Broadcom Inc
1030 Swabia Court
Suite 400
Durham, NC 27703

SAI Quick start and New platform bringup guide

Version 4.1
Status: Draft

08/06/2018
Tushar Tyagi

Revision

Revision	Date	Author	Changes
1.0	09/22/2015	Tushar Tyagi	Initial version.
2.0	10/20/2016	Tushar Tyagi	Version update. Include public domain release restrictions.
3.0	10/24/2017	Tushar Tyagi	Updates for SAI 3.x with single Lib.
4.0	07/16/2018	Tushar Tyagi	Updates for SAI based directly on SDK with no OpenNSL. Added New platform brigup info.
4.1	08/06/2018	Tushar Tyagi	Update for SDK knet cb module and GPL driver support.

References

S.No	Reference	Author	Date/Version	Link

Contents

1 Introduction

This document provides the information for building and using the SAI adapter source code. This does not include the SAI 'adapter-host' code which is the layer of software that runs above and instantiates the SAI adapter software. This document includes general guidelines and requirements to port BRCM SAI to a new Linux-based Broadcom switching platform.

This document also provides guidelines to be followed for releasing SAI adapter libraries into public domain.

2 Pre-requisites

SAI adapter software runs directly on top of Broadcom's switch SDK.

S/W:

- SAI adapter has been validated on top of SDK version 6.5.x. For the exact SDK version pls check the details in the SAI release notes or code drop.
- The adapter implementation is based upon OCP SAI interface version 1.3

H/W:

- SAI adapter has been validated and supports Dell S6000 box which includes the Trident2 ASIC and the Dell S6100 box which includes the Tomahawk ASIC. The SAI adapter also supports BRCM Trident2, Tomahawk, Tomahawk2, Trident3, Tomahawk3 SVKs.
- Linux version running on the box:
  uname -a
  Linux sonic 3.16.0-5-amd64 #1 SMP Debian 3.16.51-3+deb8u1 (2018-01-08) x86_64 GNU/Linux
- Debian version running on the box:
  lsb_release -a
  No LSB modules are available.
  Distributor ID: Debian
  Description: Debian GNU/Linux 8.0 (jessie)
  Release: 8.0
  Codename: jessie

3 Installation

Extract the SAI tar ball. We include the SAI header files which have been downloaded from the SAI Github. Since the OCP SAI Github repository is still evolving thus we only support and include a specific version of the SAI headers in the tar ball which should not be modified.

4 Release contents

The SAI adapter package contains ‘c’ code, header files, make files, config data and docs. The SDK source is not provided as part of SAI package. SAI package contains a placeholder directory for the SDK src.

SAI source is organized as follows:

<base dir/>src/brcm_*.c
include/brcm_sai_*.h
include/sai/*.h
driver/*
os/*
sdk-src/*
debian/*
output/x86-xgs5-deb80/Makefile, package.cfg
data/*
doc/*.pdf

5 Building the code

- With BRCM SAI version 3.3 and greater, OpenNSL is no longer used and therefore its *NOT needed* to define OPENNSL_BASE prior to building.
- Copy the SDK source tar.gz files (eg sdk-all-6.5.13.tar.gz, sdk-all-6.5.13-gpl-modules.tar.gz) to the <sai base>/sdk-src/ directory
- Define KERNEL_SRC to point to the desired kernel headers
- To make a SAI debian package go to the SAI base directory and issue “fakeroot debian/rules binary”. This will generate the required debian packages in the parent directory. All supported commands:
  - fakeroot debian/rules binary
  - fakeroot debian/rules clean
  - fakeroot debian/rules install
- After the build completes the output directory will contain few new directories like bin and objects.
- The output/<platform/>bin directory will contain the libsai.so.1.0 library. This contains everything including SDK and SAI adapter.
- The output/<platform/>objects directory will contain temporary object files.
- Issuing the ‘clean’ command will remove all temporary and generated files.

6 Runtime requirements

For running the Adapter host, the Adapter host software environment has to ensure that the SDK linux kernel modules have been insmod’ed on the system. Following are the linux kernel objects which are available as part of the SAI package:

- linux-kernel-bde.ko

- linux-user-bde.ko
- linux-bcm-knet.ko
- linux-knet-cb.ko

Kernel module specific steps: (run as root: sudo -s)

- mknod /dev/linux-kernel-bde c 127 0
- mknod /dev/linux-user-bde c 126 0
- mknod /dev/linux-bcm-knet c 122 0
- mknod /dev/linux-knet-cb c 121 0

- insmod linux-kernel-bde.ko dmasize=32M
- insmod linux-user-bde.ko
- insmod linux-bcm-knet.ko
- insmod linux-knet-cb.ko

7 New Platform Support

There are 3 main components involved to bring up SAI on a new Linux based platform:

- Linux kernel
- SAI platform support
- SDK configuration / kernel modules

7.1 Linux Kernel

In order to properly install SAI and the SDK kernel driver objects on the target system, SAI must be compiled against the same kernel version that is installed on the switch. Prior to building SAI, you must first define "KERNEL_SRC" to point to the location of the linux kernel headers matching the switch kernel version. For example, if you were compiling against the 3.16.0.5 kernel, you would define the KERNEL_SRC build variable as following:

KERNEL_SRC=<path_to_header_directory>/usr/src/linux-headers-3.16.0-5-common

NOTE: The SAI 3.3 branch does not package or provide any kernel headers. Kernel headers should be downloaded / extracted to a local directory that can be referenced and included in the SAI build. The Broadcom SDK expects all kernel headers to be located in a s single directory. If multiple kernel header directories exist then symbolic links should be created to support a monolithic KERNEL_SRC header directory structure.

7.2 SAI Platform Support

Platform specific code is contained in multiple directories of the source code tree. For supported platforms, the content of these directories can be found in the ODP package, which can be referenced and adapted to new platforms as appropriate.

- <output/x86-xgs5-deb80>: the name of the directory that receives the output of the build process. This directory contains the top-level makefile for building SAI for this platform and will contain the build binary output following a successful build.

- <os/platform/x86-xgs5-deb80> : platform specific SAI initialization code is present in this directory.

- <os/cpu/x86-64> : This folder is a placeholder for future use and is not used at this time. At this time, SAI only supports x86-64 CPU processors.

7.2.1 Code changes required for adding new platform support

To add support for new platform, the following source code changes needs to be made:

7.2.1.1 Platform initialization code

Under <os/platform/x86-xgs5-deb80/platform_init.c>, add platform initialization code. SAI requires each platform to implement an *int platformInit(sai_init_t *init)* function that initializes SDK/switch device to a default state. After the SDK is loaded and the switch chip is initialized, SAI will invoke a user defined platform configuration file to configure the switch (see switch configuration file section for more details.)

To prevent SAI from loading on an unsupported platform, the *platformInit* routine verifies that the detected Broadcom switching silicon PCI device id is supported and the ONIE platform name is found in the list of supported platforms in the *platform_data[]* data structure.

The PCI device id can be found in linux using the “lspci” utility and searching for the Broadcom PCI controller or consulting the appropriate switching silicon documentation on Broadcom CSP.
The ONIE platform name of the switch can be retrieved in one of the following files in the linux file system:

ONIE Installer = /etc/machine.conf
SONiC Debian = /host/machine.conf
Broadcom built Debian / Ubuntu = /etc/machine.conf

To add support for a new platform, an entry must be made in the *platform_data[]* structure. The structure definition is shown below with the fields documented:

- platform_brd_id board_id - Add the new switch ENUM to platform_init.h
- uint32_t dev_id - PCI device ID of switching silicon
- char *onie_platform - ASCII string of ONIE platform name
- char *description - Description of the board, to be displayed upon detection during driver initialization
- char *platform_script - Optional string array of SDK commands which will be sent after initial SDK initialization. This can be used for any type of SOC initialization (such as external phy configuration) that cannot be performed in the platform configuration file brcm_sai_config.bcm. Since this is a user defined array, the last entry of these strings must be the "NULL" text string.

```
typedef struct platform_data_s
{
    platform_brd_id_t board_id;
    uint32_t  dev_id;
    char *onie_platform;
    board_id_t board_id_f;
    char *description;
    char *platform_script;
} platform_data_t;
```

Examples of these entities can be found by examining the file <os/platform/x86-xgs5-deb80/platform-init.c>

7.2.1.2 Switch configuration file

Create switch configuration file "brcm_sai_config.bcm". This file will need to be placed on the switch in the "/tmp" directory before SAI is invoked. This configuration file will be processed by the SAI platform code and will apply all necessary SDK SOC properties and configurations to enable the switch to function under SAI. This includes, but is not limited to configuring port/lane mappings between front panel numbers and internal logical port numbers, handling pre-emphasis settings, port speeds, etc.

7.3 SDK build configuration and kernel modules

7.3.1 SDK build configuration

Note: SAI 3.3 does not package the SDK source code required for compilation. Visit the Broadcom Customer Support Portal (https://www.broadcom.com/support/) and download the supported SDK GA release files (currently **sdk-all-6.5.13.tar.gz** , **sdk-**

6.5.13-gpl-modules.tar.gz) and place the file in the <sai base>/sdk-src/ directory prior to building SAI.

When compiling SAI and the SDK, the switching silicon support must properly be defined in the file <os/platform/x86-xgs5-deb80/Make.local>. This is a platform-specific SDK customization file that indicates the support for various PHY/Switch devices. The existing Make.local already supports the Trident2, Tomahawk, Tomahawk2, Trident3 and Tomahawk3 switching silicon.

7.3.2 SDK Kernel Modules

In order to run SAI on a Linux based platforms, the following Linux Kernel modules must be installed on the switch. Note that these modules are dependent on the version of underlying Linux kernel.

- linux-kernel-bde.ko : Device Enumerator module. This module abstracts the platform and system-specific aspects of the physical switch device configuration:
    - PCI config access
    - Register access
    - Interrupt handling
    - DMA memory management
    - Physical/Virtual address translation
    - Note: All the above operations are critical for the operation of SAI and requires platform-specific/OS integration knowledge.

- linux-user-bde.ko: Support module for translating device access from the userspace to the kernel. This module is required to run SAI in user space. This module is dependent on linux-kernel-bde.ko.
- linux-bcm-knet.ko: In certain applications, it is desirable to have switch packets go in and out of the operating system's network protocol stack. The KNET (Kernel Network) infrastructure has been implemented for Linux User Mode environment that takes over the DMA and interrupt functions of the Transmit/Receive functions. Packets received from the switch are forwarded to one or more Linux network devices interfaces based on a set of filter rules. Likewise, packets sent out via the Linux network interfaces are sent to the switch based on the VLAN/port association.
- linux-knet-cb.ko: This is an extension callback module for the knet module described above and is used to implement custom filtering and action on the packets based upon application requirements. Once the knet module has done its required processing on the packet it invokes this callback module along with the packet buffer and some user data for enabling any further custom processing.

8 Restrictions on Public Domain Releases

If the SAI library has to be released into the public domain, then the instructions in this section must be strictly followed. Deviating from these steps and releasing the subsequent binaries into the public domain is not compliant with Broadcom licensing.

All symbols belonging to source code that is not available as open source need to be stripped from the released SAI libraries. Following is the required strip command format to be used for the sai library:

strip --strip-unneeded <sai lib>

No SAI code should be modified to include unpublished API references (including enums and flag values) and released as open source.

Note: Info on publishing a CDP release will be available soon.

*** * * END OF DOCUMENT * * ***

