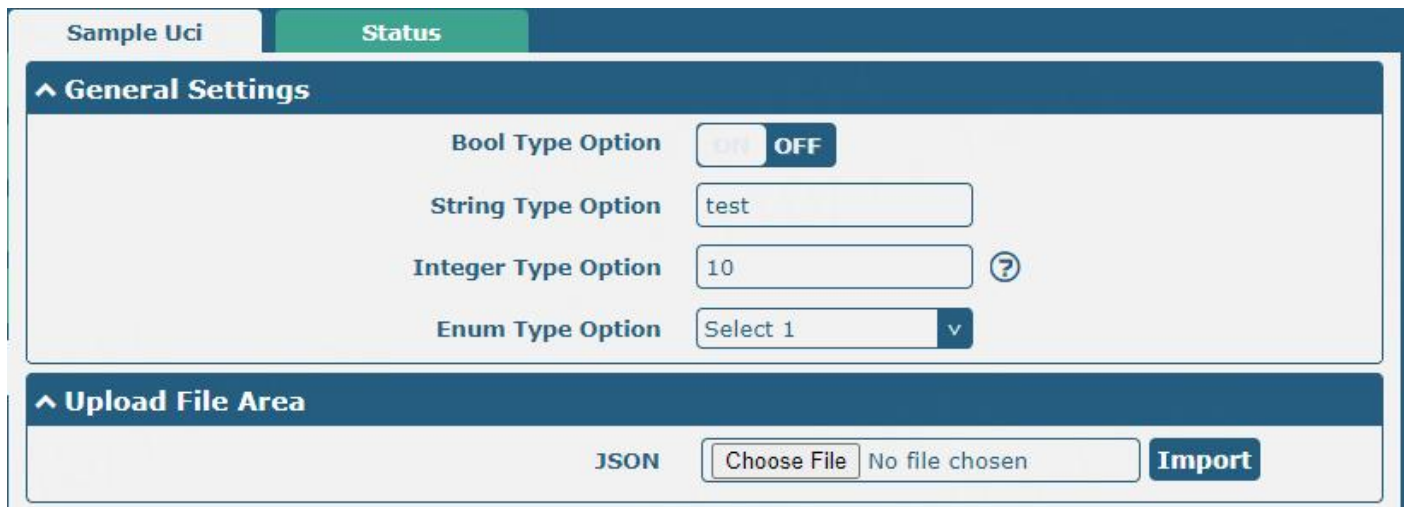
The new AREA title is undefined because there is no corresponding value in uci.xml.

Modify the uci.xml file as follows.

```
<sample_uci_upload desc="configuration of SDK Sample Uci" group="Services" group_weight="40" menu="SDK Sample Uci Upload" menu_weight="901" oid="1001">
  <option_bool type="bool" desc="Bool Type Option" default="false" tab="Sample Uci" tab_weight="0" area="General Settings" oid="1" />
  <option_string default="test" desc="String Type Option" oid="2" />
  <option_int type="int" desc="Integer Type Option" range="1..100" default="10" row_help="This is an integer type config option." oid="3" />
  <option_enum type="enum" desc="Enum Type Option" range="select1 select2" default="select1" range_desc="Select_1 Select_2" oid="4" />

  <upload_file_area desc="Upload File" area="Upload File Area" oid="5" />
</sample_uci_upload>
```

The corresponding Web page is shown below.



9.14.6 Replace Default JS file when Device Starts

The js file can be modified in the device first, and the modified code can be tested after refreshing the browser without recompiling and installing the app.

Once the js file is functionally tested, the final js file needs to be added to the app installation package, and the system default js file needs to be replaced with the new js file when the device is launched.

9.14.7 Add js File to Source Code

Create a new `services_sdk_sample_uci_upload_sample_uci.js` file in `package/sample_uci_upload/file` directory with the contents of the previously tested contents.

9.14.8 Add js File to rpkg Installer

Modify the `package/sample_uci_upload/Makefile` file and add a copy of the js file to `/etc/js` under `Package/Install`

```
define Package/Install
    $(INSTALL_DIR) $(1)/usr/bin/
    $(INSTALL_BIN) $(PKG_BUILD_DIR)/sample_uci_upload $(1)/usr/bin/
    $(INSTALL_DIR) $(1)/etc/sdk/
    $(INSTALL_BIN) files/sdk.sh $(1)/etc/sdk/$(PKG_NAME)
    $(INSTALL_DIR) $(1)/etc/router/uci
    $(INSTALL_DATA) files/uci.xml $(1)/etc/router/uci/$(PKG_NAME).xml
    $(INSTALL_DIR) $(1)/etc/router/usi
    $(INSTALL_DATA) files/usi.xml $(1)/etc/router/usi/$(PKG_NAME).xml
    $(INSTALL_DIR) $(1)/etc/js/
    $(INSTALL_DATA) files/services_sdk_sample_uci_upload_sample_uci.js $(1)/etc/js/
endef
```

... This way, when you install the app, the js files will be installed in the `/app/etc/js` directory. You also need to copy the js file to the `/www/web/js` directory when you start the app.

- **Modify the `sdk.sh` file**

Modify the `package/sample_uci_upload/files/sdk.sh` file and copy the js files in the `/app/etc/js` directory to the `/www/web/js` directory in the start function.

```
#!/bin/sh

. /usr/lib/functions

RETVAL=0
PROG=sample_uci_upload

start()
{
    cp -f /app/etc/js/services_sdk_sample_uci_upload_sample_uci.js /www/web/js
    ${PROG}
    return $?
}
```

After compiling and installing the app, the web page will have upload controls. It will be able to upload files to the `/tmp/upload` directory of the device. However, the files in the `/tmp` directory will be lost after the device is rebooted. You will also need to copy the uploaded files to the `/app` or `/nvm` partition of the device.

Modify the `package/sample_uci_upload/files/sdk.sh` file to

```
#!/bin/sh

. /usr/lib/functions

RETVAL=0
PROG=sample_uci_upload

TARGET_DIR=/nvm/json

start()
{
    cp -f /app/etc/js/services_sdk_sample_uci_upload_sample_uci.js /www/web/js

    mkdir -p ${TARGET_DIR}
    cp -f /tmp/upload/sample_uci.json ${TARGET_DIR}

    ${PROG}
    return $?
}
```

Where `TARGET_DIR` is the path specified by the user to save the uploaded file so that the app can use it when it starts.

9.14.9 File Upload Process

Since the uploaded files need to be copied from `/tmp/upload` partition to `/app` or `/nvm` partition when the app is launched, the operation flow is as follows.

1. Browse the file in the Form control on the web page and click "Import" to upload.
2. Modify the app's UCI parameters and click "Submit". (The purpose is to make Save & Apply highlighted in yellow)
3. Click **Save & Apply** and restart the app.

Chapter 10 Sample Packages

The SDK provides several sample packages for your reference, including the RobustOS sample package, standard open-source library, and third-party cloud platform SDK library and samples.

10.1 RobustOS Sample Packages

These sample packages are closely related to RobustOS. They will call RobustOS API or will access Robustel router’s hardware resource. You can start the development base on the sample code.

Package Type	Package name	Comment
RobustOS samples	sample_can	It is a sample code about the can bus, which receives data from the peer device and then sends it back.
	sample_https	Access the web URL, fetch the source code and save it to a local file.
	sample_mqtt	It is a sample about mqtt. It will subscribe and publish topics using mqtt. MQTT protocol is implemented by using mosquitto library.
	sample_serial	It is a sample code for serial port. In this sample, it will open a serial port, then send and receive data with a peer device.
	sample_dido	This dido sample will listen to DI port status and change DO status according to DI status.
	sample_gps	Get GPS position information by using uci interface.
	sample_shell	It is a sample of developing an SDK application by using shell language.
	sample_uci	It is an example of how to use uci API to get system/application status.
	sample_cpp	Sample about using C++ language on RobustOS SDK.
	example_sqlite	An example about using sqlite

	sample_ble	It is a sample code about how to control R2110 BLE module.
	sample_ai	It is a sample code for obtaining voltage or current data through the AI interface of the R1520.

10.2 Common Libraries

The SDK also provides some open-source libraries that have been ported into the SDK and can be used directly. If you need to use them during development, include the header file and link to the library. On the other hand, if you need to add additional open-source libraries, you can refer to these libraries and port in the new library.

Package Type	Package name	Comment
Common libraries	json-c	Json-c library
	libmodbus	Modbus library
	libpcre	Pcre library
	mosquitto	It is a C library for implementing MQTT clients and mosquitto_pub and mosquitto_sub command line MQTT clients.
	openssl	Openssl library
	paho_mqtt_c	mqtt client library in C
	libstdc++	The C++ Runtime Library
	curl	It provides a library(libcurl) and a command-line tool(curl) for transferring data using various protocols.
	sqlite	SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured SQL database engine.
	c-ares	C library for asynchronous DNS requests (including name resolves)
zlib	A Compression Library	

	e2fsprogs	It provides the filesystem utilities for use with the ext2 filesystem. It also supports the ext3 and ext4 filesystems.
--	-----------	--

10.3 Cloud Platform SDK Libraries

We also integrate some of the popular cloud platform’s SDK packages into our SDK, making it easier to connect to the platform.

Package Type	Package Name	Comment
Thingworx	lib_thingworx	Thingworx library
	tw_dev_properties	Example code about using Thingworx library
	tw_steam_sensor	Example code about using Thingworx library
	tw_steam_sensor_t	Example code about using Thingworx library
	tw_sub_properties	Example code about using Thingworx library
Aws	aws_iot_shadow_sample	AWS IoT demo about shadow
	aws_iot_subscribe_publish_sample	AWS IoT demo(subscribe/publish data to Aws iot hub)
	lib_aws_iot	AWS IoT library
Azure	azure_iot_hub_mqtt	Azure IoT mqtt demo
	azure_iot_sdk	Azure IoT library
Baidu	sample_iot_hub_baidu	A sample code about connecting to Baidu iot-hub

10.4 Package dependencies

Package	Depends
tw_steam_sensor	lib_thingworx
azure_iot_sdk	e2fsprogs
sample_sqlite	sqlite
aws_iot_shadow_sample	lib_aws_iot
sample_mqtt	c-ares mosquitto
tw_subscribed_properties	lib_thingworx
tw_steam_sensor_with_threads	lib_thingworx
azure_iot_hub_mqtt	azure_iot_sdk
sample_iot_hub_baidu	c-ares mosquitto
tw_dev_properties	lib_thingworx
aws_iot_subscribe_publish_sample	lib_aws_iot
sample_modbus	libmodbus

Chapter 11 RobustOS Open Source Components

Name	License	Version	URL
arp-scan	GPL-3.0	1.9.7	https://codeload.github.com/royhills/arp-scan/tar.gz/1.9.7
busybox	GPL-2.0	1.34.1	https://www.busybox.net/downloads
curl	MIT	7.56.0	https://curl.askapache.com/download/
can-utils	GPL-2.0-only OR BSD-3-Clause	2021.08.0	https://github.com/linux-can/can-utils/archive/refs/tags/v2021.08.0.tar.gz
conntrack-tools	GPL-2.0	1.4.6	https://netfilter.org/projects/conntrack-tools/files/conntrack-tools-1.4.6.tar.bz2
crconf	GPL-2.0-only, BSD	pre2	https://sourceforge.net/projects/crconf/files/crconf-pre2.tar.gz/download
QFirehose	N/A	V1.4	N/A
dnsmasq	GPL-2.0	2.86	https://www.busybox.net/downloads
e2fsprogs	GPL-2.0, MIT	1.46.5	http://prdownloads.sourceforge.net/e2fsprogs/e2fsprogs-1.46.5.tar.gz
ethtool	GPL-2.0	5.16	https://mirrors.edge.kernel.org/pub/software/network/ethtool/ethtool-5.16.tar.gz
expat	MIT	2.4.2	https://github.com/libexpat/libexpat/releases/download/R2_4_2/expat-2.4.2.tar.bz2
fping	BSD-4-Clause	5	https://github.com/schweikert/fping/releases/download/v5.0/fping-5.0.tar.gz
pam_tacplus	GPLv2	1.6.1	https://github.com/kravietz/pam_tacplus/
glibc	GPL-2.0 LGPL-2.1	2.24	http://ftp.gnu.org/gnu/glibc/glibc-2.24.tar.gz
gmp	GPL-2.0-or-later	6.2.1	https://gmplib.org/download/gmp/gmp-6.2.1.tar.xz
gcc	GPL-2.0 LGPL-2.1	6.3.0	https://github.com/gcc-mirror/gcc/archive/gcc-6_3_0-release.tar.gz
hostapd	GPL-2.0	2.9	https://w1.fi/releases/hostapd-2.9.tar.gz
backports	GPL-2.0	2017/11/1	N/A
backports	GPL-2.0	v5.10.85-1	N/A
mt76	N/A	2021-12-03-678071ef	https://github.com/openwrt/mt76
wireless-regdb	ISC	master-2021-08-28	https://wireless.wiki.kernel.org/en/developers/regulatory/wireless-regdb
iproute2	GPL-2.0	4.13.0	https://github.com/shemminger/iproute2/archive/refs/tags/v4.13.0.tar.gz
iptables	GPL-2.0	1.8.7	https://www.netfilter.org/projects/iptables/files/iptables-1.8

			.7.tar.bz2
shortcut-fe	GPL-2.0	gce12ca6	https://github.com/coolnowwolf/lede/tree/master/package/lean/shortcut-fe
iperf	BSD-3-Clause	3.11	https://github.com/esnet/iperf/archive/refs/tags/3.11.tar.gz
ipset	GPL-2.0	7.15	https://ipset.netfilter.org/ipset-7.15.tar.bz2
iw	GPL-2.0		https://git.kernel.org/pub/scm/linux/kernel/git/jberg/iw.git/snapshot/iw-3.17.tar.gz
json-c	MIT	0.12.1	https://github.com/json-c/json-c.git
klish	WebM, BSD-3-Clause	1.7.1	https://src.libcode.org/pkun/klish/archive/1.7.1.tar.gz
libevent	BSD-3-Clause	2.0.22	https://github.com/libevent/libevent/releases/download/release-2.0.22-stable/libevent-2.0.22-stable.tar.gz
libmnl	LGPL-2.1	1.0.4	https://www.netfilter.org/projects/libmnl/files/libmnl-1.0.4.tar.bz2
libnetfilter_conntrack	GPL-2.0-or-later	1.0.8	https://netfilter.org/projects/libnetfilter_conntrack/files/libnetfilter_conntrack-1.0.8.tar.bz2
libnetfilter_cthelper	GPL-2.0	1.0.0	https://www.netfilter.org/projects/libnetfilter_cthelper/files/libnetfilter_cthelper-1.0.0.tar.bz2
libnetfilter_cttimeout	GPL-2.0	1.0.0	https://www.netfilter.org/projects/libnetfilter_cttimeout/files/libnetfilter_cttimeout-1.0.0.tar.bz2
libnetfilter_queue	GPL-2.0	1.0.5	https://netfilter.org/projects/libnetfilter_queue/files/libnetfilter_queue-1.0.5.tar.bz2
libnfnetlink	GPL-2.0	1.0.1	https://www.netfilter.org/projects/libnfnetlink/files/libnfnetlink-1.0.1.tar.bz2
libnl-tiny	GPL-2.0 LGPL-2.1	0.1	https://github.com/openwrt/openwrt/tree/openwrt-19.07/package/libs/libnl-tiny
tcpdump	BSD	4.9.2	git://bpf.tcpdump.org/tcpdump
libpcap	BSD-3-Clause	1.10.1	http://www.tcpdump.org/release/
libxml2	MIT	2.9.11	ftp://xmlsoft.org/libxml2/libxml2-2.9.11.tar.gz
libusb	LGPL-2.1	1.0.25	https://github.com/libusb/libusb/releases/download/v1.0.25/libusb-1.0.25.tar.bz2
Linux-PAM	BSD-3c GPL	1.3.0	https://ftp.osuosl.org/pub/blfs/conglomeration/Linux-PAM/Linux-PAM-1.3.0.tar.bz2
Linux kernel	GPL-2.0	3.18.92	https://www.kernel.org/

lora_gateway	N/A	5.0.1	https://github.com/Lora-net/lora_gateway/archive/refs/tags/v5.0.1.tar.gz
LuaJIT	MIT	2.1.0-beta2	https://github.com/LuaJIT/LuaJIT/archive/refs/tags/v2.1.0-beta2.tar.gz
lua-nginx-module	BSD	0.10.2	https://github.com/openresty/lua-nginx-module/archive/refs/tags/v0.10.2.tar.gz
lua-resty-upload	BSD	0.1	https://github.com/openresty/lua-resty-upload/archive/refs/tags/v0.10.tar.gz
mbedtls	Apache-2.0	2.4.0	https://github.com/Mbed-TLS/mbedtls/archive/refs/tags/mbedtls-2.4.0.tar.gz
msmtp	GPL-2.0	1.8.19	https://marlam.de/msmtp/releases/msmtp-1.8.19.tar.xz
mtd-utils	GPL-2.0	2.1.1	http://repository.timesys.com/buildsources/m/mtd-utils/mtd-utils-2.1.1/mtd-utils-2.1.1.tar.bz2
ncurses	MIT	6.1	https://ftp.gnu.org/pub/gnu/ncurses/ncurses-6.1.tar.gz
nginx	BSD-2-Clause	1.22.0	http://nginx.org/download/nginx-1.17.6.tar.gz
ngx_devel_kit	BSD 3-Clause "New" or "Revised" License	0.2.19	https://github.com/vision5/ngx_devel_kit/archive/refs/tags/v0.2.19.tar.gz
ntp	N/A	4.2.8p15	http://www.eecis.udel.edu/~ntp/ntp_spool/ntp4/ntp-4.2/
openssh	BSD ISC	8.8p1	https://github.com/openssh/openssh-portable/archive/refs/tags/V_8_8_P1.tar.gz
openssl	OpenSSL	1.0.2u	https://www.openssl.org/source/openssl-1.0.2u.tar.gz
openvpn	GPL-2.0	2.5.5	https://github.com/OpenVPN/openvpn/archive/refs/tags/v2.5.5.tar.gz
phytool	GPL-2.0	2	https://github.com/wkz/phytool/releases/download/v2/phytool-2.tar.xz
ppp	BSD, BSD-2-Clause, BSD-4-Clause-UC, BSD-style, CMU, CMU-style, GPL, GPL-2.0+, ISC, LGPL, LGPL-2.0+, No_license_found, Public-domain, Public-domain-ref, RSA-Security, See-doc.OTHER, UnclassifiedLicense, X11-style, Zlib,	2.4.7	https://download.samba.org/pub/ppp/ppp-2.4.7.tar.gz

	Zlib-possibility		
Rp-pppoe	LGPL-2.0+	3.15	https://dianne.skoll.ca/projects/rp-pppoe/download/rp-pppoe-3.15.tar.gz
strongswan	GPL-2.0	5.9.5	https://github.com/strongswan/strongswan/releases/download/5.9.5/strongswan-5.9.5.tar.bz2
swconfig	GPL-2.0	12	https://github.com/openwrt/openwrt/tree/openwrt-21.02/package/network/config/swconfig
wpa_supplicant	BSD	2.9	https://w1.fi/wpa_supplicant/
wireguard-tools	GPL-2.0	1.0.20210914	https://github.com/WireGuard/wireguard-tools/archive/refs/tags/v1.0.20210914.tar.gz
wireless_tools	GPL-2.0	29	https://hewlettpackard.github.io/wireless-tools/wireless_tools.29.tar.gz
zlib	Zlib	1.2.11	http://prdownloads.sourceforge.net/libpng/zlib-1.2.11.tar.gz?download