OpenSync™

Building OpenSync with RDK-B

DATE: March 09, 2020

Document number: 019-1129-21

History of Changes

Version	Change
November 29, 2019	Release 1.4.0
March 9, 2020	Release 1.4.0.2

Table of Contents

References

[1] https://www.opensync.io/documentation/

[2] RDK-B OpenSync Integration.pdf

Introduction

OpenSync™ is designed to provide a software defined network - SDN platform, through which it virtualizes the networking and wireless management for easy service roll-out.

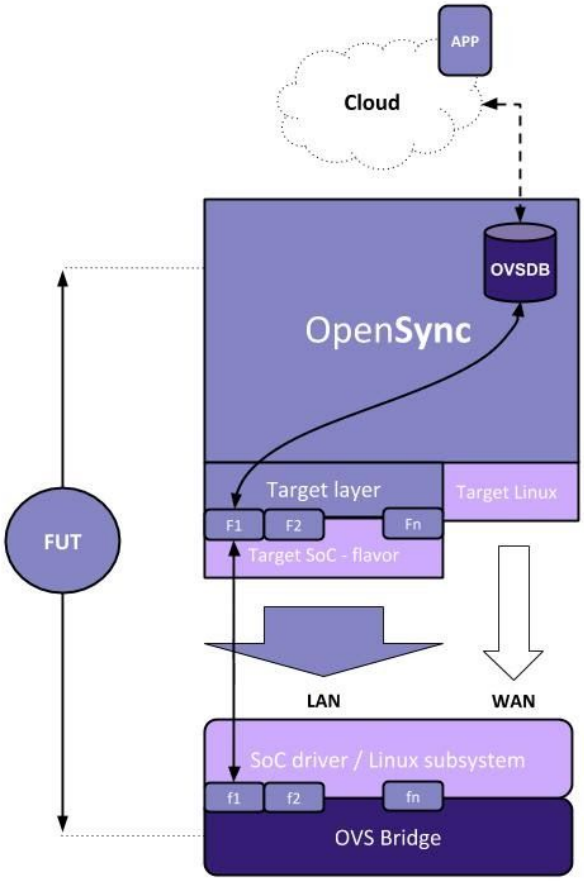

***Figure 1**: OpenSync block diagram*

Devices that can support *OpenSync* are referred to as **targets,** where the **target layer** is an adaptation layer between the *OpenSync* managers and the low-level SoC/Linux drivers.

There are many different **target layer flavors,** which can be specific to a particular chipset (e.g. Broadcom, Qualcomm, Quantenna, Celeno, etc.), or a platform such as RDK, OpenWrt, PRPL, etc.

This document describes how to build the *OpenSync* RDK integration. Further details regarding the *OpenSync* RDK layer can be found in *RDK-B OpenSync Integration* [2].

For more information on OpenSync visit https://www.opensync.io/documentation/ [1].

OpenSync on RDK Platforms

OpenSync for RDK-based platforms is built using three separate components (repositories):

- **core**

  This is an alias for the code in *OpenSync* repository.

- **platform**

  This component implements a common part of **target layer** for RDK-based platforms. The repository is available online under the name opensync-platform-rdk.

- **vendor**

  This repository is intended to provide all vendor-specific code, certificates, configuration files, etc. Vendor (platform integrator) is expected to create and maintain this component. *OpenSync* comes with a vendor repository template available at GitHub under the name opensync-vendor-rdk-template. More on customizing that repository template can be found in following sections.

Dedicated Yocto meta layer for building *OpenSync* with RDK SDKs is available on GitHub as meta-rdk-opensync.

Build Prerequisites

The build should be conducted on a host running one of the Linux distributions supported by Yocto. For more details see Yocto User Manual, and contact your SDK provider as it may impose additional requirements.

Integration Steps

Preparing the vendor repository

The content of opensync-vendor-rdk-template repository is listed below. Vendor (platform integrator) is obliged to modify the highlighted files. Modifications necessary in each file are briefly discussed below the listing.

```
├──── build
│     └──── vendor-arch.mk
├──── kconfig
│     └──── TEMPLATE
├──── ovsdb
│     └──── inet.json
├──── README.md
├──── rootfs
│     └──── model
│         └──── template_model
│             └──── usr
│                 ├──── plume
│                 │     └──── etc
│                 │         └──── certs
│                 │             └──── CERTIFICATES_PLACEHOLDER
│                 └──── sbin
│                     └──── deviceinfo.sh
└──── src
    └──── lib
        ├──── osync_hal
        │     ├──── override.mk
        │     └──── src
        │         └──── osync_hal.c
        └──── target
            ├──── inc
            │     └──── target_template.h
            ├──── override.mk
            └──── src
                ├──── map.c
                ├──── radio.c
                └──── target.c
```

vendor-arch.mk

This file is an element of *OpenSync* build system. It is used to enable/disable and configure builds for specific targets. Lines that have to be changed are marked with ‘TODO’ comments. Read the comments and update the file accordingly.

template_model

Rename this folder to the value of RDK_MODEL variable that was set in “vendor-arch.mk” file.

CERTIFICATES_PLACEHOLDER

Replace this file with appropriate certificates that will allow the device to connect to the *OpenSync* cloud. For more information on how to obtain the certificates contact Plume.

deviceinfo.sh

Implement this shell script according to instructions in *RDK-B OpenSync Integration [2]*.

osync_hal

OpenSync 1.4 introduces a new RDK-specific component - OSync HAL (OpenSync Hardware Abstraction Layer). The OSync HAL exposes many extension points to target API that can be easily overridden in the **vendor** repository. The **platform-rdk** component provides default implementation of all OSync HAL extension points, however they are disabled by default.

Integrator is expected to either provide custom implementations, or explicitly enable default ones by selecting them in kconfig. Otherwise, the following build-time errors are expected:

```
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_inet_get_iface_config'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_inet_create_gre'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_inet_add_to_bridge'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_inet_destroy_gre'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_inet_create_vlan'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_inet_destroy_vlan'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_init'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_ready'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_deinit'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_get_country_code'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_dhcp_resync_all'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_devinfo_get_redirector_addr'
| work/RDKB/lib/libplume.so: error: undefined reference to 'osync_hal_devinfo_get_cloud_mode'
```

To avoid above errors and use all default osync_hal functions, the following config file in the vendor layer (“vendor/template/kconfig/TEMPLATE”) should be created:

```
CONFIG_USE_KCONFIG=y
CONFIG_INET_STATUS_NETLINK_POLL=y
CONFIG_PLATFORM_IS_RDK=y
CONFIG_DEFAULT_OSYNC_HAL=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INIT=y
CONFIG_OSYNC_HAL_USE_DEFAULT_READY=y
CONFIG_OSYNC_HAL_USE_DEFAULT_DEINIT=y
CONFIG_OSYNC_HAL_USE_DEFAULT_FETCH_CONNECTED_CLIENTS=y
CONFIG_OSYNC_HAL_USE_DEFAULT_DEVINFO_GET_CLOUD_MODE=y
CONFIG_OSYNC_HAL_USE_DEFAULT_DEVINFO_GET_REDIRECTOR_ADDR=y
CONFIG_OSYNC_HAL_USE_DEFAULT_DHCP_RESYNC_ALL=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INET_GET_IFACE_CONFIG=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INET_SET_IFACE_CONFIG=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INET_CREATE_GRE=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INET_DESTROY_GRE=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INET_ADD_TO_BRIDGE=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INET_CREATE_VLAN=y
CONFIG_OSYNC_HAL_USE_DEFAULT_INET_DESTROY_VLAN=y
CONFIG_OSYNC_HAL_USE_DEFAULT_GET_COUNTRY_CODE=y
```

target_template.h and target/override.mk

Rename “target_template.h” to a more appropriate name, e.g. related to a specific model or a family of devices. Remember to update the corresponding entry in “override.mk”:

```
UNIT_CFLAGS  += -DTARGET_H=\"target_template.h\"
```

“src” directory and the C source code within

There is a small subset of Target API functionality that has to be implemented in the **vendor** component. The opensync-vendor-rdk-template comes with stubs for the required Target API functions in the “src” directory.

The vendor is expected to implement the missing functionality.

map.c file

The “ifmap” structure in “map.c” must be updated. For more details on interface mapping see *RDK-B OpenSync Integration [2]*.

Setting up the RDK-B SDK

This document describes compiling *OpenSync* on top of the RDKM 2019q3 release. The following steps set up the SDK:

```
repo init -u https://code.rdkcentral.com/r/manifests -m rdkb.xml -b rdkb-2019q3
repo sync -j4 --no-clone-bundle
```

The 2019q3 release comes with a partial wifi_hal implementation (provided by *halinterface* and *hal-wifi-generic* packages). Some of the API functions required by *OpenSync* are missing and integrator is expected to provide the complete wifi_hal implementation. The exact list of required APIs as well as proposed DFS API is available in *RDK-B OpenSync Integration* [2].

Adding the OpenSync meta layer

The next step is to add the *OpenSync* meta layer to the build environment:

```
git clone git@github.com:plume-design/meta-rdk-opensync.git --branch=osync_1.4.0.1
source ./meta-cmf/setup-environment
bitbake-layers add-layer ../meta-rdk-opensync/
```

Default *OpenSync* BitBake recipe downloads all components (**core**, **platform**, and **vendor**) and builds the final package. However, the recipe does not know the location of the **vendor** component, because the vendor is expected to create and maintain it. Therefore the vendor has to alter the default *OpenSync* recipe, and it is recommended to do it by creating a BitBake Append File.

Consider the following example: The goal is to integrate *OpenSync* v1.4.0.1. The *OpenSync* meta layer provides recipe “opensync_1.4.0.bb” that builds precisely that version. The vendor has to create a BitBake Append File called “opensync_1.4.0.bbappend” that specifies the location and SRCREV of the **vendor** component:

```
SRCREV_vendor = "${AUTOREV}"
VENDOR_URI = \
"git://git@domain.com/vendor_name.git;protocol=ssh;branch=master;name=vendor_name;d
estsuffix=git/vendor/vendor_name"
```

The “opensync_1.4.0.bbappend” can be placed in an arbitrary meta layer, however it is recommended to keep it outside of the *OpenSync* meta layer.

OpenSync recipes define two variables holding the build and runtime dependencies: DEPENDS and RDEPENDS respectively. The vendor is obliged to satisfy these requirements. More information on dependencies can be found in *RDK-B OpenSync Integration* [2].

For more information on BitBake refer to Official BitBake User Manual.

Starting a build

OpenSync package can be easily built using the following command:

```
bitbake opensync
```

In order to add *OpenSync* to the image, append the following line to image's recipe:

```
IMAGE_INSTALL += "opensync"
```

Then rebuild the image.