Appendix B

Switch API

Our switch-API is modeled on version 1.0 of the OpenFlow protocol for packet switches. It covers the components and basic functions of circuit switches based on switching time-slots, wavelengths and fibers. It also covers hybrid-switches with both packet and circuit interfaces and/or switching fabrics; and includes provisions for mapping packet-flows to circuit-flows. The entire API is presented in [36] – here we cover the main features.

Describing a Physical Circuit Port

```
struct ofp_phy_cport {
    uint16_t port_no;
    uint8_t hw_addr[OFP_ETH_ALEN];
    char name[OFP_MAX_PORT_NAME_LEN]; /* Null-terminated */

    uint32_t config;        /* Bitmap of OFPPC_* flags. */
    uint32_t state;         /* Bitmap of OFPPS_* flags. */

    /* Bitmaps of OFPPF_* that describe features.  All bits zeroed if
     * unsupported or unavailable. */
    uint32_t curr;          /* Current features. */
    uint32_t advertised;    /* Features being advertised by the port. */
    uint32_t supported;     /* Features supported by the port. */
    uint32_t peer;          /* Features advertised by peer. */

    /* Extensions for circuit switch ports */
    uint32_t supp_sw_tdm_gran; /* TDM switching granularity OFPTSG_* flags */
    uint16_t supp_swtype;      /* Bitmap of switching type OFPST_* flags */
    uint16_t peer_port_no;     /* Discovered peer's switchport number */
    uint64_t peer_datapath_id; /* Discovered peer's datapath id */
    uint16_t num_bandwidth;    /* Identifies number of bandwidth array elems */
    uint8_t  pad[6];           /* Align to 64 bits */
    uint64_t bandwidth[0];     /* Bitmap of OFPCBL_* or OFPCBT_* flags */

};
```

/* following capabilities are defined for circuit switches */

```
    OFPC_CTG_CONCAT     = 1 << 31, /* Contiguous concat on all TDM ports. */
    OFPC_VIR_CONCAT     = 1 << 30, /* Virtual concat on all TDM ports. */
    OFPC_LCAS           = 1 << 29, /* Link Capacity Adjustment Scheme */
    OFPC_POS            = 1 << 28, /* Packet over Sonet adaptation */
    OFPC_GFP            = 1 << 27, /* Generic Framing Procedure */
    OFPC_10G_WAN        = 1 << 26  /* native transport of 10G_WAN_PHY
                                      on OC-192 */
```

/* The following have been added for WAN interfaces */

```
    OFPPF_X          = 1 <<20, /* Dont care applicable to fiber ports */
    OFPPF_OC1        = 1 <<21, /* 51.84 Mbps OC-1/STM-0 */
    OFPPF_OC3        = 1 <<22, /* 155.52 Mbps OC-3/STM-1 */
    OFPPF_OC12       = 1 <<23, /* 622.08 Mbps OC-12/STM-4 */
    OFPPF_OC48       = 1 <<24, /* 2.48832 Gbps OC-48/STM-16 */
    OFPPF_OC192      = 1 <<25, /* 9.95328 Gbps OC-192/STM-64 */
    OFPPF_OC768      = 1 <<26, /* 39.81312 Gbps OC-768/STM-256 */
    OFPPF_100GB      = 1 <<27, /* 100 Gbps */
    OFPPF_10GB_WAN   = 1 <<28, /* 10 Gbps Ethernet WAN PHY (9.95328 Gbps) */
    OFPPF_OTU1       = 1 <<29, /* OTN OTU-1 2.666 Gbps */
    OFPPF_OTU2       = 1 <<30, /* OTN OTU-2 10.709 Gbps */
    OFPPF_OTU3       = 1 <<31  /* OTN OTU-3 42.836 Gbps */
```

Specifying a Cross-Connection

```
struct ofp_connect {
  uint16_t wildcards;              /* identifies ports to use below */
  uint16_t num_components;         /* identifies number of cross-connects
                                      to be made - num array elems */
  uint8_t  pad[4];                 /* Align to 64 bits */

  uint16_t in_port[0];             /* OFPP_* ports - real or virtual */
  uint16_t out_port[0];            /* OFPP_* ports - real or virtual */

  struct ofp_tdm_port in_tport[0];   /* Description of a TDM channel */
  struct ofp_tdm_port out_tport[0];

  struct ofp_wave_port in_wport[0];  /* Description of a Lambda channel */
  struct ofp_wave_port out_wport[0];
};
```

Specifying a TDM port

```
struct ofp_tdm_port {

  uint16_t tport;      /* port numbers in OFPP_* ports */
  uint16_t tstart;     /* starting time slot */
  uint32_t tsignal;    /* one of OFPTSG_* flags */
};
```

/* Minimum switching granularity of TDM physical ports available in a datapath. */

```
enum ofp_tdm_gran {
    OFPTSG_STS_1,        /* STS-1    / STM-0    */
    OFPTSG_STS_3,         /* STS-3    / STM-1    */
    OFPTSG_STS_3c,        /* STS-3c   / STM-1    */
    OFPTSG_STS_12,        /* STS-12   / STM-4    */
    OFPTSG_STS_12c,       /* STS-12c  / STM-4c   */
    OFPTSG_STS_48,        /* STS-48   / STM-16   */
    OFPTSG_STS_48c,       /* STS-48c  / STM-16c  */
    OFPTSG_STS_192,       /* STS-192  / STM-64   */
    OFPTSG_STS_192c,      /* STS-192c / STM-64c  */
    OFPTSG_STS_768,       /* STS-768  / STM-256  */
    OFPTSG_STS_768c       /* STS-768c / STM-256c */
};
```

Specifying a lambda channel

```
struct ofp_wave_port {
  uint16_t wport;      /* restricted to real port numbers in OFPP_* ports */
  uint8_t  pad[6];     /* align to 64 bits */
  uint64_t wavelength; /* use of the OFPCBL_* flags */
};
```

```
enum ofp_port_lam_bw {
    OFPCBL_X       = 1 << 0,/* 1 if fiber switch, 0 if wavelength switch */
    OFPCBL_100_50  = 1 << 1,/* 1 if 100GHz channel spacing, 0 if 50GHz */
    OFPCBL_C_L     = 1 << 2,/* 1 if using C-band frequencies, 0 if L-band */
    OFPCBL_OSC     = 1 << 3,/* 1 if supporting OSC at 1510nm, 0 if not */
    OFPCBL_TLS     = 1 << 4,/* 1 if using a TLS, 0 if not */
    OFPCBL_FLAG     = 1 << 5 /* 1 if reporting wave-support, 0 if reporting used
waves
};
```

Circuit flow setup, modification and teardown (controller-> datapath)

```
struct ofp_cflow_mod {

  struct ofp_header header;
  uint16_t command;             /* one of OFPFC_* commands */
  uint16_t hard_timeout;        /* max time to connection tear down,
                                   if 0 then explicit tear-down required */
  uint8_t pad[4];
  struct ofp_connect connect;    /* 8B followed by variable length arrays */
  struct ofp_action_header actions[0]; /* variable number of actions */
};
```

Action structure for OFPAT_CKT_OUTPUT, which sends packets out of a circuit port

```
struct ofp_action_ckt_output {
  uint16_t type;                   /* OFPAT_CKT_OUTPUT */
  uint16_t len;                    /* Length is 24 */
  uint16_t adaptation;             /* Adaptation type - one of OFPCAT_* */
  uint16_t cport;                  /* Real or virtual OFPP_* ports */

  /* Define the circuit port characteristics if necessary */
  uint64_t wavelength;             /* use of the OFPCBL_* flags */
  uint32_t tsignal;                /* one of the  OFPTSG_* flags. Not valid if
```

```
                                       used with ofp_connect for TDM signals */
  uint16_t tstart;                  /* starting time slot. Not valid if used
                                       with of_connect for TDM signals */
  uint16_t tlcas_enable;            /* enable/disable LCAS */
};
```

Port Status: physical circuit port has changed in the datapath

```
struct ofp_cport_status {
    struct ofp_header header;
    uint8_t reason;          /* One of OFPPR_*. */
    uint8_t pad[7];          /* Align to 64-bits. */
    struct ofp_phy_cport desc;
};
```