White Box Switch 入門

渡辺 晃佑

2026 年 2 月 26 日

目次

1 White Box Switch

1.1 White Box Switch とは

[1] 通常のスイッチと違い、H/W と S/W が分離しており、ユーザが自由にカスタマイズできるスイッチである。CNF と比べて ASIC といったハードウェア資源があることにより、高性能なネットワーク処理を可能にするが、迅速な機能変更は不可能である。

1.2 ASIC とは

ASIC(Application Specific Integrated Circuit)とは、ユーザの用途に合わせて、必要な機能を組み合わせて設計できる半導体集積回路である。

1.3 BMC と Main CPU の違い

以下の図のように WBS には BMC CPU と Main CPU があり、前者は電源やファンなどの HW 管理を、後者はテーブル・ルールの設定等の SW 管理を行う。さらに BMC は CPU を管理し、CPU は ASIC を制御する。そのため ASIC を動かすためには,Main CPU により制御するが、その際に BMC からのボード情報が必要になる。

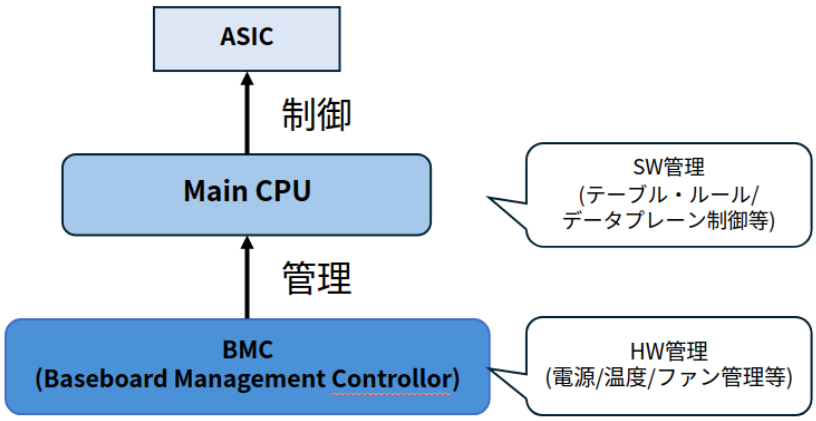


図 1: WBS の構成

2 OS のインストール

2.1 Whitebox Switch の入り方

まず Whitebox Switch の入り方を記す。 tera term 等を使い、シリアル通信により直接 Whitebox Switch に接続する。ssh 接続でもいいが、後に説明する SDE 環境構築といった長時間のビルドの際には適さなない。 ssh 接続が自動で途絶えてしまう可能性ある(経験談)。 Whitebox Switch に入ると以下のようにユーザ名、パスワードを問われる。

```
bmc login: root
Password: 0penBmc （アルファベットのOではなく、数字の0）
```

以上により、Whitebox Switch に入れる。

2.2 OS のインストールの仕方

今回使うネットワーク OS(NOS)として、Open Network Linux(ONL)と SONiC がメジャーである。まず Open Netowork Linux は、NOS の代表であるが、公式は今はサポートしていないため、多くの ONL 関連のサイトのリンクは切れていることが多い(特に github)。また ONL インストール用の bin ファイルは存在するが、SDE 環境構築の際のカーネルヘッダがない。次に SONiC は今でもサポートはされているが、Debian11 をベースにした OS しかなく、SDE の推奨 OS(Debian8,9,10)を満たさない。しかし SONiC に関しては、カーネルヘッダがある。ここで実際に OS のインストール方法を記す。以下の shell を打つことで WBS 内の OS に入ることができる。

```
root@bmc:~# sol.sh
```

Dual Boot になっており、NOS と ONIE という OS が入っているが、ONIE は NOS をインストールするための環境だと思えば良い。実際に ONIE を選択すると、

```
ONIE:/ #
```

以上のようなターミナルが開き、立て続けに多くのログが吐かれる。そこで以下のコマンドを打つとログの出力がなくなる。

```
ONIE:/ # onie-stop
```

2.3 USB を使った OS のインストール方法

サイトで探すとどれも wget を使った OS インストールファイルのダウンロードを推奨しているが、前述したようにリンク切れが多いため、厳しい。そこで OS インストール用のバイナリファイルを各 OS 名のディレクトリに配布する。

2.3.1 SONiC

SONiC は Debian をベースとして作られている。筆者は今回 Debian10 を用いた。OS のイメージファイル(bin)とカーネルヘッダ(deb)は SONiC Image Azure Pipelines@sonic_image から入手した。それらを usb にコピーし、WBS に接続する。次にターミナル上で usb をマウントする。

```
ONIE:/ # onie-nos-install sonic-barefoot.bin
```

以上のようなコマンドで OS をインストールできる。注意点として、バイナリファイルは一度、/tmp 等に移動し、USB をアンマウントを行わないとインストールに失敗する。インストールに成功するとログイン画面が表示されるため、以下のように入力する。

```
Debian GNU/Linux 10 sonic ttyS0

sonic login: admin
Password: YourPaSsWoRd
```

以上が SONiC OS のインストールの手順である。またカーネルヘッダのインストールに関しては、

図 2: SONiC 起動画面

```
sudo apt-get update
sudo apt-get install linux-headers-$(uname -r)
```

以上のような形を推奨されるが、SONiC を含む NOS のカーネルは apt で見つけることができない。そのため、先程の headers ディレクトリの中にある deb ファイルを用いる。

```
admin@sonic:~$ sudo dpkg -i linux-headers-4.19.0-12-2-
common_4.19.152-1_all.deb
admin@sonic:~$ sudo dpkg -i linux-headers-4.19.0-12-2-
amd64_4.19.152-1_amd64.deb
```

以上の形でパッケージをインストールすればよいが、2 つ目のパッケージのインストールするには kbuild と compiler-gcc-8 が必要とエラーが吐かれる。両者とも deb パッケージを探してインストールすればよい。また sudo apt-get update をすると認証がどうのこうのと言われる。そのため、/etc/apt/source.list に以下を書き込む。

```
deb http://debian-archive.trafficmanager.net/debian buster main
deb http://debian-archive.trafficmanager.net/debian-security buster/
updates main
deb http://debian-archive.trafficmanager.net/debian buster-backports main
```

また/etc/apt/sources.list.d/debian_archive_trafficmanager_net_debian.list を削除すると実行できるようになる。筆者の知識不足故理由は不明。

注意事項

- バイナリファイルによってインストールした OS を以下のようにアップグレードすると OS が壊れ、reboot 時に立ち上げられなくなる(実証済み)。

```
sudo apt-get upgrade
```

- Debian10 では最新のパッケージをインストールすると依存関係でエラーがでることが多いため、少しバージョンを下げることをおすすめする。
- 依存ライブラリ等をインストールする際に、何かしらの依存関係でエラーが出る場合、以下のように未解決パッケージを修復するとうまくいくことがある。

```
sudo apt --fix-broken install
```

3 SDE-8.2.2 のビルド

この章では SDE の環境構築の手順を踏むが、これは bf-sde-8.2.2/README を参考にしている。他のバージョンの SDE を使う場合、手順が全く異なるため注意しなければならない。README を参考にしてほしい。

3.1 全体の流れ

Stage	Content
P0	前準備
P1	バイナリパッケージのインストール
P2	bf-syslibs のインストール
P3	bf-utils のインストール
P4	bf-drivers のインストール
P5	switch のインストール
p6	bf-platforms のインストール
P7	p4-build のインストール
p8	p4-example のインストール
p9	ptf-modules のインストール
p10	bf-platforms のインストール

3.2 前準備

SDE 環境構築に必要な最低限のツールをインストールする。以下のコマンドを実行すればよい。設定として bf-sde-x.x.x ディレクトリを SDE に合わせる。

```
export SDE=/home/admin/8.2.x/bf-sde-8.2.2
$SDE/install_min_deps.sh
```

次にすべてのパッケージの解凍として以下のシェルを動かす。

```
./extract_all.sh
```

この際に SDE にディレクトリを指定しないとエラーが吐かれる。インストール先としては、install ディレクトリを指定する。root 権限でインストール先を指定してしまうと後々の依存関係で複雑になる(らしい)。

```
export SDE_INSTALL=$SDE/install
export PATH=$SDE_INSTALL/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/lib:$SDE_INSTALL/lib:$LD_LIBRARY_PATH
```

3.3 各ディレクトリによるビルドの前に

P2 P6 に関しては全て似たようなコマンド実行のため、多少省略するが、随時各ディレクトリの README を参照してほしい。また警告をエラー扱いされる場合がよくあるが、基本警告を無視し

てビルドしてかまわない。次のように make する前に config を設定し、インストールする流れだが、./configure を実行する際に、つけたいオプションをつければよい。README と./configure –help を参照してほしい。

3.4 bf-syslibs のインストール

bf-drivers を動かすために必要なライブラリのインストール用のディレクトリ。libbfsys.[a,la,so]等のファイルが生成されるはず。

```
./configure --prefix=$SDE_INSTALL
make -j$(nproc)
make install
```

3.5 bf-utils のインストール

Barefoot Networks に必要なライブラリのインストール用のディレクトリ。libbfutils.[a,la,so]等のファイルが生成されるはず。

```
./configure --prefix=$SDE_INSTALL
make -j$(nproc)
make install
```

3.6 bf-drivers のインストール

Tofino ASIC を制御するためのドライバをインストールするためのディレクトリ。正しくインストールしていると bf_[kdrv/kpkt/knet]_mod_[un]load が $SDE_INSTALL/install/bin に生成されるはず。

```
./configure --prefix=$SDE_INSTALL
make -j$(nproc)
make install
```

picoc ライブラリがないとエラーが出力された場合、以下のようにディレクトリを指定するとうまくいくかもしれない。単にインストールされていないかもしれないため、確認してほしい。

```
export PYTHONPATH=$SDE/install/lib/python2.7/site-packages:$PYTHONPATH
```

3.7 switch のインストール

標準のスイッチング機能のパイプラインを実装した p4 プログラムのコンパイル用のディレクトリ。実際に install を始めるとコンパイルがとてつもなく遅いことが実感できる。正しくインストールできると./install/share/p4/targets/tofino/に.conf ファイルが生成される。

```
./install_switch_deb.sh
./configure --prefix=$SDE_INSTALL --with-switchapi --with-tofino
make -j$(nproc)
make install
```

3.8 bf-platforms のインストール

このディレクトリは別の bf-reference-bsp-8.2.2 からコピーしてくる。BF の SDE にプラットフォーム固有のポート管理の設定をインストールする。これがないと動かないらしい。正しくインストールできるとbf_port_mgmt 等のファイルが生成される。

```
./configure --prefix=$SDE_INSTALL --with-tofino --with-tof-brgup-plat
make -j$(nproc)
make install
```

ここまでのビルドで run_switchd.sh は動くはずである。このあとのビルドに関してはただ p4c-tofino を使い、コンパイルするだけなので省略する。

3.9 p4-build のインストール

bf-platforms まで正しくビルドできているならば、$SDE_INSTALL/bin/p4c-tofino が生成されているはずである。これよりさきは、p4 プログラムをコンパイルする方法であるそのため、p4c-tofino さえ使えればこれ以上のビルドは必要ない。しかし、p4-build でコンパイルすると./run_switchd.sh を実行するのが簡単である。

```
export P4_NAME=<プログラム名>
export P4_PATH=<p4プログラムがある位置>
./configure --with-tofino --prefix=$SDE_INSTALL
make -j$(nproc)
make install
```

3.10 p4-example のインストール

3.11 ptf-moduiles のインストール

3.12 エラー一覧

```
#inlcude <libio.h>
```

このライブラリがないとエラーを履かれた際はコメントアウトでインストールはできる。

4 switchd

4.1 実際に動かす

下図に表すように ASIC を直接触ることはできないため(筆者の知識不足)、Main CPU(NOS を起動してる CPU)を通じて間接的に制御する必要がある。制御の仕方としては、データプレーンを制御するためのプログラムである P4 プログラムを ASIC にロードする必要がある。

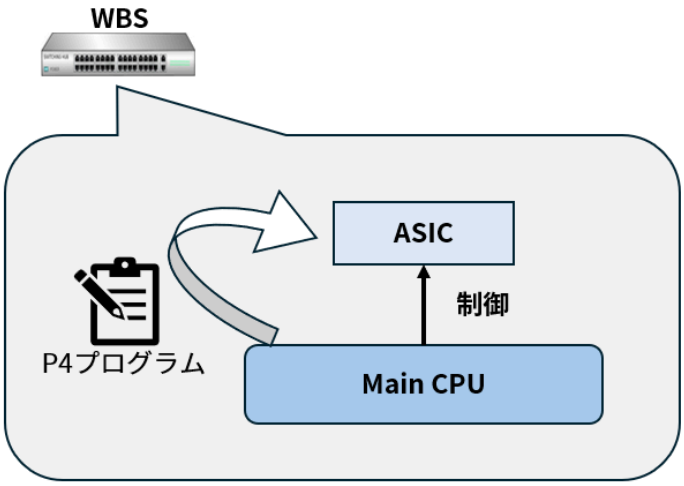


図 3: P4 プログラムをロード

先ほどのビルドを完了したら実際に ASIC に P4 プログラムをロードし、動かしてみる。正しくビルドできると次のようなログが吐かれ、bfshell が起動される。この bfshell というのは Tofino ASIC 用のコマンドラインインターフェース(CLI)で、ASIC 上で動く P4 パイプラインやテーブル操作等を行うツールである(っぽい)。

```
admin@sonic:~/bf-sde-8.2.2$ ./run_switchd.sh -p switch
Using SDE /home/admin/bf-sde-8.2.2
Using SDE_INSTALL /home/admin/bf-sde-8.2.2/install
Using TARGET_CONFIG_FILE /home/admin/bf-sde-8.2.2/install/share/p4/targets/tofino/switch.conf
Using PATH /home/admin/bf-sde-8.2.2/install/bin:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
Using LD_LIBRARY_PATH /usr/local/lib:/home/admin/bf-sde-8.2.2/install/lib:
bf_switchd: system services initialized
bf_switchd: loading conf_file /home/admin/bf-sde-8.2.2/install/share/p4/targets/tofino/switch.conf...
bf_switchd: processing device configuration...
  chip_family: Tofino
Configuration for dev_id 0
  pci_sysfs_str: /sys/devices/pci0000:00/0000:00:03.0/0000:05:00.0
  pcie_domain: 0
  pcie_bus: 5
  pcie_fn: 0
  pcie_dev: 0
  pcie_int_mode: 1
  sds_fw_path: /home/admin/bf-sde-8.2.2/install/share/tofino_sds_fw/avago/firmware
bf_switchd: processing P4 configuration...
P4 profile for dev_id 0
  p4_name: switch
  libpd: /home/admin/bf-sde-8.2.2/install/lib/tofinopd/switch/libpd.so
  libpdthrift: /home/admin/bf-sde-8.2.2/install/lib/tofinopd/switch/libpdthrift.so
  use pi: false
  context: /home/admin/bf-sde-8.2.2/install/share/tofinopd/switch/context.json
  config: /home/admin/bf-sde-8.2.2/install/share/tofinopd/switch/tofino.bin
  non_default_port_ppgs: 0
  Agent[0]: /home/admin/bf-sde-8.2.2/install/lib/libpltfm_mgr.so
  diag:
  mavericks diag:
bf_switchd: library /home/admin/bf-sde-8.2.2/install/lib/tofinopd/switch/libpd.so loaded
bf_switchd: library /home/admin/bf-sde-8.2.2/install/lib/libswitchapi.so loaded
bf_switchd: library /home/admin/bf-sde-8.2.2/install/lib/libpltfm_mgr.so loaded
bf_switchd: agent[0] initialized
Operational mode set to ASIC
Initialized the device types using platforms infra API
ASIC detected at PCI /sys/class/bf/bf0/device
ASIC pci device id is 16
bf_switchd: drivers initialized
\
bf_switchd: dev_id 0 initialized

bf_switchd: initialized 1 devices
bf_switchd: libpd initialized for switch
bf_switchd: switchapi initialized
bf_switchd: spawning cli server thread
bf_switchd: spawning driver shell
bf_switchd: server started - listening on port 9999

        ****************************************
        *      WARNING: Authorised Access Only     *
        ****************************************

bfshell>
```

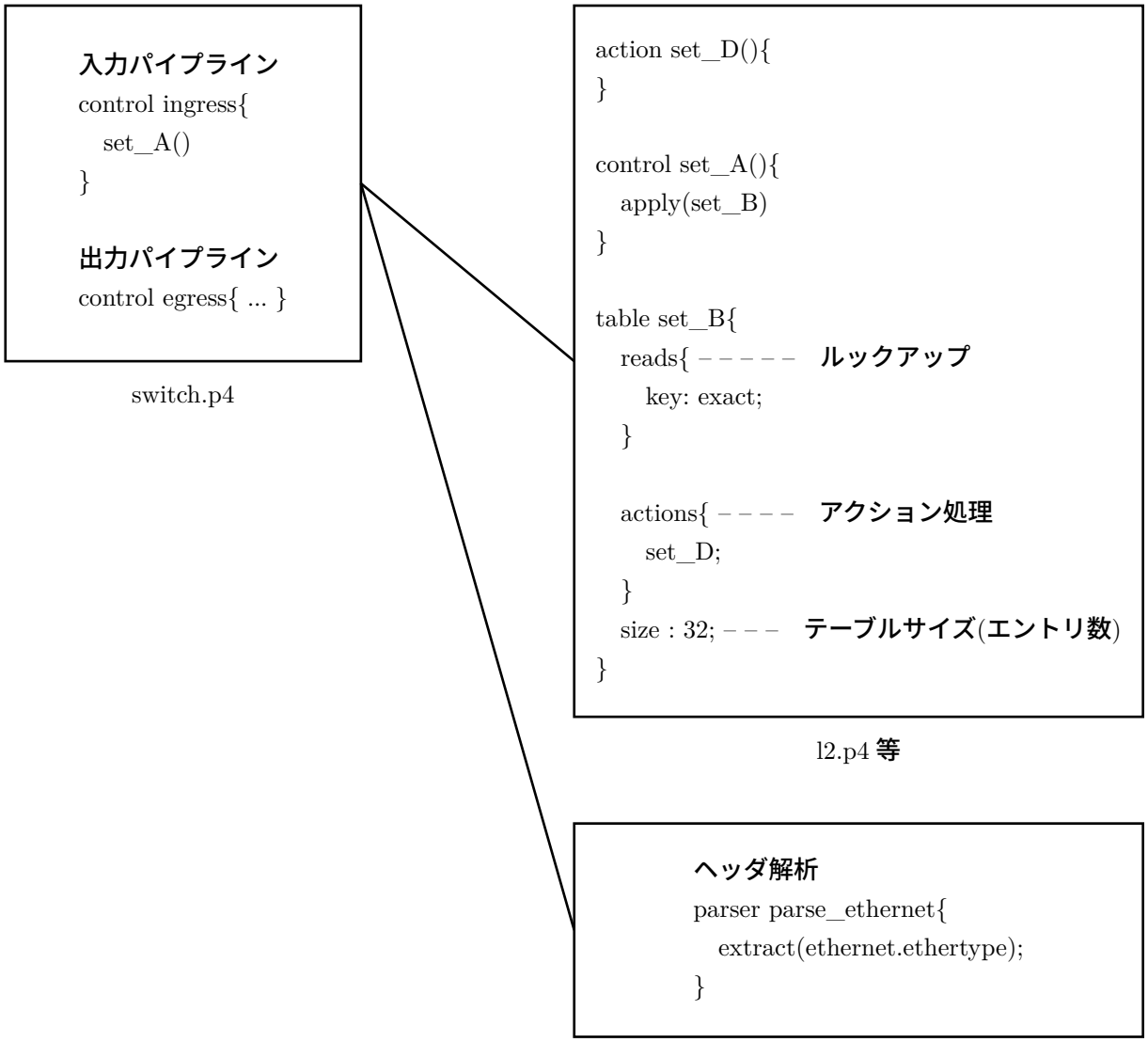
図 4: **動作確認**

4.2 p4 コンパイラ

ASIC にロードするプログラムは P4 言語で書かれるが、P4 プログラムのコンパイラを説明する。コンパイルによって生成されるファイルは 2 つあり、tofino.bin と context.json である。前者は実際に ASIC にロードするバイナリファイルであり、ASIC を動かすためのファイルである。後者はテーブル情報 正しい構文で書かれた p4 プログラムだと以下のようなログが出力される。

```
Parsing successful
Semantic checking successful
```

4.3 P4 プログラムの構成



4.4 あれこれ

4.4.1 アクション命令(action)

- modify_field(A.a,number)

A.a(ヘッダ)に number(値)を格納する。

- add_to_field(ipv4.ttl,-1)

ttl(time to live)カウントを１減らす

4.4.2 ヘッダ型(header_typ)

- ingress_metadata_t
- egress_metadata_t

- intrinsic_metadata_t
- global_config_metadata_t

4.4.3 テーブル構文(table)

-

5　エントリ追加

エントリ追加・削除等の操作をするにはまず bfshell **を起動市、**pd **コマンドを打つ。**

```
bfshell> pd
pd-test:0>
```

6 ucli の一覧

6.1 dump-trace

Display the trace logs

6.2 reset-trace

Clear the trace logs

6.3 set-trace-level

Set the trace level for a module

6.4 set-log-level

set the log level for a module

6.5 get_trace

get_trace

6.6 ver

Display versions of all loaded/Linked BF SDE components

6.7 add-vdev

Add virtual device

6.8 rmv-dev

6.9 dvm

Remove device

- add_dev ： add_dev <asic> <bar>
- rmv_dev ： rmv_dev <asic>
- log_dev ： log_dev <asic> <filepath>
- restore_dev ： restore_dev <asic> <filepath>
- add_port ： add_port <asic> <port>
- rmv_port ： rmv_port <asic> <port>
- cfg ： cfg
- log

6.10 lld

- access

Enter chip access mode.

- find

Find block/registers matching a pattern.

- reg_wr

Write a Tofino register via PCIe: reg_wr <asic> <reg> <value>

- reg_wro

Write (only) a Tofino register via PCIe: reg_wr <asic> <reg> <value>

- reg_rd

Read a Tofino register via PCIe: reg_rd <asic> <reg>

- reg_rdo

Read (but do not decode) a Tofino register via PCIe: reg_rd <asic> <reg>

- reg_rdm

Read (but do not decode) multiple Tofino registers via PCIe: reg_rd <asic> <reg>

- ind_wr

Write a Tofino memory: ind_wr <asic> <reg> <data0> <data1>

- ind_rd

Indirect Read a Tofino memory: ind_rd <asic> <reg>

- mem_rd8

Dump memory as u8's

- mem_rd32

Dump memory as u32's

- mem_rd64

Dump memory as u64's

- find_int_regs

find_int_regs

- int_dump

Dump interrupts

- int_poll

Poll for interrupts <all interrupting>

- int_new

Dump new interrupts

- int_clr

Clear new interrupts

- clr_ints

Clear new interrupts

- new_ints

Dump new interrupts

- int_all

Dump all interrupts

- int_en_all

Enable all interrupts

- int_dis_all

Disable all interrupts

- int_inj_all

Inject all interrupts

- int_stress

Inject all interrupts many times

- int_cb

Test interrupt callbacks

- intx_svc

Poll for interrupts <dev_id> <msk>

- msi_svc

Service an MSI interrupt <dev_id> <msi_int> <msk>

- msix_svc

Service an MSI-x interrupt <dev_id> <msi_x_int> <msk>

- lgset

Set logging parameters

- dma_log

Dump the DMA log

- dump

Dump FIFO and DR summary.

- dr_dump

Dump DR contents.

- cfg_diag

cfg_diag <pipe-mask> <stage-mask>

- wl

issue write-list DMA <42b-register-addr> <# entries> <entry_sz> <alignment> <n_dma>

- wb

issue write-block DMA <addr> <# entries> <entry_sz> <alignment> <n_dma> <fill-pattern>

- wb_mcast

issue write-block DMA <mcast-vector> <addr> <# entries> <entry_sz> <alignment> <n_dma> <fill-pattern>

- rb

issue read-block DMA <addr> <# entries> <entry_sz> <alignment> <n_dma>

- ilist

issue ilist DMA <pipe> <stage> <0-3> <addr> <# entries> <alignment>

- ilist2

issue ilist (rad-capable) DMA <pipe> <stage> <0-3> <addr> <# entries> <alignment>

- memtest

memtest <mau prsr tm> <reg dma> <quick extended>: Run tests on all memories

- int_test

Test interrupts

- holetest

holetest <mau prsr tm chiplvl> <quick extended>: Run tests on all memories

- sysex

sysex <tx_pkt ilist end>

- reg_test

reg_test

- tm_ind_test

tm_ind_test <quick | extended> : TM indirect memory w/r test

- tx_pkt

tx_pkt <len> <cos> <burst_sz>

- diag

Set up the diag event FM DR

- mem_info

dump memory info

- dma_start

Start a DMA descriptor ring (DR)

- dma_service

Service a DMA descriptor ring (DR)

- efuse

Dump EFUSE settings

- cfg_diff

cfg_diff <file-path>

- cfg_load

cfg_load <file-path>

- cfg_verify

cfg_verify <file-path>

- find_tm_mems

Find tm memories in the register hierarchy

- test_tm_mems

Test tm memories in the register hierarchy

- global_ts_enable

Enable/Disable global time stamp <asic> <0: disable 1: enable>

- global_ts_get

Get global time stamp: global_ts_get <asic>

- global_ts_set

Set global time stamp: global_ts_set <asic> <ts_value>

- log

6.11 devdiag

- memtest

Run memory test on MAU/PARDE/TM Usage: -d <device> -i <memory: 0:mau,1:parde,2:tm> -t <test-type: 0:pio,1:dma> -l <len: 0:quick,1:ext> [-m <pattern 0:random,1:zeroes,2:ones,3:checkerboard,4:inv-checkerboard,5:prbs,6:user-defined>] [-a <pattern_data0> -b <pattern_data1>] [-p <log_pipe_bmp>]

- memtest-result

Print result of last run memtest Usage: -d <device>

- regtest

Run register test on MAU/PARDE/TM Usage: -d <device> -i <reg: 0:mau,1:parde,2:tm> -t <test-type: 0:pio,1:dma> -l <len: 0:quick,1:ext> [-p <log_pipe_bmp>]

- regtest-result

Print result of last run regtest Usage: -d <device>

- int-test

Interrupt test Usage: -d <device> [-p <log_pipe_bmp>]

- int-test-result

Print result of last run interrupt test Usage: -d <device>

- log

6.12 port_mgr

- bf_port_add

Add a port <asic> <gb> <fec> <pipe> <port>

- bf_port_rmv

Remove a port <dev> <pipe> <port>

- bf_port_enable

Enable a port <dev> <pipe> <port>

- bf_port_disable

Disable a port <dev> <pipe> <port>

- bf_port_pause_setEnable/Disable

Link PAUSE on a port <dev> <tx_en> <rx_en> <pipe> <port>

- bf_port_pfc_set Enable/Disable

PFC PAUSE on a port <dev> <tx_en_map> <rx_en_map> <pipe> <port>

- bf_port_preamble_len_setSet

custom preamble length on a port <dev> <preamble_len> <pipe> <port>

- bf_port_ifg_set

Set inter-frame gap on a port <dev> <IFG> <1-check whether against IEEE 802.3 allowed ranges,0-no check> <pipe> <port>

- bf_port_mtu_set

Set MTU on a port <dev> <tx_mtu> <rx_mtu> <pipe> <port>

- bf_port_loopback_mode_setSet

loopback mode <dev> <mode: 0=none,1=MAC near,2=MAC far,3=PCS near,4=sds near,5=sds far> <pipe> <port>

- bf_port_txff_truncation_setSet

Tx Fifo truncation settings on a port <dev> <size> <en> <pipe> <port>

- bf_port_txff_mode_setSet

Tx Fifo mode on a port <dev> <crc_chk_dis> <crc_rmv_dis> <fcs_ins_dis> <pad_dis> <pipe> <port>

- bf_port_xoff_pause_time_setSet

XOFF pause time on a port <dev> <xoff_pause_time> <pipe> <port>

- bf_port_xon_pause_time_setSet

XON pause time on a port <dev> <xon_pause_time> <pipe> <port>

- bf_port_force_lf_setSet force LF on/off on a port <dev> <lf_val> <pipe> <port>
- bf_port_force_rf_setSet

force RF on/off on a port <dev> <rf_val> <pipe> <port>

- bf_port_force_idle_setSet force IDLE on/off on a port <dev> <idle_val> <pipe> <port>
- bf_port_stats_clearClear stats on <dev> <pipe> <port>
- bf_port_lf_rf_getGet LF and RF interrupt state on <dev> <pipe> <port>
- bf_port_altrefclk_selSet

clockpad source <dev> <clk_pad: 0=pri 1=sec> <clk_src: 0=none 1=rx_recoverd 2=tx> <pipe> <port>

- bf_port_altrefclk_div_setSet clock pad divider <dev> <clk_pad: 0=pri 1=sec> <divider 1:1, 2:2, 3:3, 4:8> <daisy_sel 0:mac 8-39 1:mac 0-7, 40-63>
- bf_port_tx_clk_selSet

Tx clk source <dev> <clk_src: 0=eth_ref 1=alt_ref> <pipe> <port>

- bf_port_addm

Add multiple ports <dev> <speed/Gbps> <fec> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_rmvm

Remove multiple ports <dev> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_enablem Enable multiple ports <dev> <map[3]> <map[2]> <map[1]> <map[0]>
- bf_port_disablemDisable

multiple ports <dev> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_pause_setmEnable/Disable

Link PAUSE on multiple ports <dev> <tx_en> <rx_en> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_pfc_setmEnable/Disable

PFC PAUSE on multiple ports <dev> <tx_en_map> <rx_en_map> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_preamble_len_setmSet custom preamble length on multiple ports <dev> <preamble_len> <map[3]> <map[2]> <map[1]> <map[0]>
- bf_port_ifg_setmSet

inter-frame gap on multiple ports <dev> <IFG> <1-check whether against IEEE 802.3 allowed ranges,0-no check><map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_mtu_setmSet MTU on multiple ports <dev> <tx_mtu> <rx_mtu> <map[3]> <map[2]> <map[1]> <map[0]>
- bf_port_loopback_mode_setmSet

loopback mode on multiple ports <dev> <mode: 0=none,1=MAC near,2=MAC far,3=PCS near,4=sds near,5=sds far> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_txff_truncation_setmSet Tx Fifo truncation settings on multiple ports <dev> <size> <en> <map[3]> <map[2]> <map[1]> <map[0]>
- bf_port_txff_mode_setmSet Tx Fifo mode on multiple ports <dev> <crc_chk_dis> <crc_rmv_dis> <fcs_ins_dis> <pad_dis> <map[3]> <map[2]> <map[1]> <map[0]>
- bf_port_xoff_pause_time_setmSet

XOFF pause time on multiple ports <dev> <xoff_pause_time> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_xon_pause_time_setmSet

XON pause time on multiple ports <dev> <xon_pause_time> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_force_lf_setmSet force LF on/off on multiple ports <dev> <lf_val> <map[3]> <map[2]> <map[1]> <map[0]>
- bf_port_force_rf_setmSet force RF on/off on multiple ports <dev> <rf_val> <map[3]> <map[2]> <map[1]> <map[0]>
- bf_port_force_idle_setmSet

force IDLE on/off on multiple ports <dev> <idle_val> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_port_stats_clearmClear stats on multiple ports <dev> <map[3]> <map[2]> <map[1]> <map[0]>
- ports Dump port config
- oper Dump oper status of all configured ports
- pcs_ctrs

Dump PCS counters of all configured ports

- qsts Dump all quad related status <dev_port>

- fec

Dump FEC status for all ports with FEC enabled

- but_us

Get bring-up time in microseconds

- but

Get bring-up time

- lt

Dump link-training status <dev> <dev_port>

- qchk

Check a quad for xtalk <dev_port> <passes>

- ch_ena

Dump channel enables of all configured ports

- rmon Dump MAC stats <dev> <mac_block> <ch>
- r_rmon

Dump MAC stats (using register interface) <dev> <mac_block> <ch>

- mac_poll

Poll MAC link state <dev> <mac_block> <ch> <n>

- use_short_timersDefine

the map of ports to use short timers <short_timers_map>

- emu_setup

Set up basic emulation environment (0-15 100g ports)

- warm

Compare sw/hw port cfg <dev> <pipe> <port>

- temp_start

Start a temperature reading <channel>

- temp_get

Complete a temperature reading <channel>

- volt_start

Start a voltage reading <channel>

- volt_get

Complete a voltage reading <channel>

- fp

Display front-port views

- op

Serdes oper status

- an

Serdes AN status

- sd

Serdes APIs

- prbsm

Start or stop PRBS on multiple ports <dev> <1-start,0-stop> <PRBS Speed/Gbps> <map[3]> <map[2]> <map[1]> <map[0]>

- bf_serdes_tx_patsel_setSet

Tx pattern on <dev> <pipe> <port> <lane> <pattern>

- bf_serdes_rx_patsel_setSet

Rx pattern on <dev> <pipe> <port> <lane> <pattern>

- map_dev_port_to_ring_sdMap

dev_port to corresponding ring/sd for serdes commands <dev> <dev_port> <lane>

- warm_init_end

Terminate fast-reconfig

- log

6.13 mc_mgr

- init

Initialize the MC driver.

- show-sess

Show driver sessions.

- create-sess

Create driver sessions.

- destroy-sess

Destroy driver sessions.

- mgrp-create

Create a multicast group.

- mgrp-destroy

Destroy a multicast group.

- node-create

Create a node.

- node-destroy

Destroy a node.

- l1-associate

Associate an L1 node.

- l1-dissociate

Dissociate an L1 node.

- l1-dump

Dump an L1 node.

- ecmp-dump

Dump an ECMP group.

- ecmp-create

Create an ECMP group.

- ecmp-destroy

Destroy an ECMP group.

- ecmp-mbr-add

Add a node to an ECMP group.

- ecmp-mbr-rmv

Remove a node from an ECMP group.

- ecmp-associate

Add an ECMP group to a multicast group.

- ecmp-dissociate

Remove an ECMP group from a multicast group.

- ecmp-modify

Change values in an ECMP group (SW only).

- ecmp-list-all-handles

Display all ECMP handles on device.

- rdm-read

Read the RDM memory.

- rdm-alloc-dump

Dump RDM Block Assignment.

- set-in-prog

Set the API-in-progress flag.

- clr-in-prog

Clear the API-in-progress flag.

- mgrp-show

Show group allocation.

- mgrp-set

Mark an MGID as used.

- mgrp-clr

Mark an MGID as free.

- tbl-ver-rd

Read the table version.

- tbl-ver-wr

Write the table version.

- gbl-rid-rd

Read the global RID.

- gbl-rid-wr

Write the global RID.

- pvt-rd

Read the PVT table.

- pvt-wr

Write the PVT table.

- lit-rd

Read the LIT table.

- lit-wr

Write the LIT table.

- lit-np-rd

Read the LIT NP table from HW.

- lit-np-wr

Write the LIT NP table.

- pmt-rd

Read the PMT table.

- pmt-wr

Write the PMT table.

- mit-rd

Read the MIT table.

- mit-wr-hw

Write the HW MIT table.

- port-mask-rd

Read port mask table.

- port-mask-wr

Write port mask table.

- port-ff-mode-rd

Read port fast failover mode.

- port-ff-mode-wr

Write port fast failover mode.

- backup-port-mode-rd

Read backup port mode.

- backup-port-mode-wr

Write backup port mode.

- backup-port-rd

Read backup port table.

- backup-port-wr

Set backup port.

- port-down-mask-rd

Read port down mask.

- port-down-mask-clr

Clear port down mask.

- cpu-port-rd

Read CPU port info.

- cpu-port-wr

Write CPU port info.

- l1-per-slice-rd

Read L1-per-slice config.

- l1-per-slice-wr

Write L1-per-slice config.

- max-nodes

Write max L1 and L2 config.

- max-l1-rd

Read max L1 config.

- max-l2-rd

Read max L2 config.

- log

6.14 pipe_mgr

- log-ilist

Start/stop logging of ilist contents Usage: log-ilist -d <dev> -e <0/1 to stop/start>

- drv-state

Dump the driver interface state.

- decode-ilist

Decode the contents of an instruction list Usage: decode-ilist -d <dev> -a <address> -l <length in bytes>

- dev

Dump device info. Usage: dev [-d <dev_id>]

- pipe

Dump pipeline info. Usage: pipe -d <dev_id> [-p <pipe_id>]

- tbl

Dump table info. Usage: tbl -d <dev_id> [-h <hdl> [-s <stg>]]

- entry_count

Dump table entry count. Usage: entry_count -d <dev_id> -h <hdl>

- ipv4-route-tcam-add

Add an IPv4 route to the route TCAM

- show-mat-tbl-entry

Dump Match table HW info by entry handle. Usage: show-mat-tbl-entry -d <dev_id> -h <tbl_hdl> -e <entry_hdl> [-p <pipe>] [-i <subindex>]

- show-act-tbl-entry

Dump Action table HW info by entry handle. Usage: show-act-tbl-entry -d <dev_id> -h <tbl_hdl> -e <entry_hdl> -f <act_fn_hdl> -p <pipe> -s <stage>

- show-act-tbl-entry-idx

Dump Action table HW info by entry index. Usage: show-act-tbl-entry-idx -d <dev_id> -h <tbl_hdl> -i <entry_idx> -f <act_fn_hdl> -p <pipe> -s <stage>

- dump-mem

Dump physical HW memory. Usage: dump-mem -d <dev_id> -a <phy_addr>

- dump-mem-full

Dump physical memory. Usage: dump-mem-full -d <dev_id> -p <pipe_id> -s <stage_id> -m <mem_id> -t <mem_type> -l <line_no>

- read-map-ram

Read a mapRAM by unit-id or row+column Usage: read-map-ram -d <dev_id> -p <pipe> -s <stage_id> <-u <Unit id> | -r <row> -c <column>>

- write-map-ram

Write a mapRAM by unit-id or row+column Usage: write-map-ram -d <dev_id> -p <pipe> -s <stage_id> <-u <Unit id> | -r <row> -c <column>> -l <line> -v <value>

- read-unit-ram

Read a unitRAM by unit-id or row+column Usage: read-unit-ram -d <dev_id> -p <phy_pipe> -s <stage_id> <-u <Unit id> | -r <row> -c <column>> -l <line>

- write-unit-ram

Write a unit RAM by unit-id or row+column Usage: write-unit-ram -d <dev_id> -p <phy_pipe> -s <stage_id> <-u <Unit id> | -r <row> -c <column>> -l <line> -v <value_lo> [-v <value_hi>]

- read-virt

Virtually read a logical table Usage: read-virt -d <dev_id> -p <pipe> -s <stage_id> -l <logical table id> -v <vpn> -w <ram word> -t <stats|meter|stateful|selection|idle>

- write-mem

Write to physical HW memory. Usage: write-mem -d <dev_id> -a <phy_addr> -i “data” [-s <start-bit> -e <end-bit>]

- write-tcam

Write to tcam physical memory. Usage: write-tcam -d <dev_id> { </-h <tbl_hdl> -a <phy_addr>> | </-p <phy_pipe> -s <stage_id> -r <row> -c <col> -l <line> } -k “key” -m “mask” -y <payload> -t <mrd> [-b <start-bit> -e <end-bit>]

- write-reg

Write to registers. Usage: write-reg -d <dev_id> -a <reg_addr> -i <data>

- show-act-tbl-vaddr

Dump Action table by Virtual address Usage: show-act-tbl-vaddr -d <dev_id> -a <vpn_addr> -h <tbl_hdl> -f <act_fn_hdl> -p <pipe_id> -s <stage_id>

- show-mat-tbl-vaddr

Dump Match table by Virtual address Usage: show-mat-tbl-vaddr -d <dev_id> -a <vpn_addr> -h <tbl_hdl> -p <pipe_id> -s <stage_id> [-t <is_tind>] [-l <logical_table_id>]

- shadow-mem

Dump shadow memory. Usage: shadow-mem -d <dev_id> -p <pipe_id> -s <stage_id> -m <mem_id> -t <mem_type> [-l <line_no>]

- phv-dump

Dump PHV allocation Usage: phv-dump -d <dev_id> -p <pipe_id> -s <stage_id> -i <direction>

- snap-create

Create a snapshot Usage: snap-create -d <dev_id> -p <pipe_id: all-pipes=0xFFFF> -s <start_stage> -e <end_stage> -i <direction>

- snap-delete

Delete a snapshot Usage: snap-delete -h <handle>

- snap-trig-add

Add a field to snapshot trigger Usage: snap-trig-add -h <handle> -n <field_name> -v <value> -m <mask>

- snap-trig-clr

Delete all fields of snapshot trigger Usage: snap-trig-clr -h <handle>

- snap-state-set

Enable/Disable Snapshot state Usage: snap-state-set -h <handle> -e <enable (0:disable, 1:enable)> [-t <timeout_usec>]

- snap-timer-en

Enable/Disable Snapshot timer Usage: snap-timer-en -h <handle> -e <enable>

- snap-intr-clr

Clear snapshot interrupts Usage: snap-intr-clr -h <handle> [-p <pipe_id> -s <stage_id>]

- snap-hdl-dump

Dump all Snapshot handles on device Usage: snap-hdl-dump -d <dev_id> [-h <handle>]

- snap-cfg-dump

Show Snapshot config Usage: snap-cfg-dump -h <handle> [-p <pipe_id> -s <stage_id>]

- snap-state-get

Show Snapshot state in ASIC Usage: snap-state-get -h <handle> [-p <pipe_id> -s <stage_id>]

- snap-capture-get

Show Snapshot capture from Asic Usage: snap-capture-get -h <handle> [-p <pipe_id> -s <stage_id>]

- tbl-cntr-type-set

Set type for logical table counter Usage: tbl-cntr-type-set -d <dev_id> -p <pipe_id> { -n <tbl_name> | -s <stage> | -a <all_stages> } -t <type: 0=dis,1=tbl-miss,2=tbl-hit,3=gw-miss,4=gw-hit,5=gw-inhibit>

- tbl-cntr-clr

Clear the logical table counter value Usage: tbl-cntr-clr -d <dev_id> -p <pipe_id>{ -n <tbl_name> | -s <stage> | -a <all_stages> }

- tbl-cntr-print

Print logical table counter value Usage: tbl-cntr-print -d <dev_id> -p <pipe_id>{ -n <tbl_name> | -s <stage> | -a <all_stages>}

- batch-begin

Open a batch for a session Usage: batch-begin -s <session>

- batch-end

Close and push a session's batch Usage: batch-end -s <session>

- pbus-irritator

Start/stop background pbus traffic Usage: pbus-irritator -c <start|stop>

- bkgrnd-stat-dump

Start/stop background stat table Usage: bkgrnd-stat-dump -c <start|stop>

- intr_dump

Dump device interrupts. Usage: intr_dump -d <dev_id>

- tcam-scrub-set

Set the tcam scrub timer value Usage: tcam-scrub-set -d <dev_id> -t <msec>

- tcam-scrub-get

Get the tcam scrub timer value Usage: tcam-scrub-get -d <dev_id>

- memid-hdl

Dump memory unit to handle mapping info Usage: memid-hdl -d <dev_id> [-p <pipe_id>] [-s <stage_id>] [-m <mem_id>]

- log

- sel_tbl
- exm_tbl_mgr
- adt_mgr
- stat_mgr
- tcam_tbl
- learn
- idle
- meter_mgr
- stful-mgr
- pkt_path_counter

6.15 traffic_mgr

- log
- cfg_table
- cfg
- usage
- watermark
- dropstatus
- dl_dropstatus
- pfcstatus
- counter
- clr_counter
- clr_watermark
- ut_hitless_cfg
- mirror

6.16 pm

- port-add

<port_str> <speed (1G, 10G, 25G, 40G, 40G_NB, 50G, 100G, 40G_NON_BREAKABLE)> <fec (NONE, FC, RS)>

- port-del

<port_str>

- port-enb

<port_str>

- port-dis

<port_str>

- show

-a -p <port_str> [-d]

- port-stats-clr

<port_str>

- port-stats-timer

stop,1->start

- port-stats-period-set

<period in millisec> min:500

- an-set

<port_str> <0->default,1->enable,2->disable>

- prbs-set

<port_str> <prbs_speed: 10G, 25G> <prbs_mode: 31, 23, 15, 13, 11, 9, 7>

- prbs-clnup

<port_str>

- port-sd-show

<port_str>

- port-sds-cfg

<port_str>

- port-sd-perf

<port_str>

- port-sd-plot-eye

<port_str>

- port-sd-dfe-set

<port_str> <lane/all> <dfe-ctrl><hf_val> <lf_val> <dc_val>

- port-sd-set-tx-eq

<port_str> <lane/all> <pre> <atten> <post> <slew>

- port-sd-rx-inv

<port_str> <lane/all> <polarity>

- port-sd-tx-inv

<port_str> <lane/all> <polarity>

- port-sd-dfe-ical

<port_str> <lane/all>

- port-sd-dfe-pcal

<port_str> <lane/all>

- port-sd-chg-to-prbs

<port_str>

- port-ct-set

<port_str> <0->disable,1->enable>

- fsm-stop
- fsm-go
- fsm-step
- rate-period

<time_period/sec, 0:delete setting>

- rate-show
- port-error-show

-a -p <port_str>

- port-error-clear-a

-p <port_str>

- log

6.17 pkt_mgr

- pkt_stats

pkt_stats <dev>

- pkt_tx_setup

pkt_tx_setup <dev_id> <cos>

- pkt_tx_cleanup

pkt_tx_cleanup <dev_id> <cos>

- pkt_tx

pkt_tx <dev_id> <len> <cos> <pkt_batch_size> <batch_cnt>

- pkt_int_en

pkt_int_en <dev_id> <1: enable, 0: disable>

- log

6.18 switchd

- exit

Exit switchd

- sys_dma2virt

Convert dma address to virtual address

- sys_virt2dma

Convert virtual address to dma address

- pcie_log

Dump register access log

- pcie_logx

Filtered dump of register access log

- pcie_log_clear

Dump register access log

- pcie_log_last

Dump last 'n' entries of register access log

- lrn_int_ena

Enable learn DR processing using interrupts

- log

6.19 bf_pltfm

- log
- bd_cfg

参考文献

[1]「ホワイトボックススイッチのインストール環境である ONIE を仮想マシンに構築する」. [Online]. 入手先: https://gihyo.jp/admin/serial/01/ubuntu-recipe/0665