TT Message Design

The TT message is designed to add real time transport features to OpenFlow standrad. By using TT message, users can download Time-Triggered flow scheduling tables to openflow switch by OpenFlow, and we keep the current OpenFlow services, messages and API, provide a non-intrusive service to achieve this goal.

***.1 Experimenter ID**

The Experimenter ID of this extension is:
ONF_EXPERIMENTER_ID = 0x4F4E4600

***.2 TT Flow Mod Messages**

This extension denes the following messages types:

```
/* Message types */
enum onf_exp_type {
    ONF_ET_TT_FLOW_MOD = 2400,
};
```

The message ONF_ET_TT_FLOW_MOD is used to modify a tt flow table, there is only one tt flow table in a specific switch:

```
/* Message structure for ONF_ET_TT_FLOW_MDOD. */
struct onf_tt_flow_mod {
    struct          ofp_header header;
    uint32_t        experimenter; /* ONF_EXPERIMENTER_ID. */
    uint32_t        exp_type; /* ONF_ET_TT_FLOW_MOD. */

    /* Command type */
    unint8_t        command; /* One of OFPFC_* */

    /* Entry field */
    uint8_t         port; /* The entry related port. */
    uint8_t         etype; /* Send entry or receive entry. */
    uint8_t         flow_id; /* The identify of a flow. */
```

```
    uint32_t        scheduled_time; /* The scheduled time that the
flow packet is received or sent. */
    uint32_t        period; /* The scheduling period. */
    uint32_t        buffer_id; /* Buffered packet to apply to. */
    uint32_t        pkt_size; /* The flow packet size. */

};
OFP_ASSERT(sizeof(struct onf_tt_flow_mod) == sizeof(struct
onf_exp_header) + 20);
```

The **`experimenter`** field is the Experimenter ID.

The **`exp_type`** field is set to the TT message types.

The **`command`** field specify the modify request type, the request types that are currently defined are:

```
enum ofp_tt_flow_mod_command {
    OFPFC_ADD = 0, /* New flow. */
    OFPFC_MODIFY = 1, /* Modify all matching flows. */
    OFPFC_DELETE = 3, /* Delete all matching flows. */
};
```

The **`port`** field is the port associated with flow table entry.

The **`etype`** field is the entry type to indicate whether the entry is send or receive entry. The TT entry type that are currently defined are:

```
/* TT entry type */
enum onf_tt_entry_type {
    ONF_TT_SEND = 0,
    ONF_TT_RECV = 1,
};
```

The **`flow_id`** field is the flow identifer, a number gived by the application. The flow identifer should be unique across the entire network.

The **`scheduled_time`** field

The **`period`** field is the time(ns) when this flow is received or sent for this port.

The **`buffer_id`** field refers to a packet bufferd at the switch.

The **`pkt_size`** field give the size of flow packet.

(work in process)

*.3 TT Errors

Where not otherwise specified below, implementations must use error codes defined in the OpenFlow specification to report issues arising from applying individual modifications.

Errors specific to this extension have the following structure:

```
/* Message structure for all errors. */
struct onf_error_msg {
    struct ofp_header header;
    uint16_t type; /* OFPET_EXPERIMENTER. */
    uint16_t exp_code; /* One of ONFERR_ET_* above. */
    uint32_t experimenter; /* ONF_EXPERIMENTER_ID. */
    uint8_t data[0]; /* Up to 64 bytes of failed request. */
};
OFP_ASSERT(sizeof(struct onf_error_msg) == sizeof(struct
ofp_error_experimenter_msg));
```

The **`type`** field must be set to OFPET_EXPERIMENTER.

The **`experimenter`** field is the Experimenter ID (see 3.1).

The **`data`** field contains a copy of the failed request message, truncated to 64 bytes.

The **`exp_type`** field is the experimenter error type. The currently defined experimenter error types are:

```
/* Error codes */
enum onf_error_exp_type {
    ONFERR_ET_UNKNOWN_PORT = 2400,
    ONFERR_ET_BAD_TYPE,
    ONFERR_ET_TIME_MATCH_FAILED,
    …
};
```