Application Note

SF2 CFP

Starfighter2 Switch Compact Field Processor (CFP) Application Notes

63XX-SF2-CFP-AN100-R

Corporate Headquarters: San Jose, CA

November 8, 2016

For a comprehensive list of changes to this document, see the Revision History.

Broadcom reserves the right to make changes without further notice to any products or data herein to improve reliability, function, or design. Information furnished by Broadcom is believed to be accurate and reliable. However, Broadcom does not assume any liability arising out of the application or use of this information, nor the application or use of any product or circuit described herein, neither does it convey any license under its patent rights nor the rights of others.

Table of Contents

Purpose 5
CFP Overview 5
ETHSWCTL Command for CFP 6
Command Overview 6
CFP Priority and Index Considerations 6
Operations 6
--read 6
--add 6
--append 6
--insert 6
--delete 6
--delete_all 6
Positions 7
--priority <0-2> 7
--index <0-255> 7
Matching Conditions 7
Acceptable Digital format 7
--spmap <sourcePortMap> 7
--da <hex 6 bytes>,[<mask>] 7
--sa <hex 6 bytes>,[<mask>] 7
--svlan_tag <0-0xffff>,[<mask>] 7
--cvlan_tag <0-0xffff>,[<mask>] 7
--l2 <L2_Framing> 8
--pppoe <1|0> 8
--etype_sap <0-0xffff>,[<mask>] 8
--l3 <L3_Framing> 8
--dscp <0-0x3f>,[<mask>] 8
--ip_protocol <0-255>,[<mask>] 8
--ipsa <IPv4 address>,[<mask>] 8
--ipda <IPv4 address>,[<mask>] 8
--tcp_udp_sport <2 byte value>,[<mask>] 8
--tcp_udp_dport <2 byte value>,[<mask>] 8
Actions 9
--new_dscp_ib <0-0x3f> 9
--chng_fpmap_ib <1-3> 9
--fpmap_ib <0-0x1ff> 9
Examples 9

User Space CFP Programing.. 11
Revision History 14

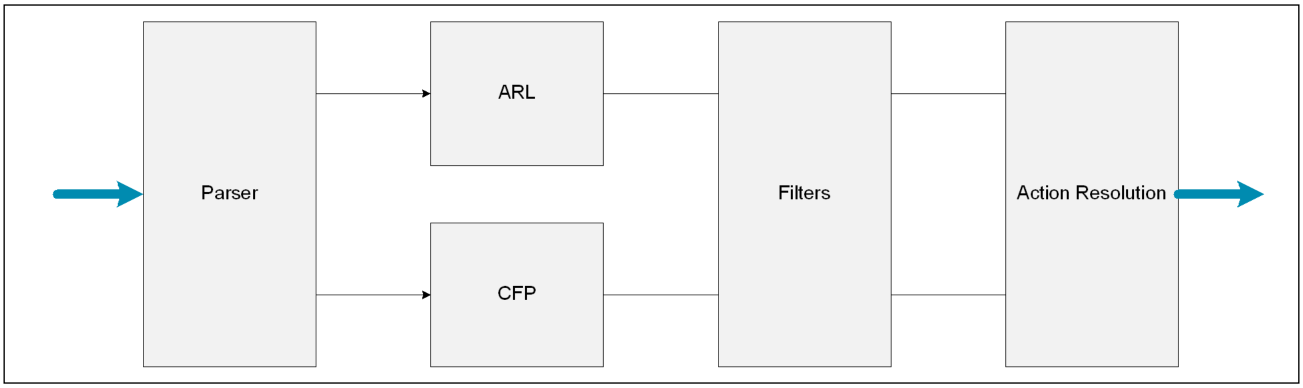
Purpose

This document explains Starfighter2 (SF2) Compact Field Processor (CFP) usage, the command line user interface, and the user space API.

CFP Overview

The Compact Field Processor is designed under the context of the current RoboSwitching architecture, in which **the CFP and ARL are parallel switching control modules and there is no inter-dependency between them**. This is illustrated in the following figure.

Figure 1: CFP Design Context



With CFP, the user can build logic to make switch forwarding decisions based on customized packet parsing logic. The CFP operation has higher privilege than regular ARL decisions, thus, it can be used to override regular operations for special requests. Since VLAN operations are part of ARL operations, this also means the CFP can override VLAN partitioning. In addition, the CFP can also be used to make decisions based on the complicated frame parameters using powerful 255-entry, 232-bit wide TCAM pattern matching capability.

To avoid the use of complicated CFP hardware logic, user-friendly CFP commands and programming API are provided here.

ETHSWCTL Command for CFP

Command Overview

```
ethswctl -c cfp <Operations> [<Position>] <Matching Conditions> <Actions>
```

The ethswctl command arguments contain four groups: <Operations>, [<Position>], <Matching Conditions>, and <Actions>.

CFP Priority and Index Considerations

The CFP can have up to 256 rules managed by priority group and index. Each rule belongs to one of three priority groups from 0 to 2. Priority 2 has the highest priority. The index inside each priority group can have a value from 0 to 255. However, the total rules of the three priority groups can still only add up to 256. The lower index number has the higher priority inside the same priority group.

Note that the index number in each priority group is maintained dynamically by the driver. This means the index will be dynamically adjusted during the operation to always maintain a continuous index number from 0 without a hole. For example, if there are index 0, 1, 2, and 3 four rules in the database currently, and user operation deleted the index 1 rule, the succeeding rules will be moved up, becoming 0, 1, and 2 to fill the hole. This design aims to provide easier users rule management. Always check the current index by using `--read` operation.

Operations

--read

Read CFP rule(s). If `--index` is omitted, all rules in a priority group will be read. If `--priority` is omitted, all rules of three priority groups with the same index will be read. If both `--priority` and `--index` are omitted, all rules will be read out.

--add

Add a rule to the top of rules, thus index 0 of a priority group. If `--priority` is omitted, priority group 0 will be used.

--append

Append a rule to the end of rules of a priority group. If `--priority` is omitted, priority group 0 will be used.

--insert

Insert a rule in front of a specified rule. `--priority` and `--index` are mandatory for this operation.

--delete

Delete a single CFP rule. `--priority` and `--index` are mandatory.

--delete_all

Delete multiple CFP rules. If `--index` is omitted, then all rules of the priority will be deleted. If `--priority` is omitted, then rules of all priorities with the index will be deleted. If both `--priority` and `--index` are omitted, then all rules will be deleted.

Positions

The Position arguments consist of `--priority` and `--index`. Both parameters represent rule matching priorities. The priority has higher privilege than index during the TCAM pattern matching.

--priority <0-2>

Priority Group 0-2; 2 is the highest priority; 0 is the lowest priority. Priority group has higher priority than rule index (0 to 255) inside a priority group.

--index <0-255>

The priority index inside a priority group. Index 0 has the highest priority.

Matching Conditions

Each matching condition argument other than `--spmap` is followed by an optional mask If the mask is omitted, all bits will be matched. Multiple match conditions can be input in one command line.

Acceptable Digital format

The digital format will take the following format in summary.

Table 1: Acceptable Digital format

Field Type	*Format*
MAC address	6-byte hex string: 102233445566
IP address	Regular IP address format: 192.168.1.1,255.255.255.0
Others	Numbers either in hex or decimal, for example, 0x10 or 16

--spmap <sourcePortMap>

The rule will be applied to the port bitmap from where the frame is received. If this argument is omitted, the rule will be applied to all ports.

--da <hex 6 bytes>,[<mask>]

Destination MAC address in hex format.

--sa <hex 6 bytes>,[<mask>]

Source MAC address in hex format.

--svlan_tag <0-0xffff>,[<mask>]

SVLAN(TPID=0x9100) 2 bytes Service Provider VLAN Tag.

--cvlan_tag <0-0xffff>,[<mask>]

CVLAN(TPID=0x8100) 2 bytes Customer VLAN Tag.

--l2 <L2_Framing>

- 0-Ethernet II
- 1-SNAP Public
- 2-LLC
- 3-SNAP Private

--pppoe <1|0>

PPPoE session or not.

--etype_sap <0-0xffff>,[<mask>]

Ethernet Type when L2 = Ethernet II or SNAP_Public, or {DSAP,SSAP} when L2 = LLC or SNAP_Private.

--l3 <L3_Framing>

- 0-IPv4
- 1-IPv6
- 2-Non IP
- Default-IPv4

--dscp <0-0x3f>,[<mask>]

IP DSCP field.

--ip_protocol <0-255>,[<mask>]

IP protocol number.

--ipsa <IPv4 address>,[<mask>]

Source IP address.

--ipda <IPv4 address>,[<mask>]

Destination IP address.

--tcp_udp_sport <2 byte value>,[<mask>]

TCP/UDP source port.

--tcp_udp_dport <2 byte value>,[<mask>]

TCP/UDP destination port.

Actions

The action arguments define what actions will be taken for a packet matching a specific rule. Currently, only new DSCP and new forwarding port bitmaps are supported. Words in-band and out-band are used to describe traffic complying with bandwidth control definitions (in-band), or violating these definitions (out-band). Since QoS bandwidth control is not currently supported, all traffic belongs to in-band traffic.

--new_dscp_ib <0-0x3f>

New in-band traffic's DSCP will be set to frame once matched.

--chng_fpmap_ib <1-3>

Change in-band traffic Forwarding Port Map.

1. Remove DST_MAP from ARL result.
2. Replace ARL result with DST_MAP.
3. Add DST_MAP to ARL result.

If this argument is omitted and the `--dst_fpmap_ib` argument is provided, replacement operation 2 will be performed.

--fpmap_ib <0-0x1ff>

New in-band Forwarding Port Map used by `--chng_fwd_map_ib`.

Examples

```
ethswctl -c cfp --add --da 102233445566 --chng_fpmap_ib 2 --fpmap_ib 0x0
```

Above rule will discard matched DA packets. The rule will be added to priority group 0, index 0.

```
ethswctl -c cfp --append --ipda 192.168.0.0,255.255.0.0 --chng_fpmap_ib 2 --fpmap_ib 0x108
```

Above rule will force packets with destination IP address 192.168.X.Y to ports 8 and 3. The rule will be added to the tail of priority group 0.

```
ethswctl -c cfp --append --priority 0 --spmap 0x1 --chng_fpmap_ib 2 --fpmap_ib 0x0
```

Above rule will force all packets on port 1 to be discarded. The rule will be added to the tail of priority group 0.

```
ethswctl -c cfp --append --priority 0 --spmap 0x1 --chng_fpmap_ib 2 --fpmap_ib 0x2 --da 102233445566
```

Above rule will force packets on port 0 with destination MAC address 10:22:33:44:55:66 forwarded to port 1.

```
ethswctl -c cfp --insert --index 1 --priority 0 --spmap 0x1 --chng_fpmap_ib 2 --fpmap_ib 0x2 --da
102233445566,fffff
```

Above rule will force packets from port 0 with destination MAC address xx:xx:xx:xx:55:66 to be forwarded to port 1. The rule is inserted in front of priority group 0, index 1.

```
ethswctl -c cfp --append --priority 0 --ipda 192.168.1.3,255.255.0.0 --chng_fpmap_ib 0x2 --fpmap_ib
0x8
```

Above rule will force frames with IP destination address 192.168.xxx.xxx to be forwarded to port 3.

```
ethswctl -c cfp --append --priority 0 --chng_fpmap_ib 2 --fpmap_ib 0x100 --da
0180c2000000,ffffff000000
```

Above rule will force packets with destination MAC address of 01:80:c2:xx:xx:xx, which is IEEE 802.1 reserved multicast address, forwarded to port 8, in other words, IMP port.

```
ethswctl -c cfp --read --priority 0
```

Read all CPF rules out from priority group 0.

Example: Screen output.

```
CFP rules: Priority 0
==============================================================================
  --index 0 --l3 0 --ipsa 10.10.1.2 --ipda 192.168.0.0,255.255.0.0 --chg_fpmap_ib 2 --fpmap_ib 0x8
   --index 1 --da 0180c2000000,ffffff000000 --l3 2 --chg_fpmap_ib 2 --fpmap_ib 0xa
   --index 2 --da 0180c2000000,ffffff000000 --l3 2 --chg_fpmap_ib 2 --fpmap_ib 0xa
   --index 3 --da 000000000002 --l3 2 --chg_fpmap_ib 2 --fpmap_ib 0x8

Success.
#
```

```
ethswctl -c cfp --delete --priority 0 --index 1
```

Delete CPF rule of priority 0 index 1. This will cause the succeeding CFP rules to be moved up to fill the deleted rule position in index 1.

```
ethswctl -c cfp --delete_all --priority 0
```

Delete all CFP rules in priority 0.

```
ethswctl -c cfp --delete_all
```

Delete all CFP rules.

```
ethswctl -c cfp --add --priority 0 --chng_fpmap_ib 2 --fpmap_ib 0x8 --tcp_udp_sport 1024
```

Above rule will force packets from any port with TCP or UDP source port number 1024 to be forwarded to port 3.

```
ethswctl -c cfp --add --priority 0 --chng_fpmap_ib 2 --l3 2 --fpmap_ib 0x8 --etype_sap 0x0806
```

Above rule will force non-IP packet with EtherType 0x0806, that is, ARP packet, forwarded to port 3.

```
ethswctl -c cfp --add --priority 0 --chng_fpmap_ib 2 --ipsa 10.10.1.2 --ipda 192.168.1.5,255.255.0.0
--fpmap_ib 8
```

Above rule will force IP packets with source address of 10.10.1.2 and destination IP address of 192.168.xx.xx to be forwarded to port 3.

```
ethswctl -c cfp --add --priority 0 --spmap 0x1 --chng_fpmap_ib 3 --fpmap_ib 0x8
```

Above rule will mirror all traffic received on port 0 to port 3.

```
ethswctl -c cfp -add --priority 0 --da 010022334455 --chng_fpmap_ib 1 --fmap_ib 0x100
```

Above rule will force multicast frames with a destination MAC address of 010022334455 to be sent to ports excluding port 8 IMP port.

User Space CFP Programing

The user space program can perform any command line operations through API function `int bcm_cfp_op(cfpArg_t *cfpArg)`.

```
/* Defined at bcmdrivers/opensource/include/bcm963xx/bcm/bcmswapitypes.h */
typedef struct cfpArg_s {
    unsigned int rc;
    unsigned int argFlag;
    unsigned int unit;
    unsigned int priority;
    unsigned int index;
    unsigned int l2_framing;
    unsigned int l3_framing;
    unsigned int chg_fpmap_ib;
    unsigned int fpmap_ib;
    unsigned int spmap;
    unsigned int pppoe;
    unsigned int etype_sap;
    unsigned int etype_sap_mask;
    unsigned int svtag;
    unsigned int svtag_mask;
    unsigned int cvtag;
    unsigned int cvtag_mask;
    unsigned int ip_protocol;
    unsigned int ip_protocol_mask;
    unsigned int ipsa;
    unsigned int ipsa_mask;
    unsigned int ipda;
    unsigned int ipda_mask;
    unsigned int tcpudp_sport;
    unsigned int tcpudp_sport_mask;
    unsigned int tcpudp_dport;
    unsigned int tcpudp_dport_mask;
    unsigned int dscp;
    unsigned int dscp_mask;
    unsigned int new_dscp_ib;
    unsigned int op;
    unsigned long long da;
    unsigned long long da_mask;
    unsigned long long sa;
    unsigned long long sa_mask;
} cfpArg_t;
enum {
    CFP_RC_SUCCESS,
    CFP_RC_NON_EXISTING_INDEX,
    CFP_RC_CFP_FULL,
    CFP_RC_UDF_FULL,
};
```

```
enum {CfpL3Ipv4, CfpL3Ipv6, CfpL3NoIP};
enum {CfpL2EtherII, CfpL2SnapPublic, CfpL2LLC, CfpL2SnapPrivate};

enum {
    CFPOP_READ,
    CFPOP_READ_NEXT,
    CFPOP_ADD,
    CFPOP_INSERT,
    CFPOP_APPEND,
    CFPOP_DELETE,
    CFPOP_DELETE_ALL,
};

enum {
    CFP_ARG_SPMAP_M =    (1<<0),
    CFP_ARG_DA_M  =      (1<<1),
    CFP_ARG_SA_M  =      (1<<2),
    CFP_ARG_IP_PROTOCOL_M =  (1<<3),

    CFP_ARG_L2_FRAMING_M = (1<<4),
    CFP_ARG_L3_FRAMING_M = (1<<5),
    CFP_ARG_DSCP_M = (1<<6),
    CFP_ARG_PRIORITY_M = (1<<7),

    CFP_ARG_INDEX_M = (1<<8),
    CFP_ARG_IPSA_M = (1<<9),
    CFP_ARG_IPDA_M = (1<<10),
    CFP_ARG_TCPUDP_SPORT_M = (1<<11),

    CFP_ARG_TCPUDP_DPORT_M = (1<<12),
    CFP_ARG_NEW_DSCP_IB_M = (1<<13),
    CFP_ARG_CHG_FPMAP_IB_M = (1<<14),
    CFP_ARG_FPMAP_IB_M = (1<<15),

    CFP_ARG_SVLAN_TAG_M = (1<<16),
    CFP_ARG_CVLAN_TAG_M = (1<<17),
    CFP_ARG_PPPOE_M     = (1<<18),
    CFP_ARG_ETYPE_SAP_M = (1<<19),

    CFP_ARG_OP_M =        (1<<20),
};
```

The API function below to handle all CFP requests is declared in `userspace/private/include/ethswctl_api.h` `int bcm_cfp_op(cfpArg_t *cfpArg)`.

The following example shows how to force a packet with destination MAC address 01:80C2:xx:xx:xx to be forwarded to IMP port 8. More examples can be found in the ethswctl command line program in `userspace/apps/ethswctl/ethswctl.c`.

```
int bcm_cfp_sample(void)
{
    int rc;
    cfpArg_t _cfpArg, *cfpArg = &_cfpArg;
    memset(0, cfgArg, sizeof(*cfpArg));

    /* Set operations */
    cfpArg->op = CFPOP_ADD;
    cfpArg->unit = 1; /* Set switch in unit 1 */
    cfpArg->da = 0x0180c2000000LL;
    cfpArg->da_mask = 0xffffff000000LL;
    cfpArg.l3_framing = CfpL3NoIP;
    cfpArg->chg_fpmap_ib = 2; /* Set ARL replacement operation */
    cfpArg->fpmap_ib = 0x100; /* set destination port to IMP port */

    /* Set operation flag */
    cfpArg->argFlag = CFP_ARG_OP_M|CFP_ARG_DA_M|
        CFP_ARG_L3_FRAMING_M|CFP_ARG_CHG_FPMAP_IB_M;

    /* Call CFP API functin to create rule */
    rc = bcm_cfp_op(cfpArg);

    if (rc) {
        printf(stderr, "Operation failed.\n");
        goto out;
    }

    /* Check return value stored in cfpArg->rc */
    switch (cfpArg->rc) {
        case CFP_RC_SUCCESS:
            printf("Operation success.\n");
            break;
        case CFP_RC_CFP_FULL:
            fprintf(stderr, "Failed: CFP TCAM resource is full\n");
            rc = -1;
            break;
        case CFP_RC_UDF_FULL:
            fprintf(stderr, "Failed: CFP UDF resource is full\n");
            rc = -1;
            break;
    }
out:
    return rc;
}
```

Revision History

Revision	*Date*	*Change Description*
63XX-SF2-CFP-AN100-R	November 8, 2016	Initial release

BROADCOM®
Web: www.broadcom.com
Corporate Headquarters: San Jose, CA