# FSL91030(M) chip

**SDK** API Documentation

Version: **V1.4**

Wuhan Feisiling Microelectronics Technology Co., Ltd.

**December 2022**

Table of contents



  

  

  

  

  

# 1 VLAN Management

## 1.1 int fsl_vlan_create

describe

Create a VLAN.

grammar

int fsl_vlan_create(int unit, fsl_vlan_t vid)

parameter

| | |
|---|---|
| unit | Device No |
| vid | vlan id |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.2 int fsl_vlan_port_add

describe

Add inbound and outbound port members for a VLAN.

grammar

int fsl_vlan_port_add(int unit, fsl_vlan_t vid, fsl_pbmp_t pbmp, fsl_pbmp_t ubmp,fsl_pbmp_t lbmp)

parameter

| | |
|---|---|
| unit | Device No |
| vid | vlan id |
| pbmp | Port members to be added |
| upmp | Untag port member to be added |
| lbmp | Trunk members to be added |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.3 int fsl_vlan_destroy

describe

Delete a specified VLAN.

grammar

int fsl_vlan_destroy(int unit, fsl_vlan_t vid)

parameter

unit Device No

vid vlan id

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.4 int fsl_vlan_destroy_all

describe

Delete all VLANs.

grammar

int fsl_vlan_destroy_all(int unit)

parameter

unit Device No

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.5 int fsl_vlan_port_remove

describe

Removes a port member from a specific vlan.

grammar

int fsl_vlan_port_remove(int unit, fsl_vlan_t vid, fsl_pbmp_t pbmp,fsl_pbmp_t lbmp)

parameter

| | |
|---|---|
| unit | Device No |
| vid | vlan id |
| pbmp | The port member to be deleted, high is a valid indication |
| lbmp | Trunk member to be deleted, high is a valid indication |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.6 int fsl_vlan_control_set

describe

Set the value of the corresponding attribute of vlan.

grammar

int fsl_vlan_control_set( int unit, fsl_vlan_control_t type,fsl_vlan_t vid, int value)

parameter

| | |
|---|---|
| unit | Device No |
| type | vlan attribute enumeration type |
| vid | vlan id |

value Property value to be set

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.7 int fsl_vlan_control_get

describe

Get the value of the corresponding attribute of vlan

grammar

int fsl_vlan_control_get( int unit, fsl_vlan_control_t type, fsl_vlan_t vid,int *value)

parameter

unit Device No

type vlan attribute enumeration type

vid vlan id

value vlan attribute value

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.8 int fsl_vlan_port_ingress_erps_set

describe

Inbound ring network protection settings.

grammar

int fsl_vlan_port_ingress_erps_set(int unit, fsl_vlan_t vid, fsl_pbmp_t pbmp,fsl_pbmp_t lbmp)

parameter

| | |
|---|---|
| unit | Device No |
| vid | vlan id |
| pbmp | Port members to be protected |
| lbmp | Trunk members to be protected |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.9 int fsl_vlan_port_egress_erps_set

describe

Outgoing ring network protection settings.

grammar

int fsl_vlan_port_egress_erps_set(int unit, fsl_vlan_t vid, fsl_pbmp_t pbmp,fsl_pbmp_t lbmp)

parameter

| | |
|---|---|
| unit | Device No |
| vid | vlan id |
| pbmp | Port members to be protected |
| lbmp | Trunk members to be protected |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.10 int fsl_vlan_port_default_action_set

describe

Create a default VLAN action for incoming ports

grammar

int fsl_vlan_port_default_action_set(int unit, int port, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| port | Inbound physical port |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.11 int fsl_vlan_port_default_action_get

describe

Get the default VLAN action for the incoming port.

grammar

int fsl_vlan_port_default_action_get(int unit, int port, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| port | Inbound physical port |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.12 int fsl_vlan_port_egress_default_action_set

describe

Create a default VLAN action for outbound ports.

grammar

int fsl_vlan_port_egress_default_action_set(int unit, int port, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| port | Outgoing physical port |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.13 int fsl_vlan_port_egress_default_action_get

describe

Get the default VLAN processing action of the outgoing port.

grammar

int fsl_vlan_port_egress_default_action_get(int unit, int port, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| port | Outgoing physical port |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.14 int fsl_vlan_port_default_action_delete

describe

Delete the default VLAN processing action of the inbound port.

grammar

int fsl_vlan_port_default_action_delete(int unit, int port)

parameter

| | |
|---|---|
| unit | Device No |
| port | Inbound physical port |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.15 int fsl_vlan_port_egress_default_action_delete

describe

Delete the default VLAN processing action of the outgoing port.

grammar

int fsl_vlan_port_egress_default_action_delete(int unit, int port)

parameter

| | |
|---|---|
| unit | Device No |
| port | Outgoing physical port |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.16 int fsl_vlan_port_protocol_action_add

describe

Add protocol vlan action to the port.

grammar

int fsl_vlan_port_protocol_action_add(int unit, int inIslag, int inPort,

int ethType, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| inPort | Ingress Port |
| ethType | Ethernet Type |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.17 int fsl_vlan_port_protocol_action_delete

describe

Delete the protocol vlan action of the port.

grammar

int fsl_vlan_port_protocol_action_delete(int unit, int inIslag, int inPort, int ethType)

parameter

| | |
|---|---|
| unit | Device No |
| inPort | Ingress Port |
| ethType | Ethernet Type |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.18 int fsl_vlan_port_protocol_action_get

describe

Get the protocol vlan action of the port.

grammar

int fsl_vlan_port_protocol_action_get(int unit, int inIslag, int inPort, int ethType, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| inPort | Ingress Port |
| ethType | Ethernet Type |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.19 int fsl_vlan_port_protocol_action_delete_all

describe

Delete the protocol vlan action of all ports.

grammar

int fsl_vlan_port_protocol_action_delete_all(int unit)

parameter

| | |
|---|---|
| unit | Device No |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.20 int fsl_vlan_translate_action_add

describe

Add inbound vlan-based vlan translation action.

grammar

int fsl_vlan_translate_action_add(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan, int inner_vlan, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| outer_vlan | Outer VLAN ID or tag |
| inner_vlan | Inner VLAN ID or tag |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.21 int fsl_vlan_translate_action_get

describe

Get inbound vlan-based vlan translation action.

grammar

int fsl_vlan_translate_action_get(int unit,int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan, int inner_vlan, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| outer_vlan | Outer VLAN ID or tag |
| inner_vlan | Inner VLAN ID or tag |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.22 int fsl_vlan_translate_action_delete

describe

Delete the inbound vlan-based vlan translation action.

grammar

int fsl_vlan_translate_action_delete(int unit,int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan, int inner_vlan)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |

| | |
|---|---|
| outer_vlan | Outer VLAN ID or tag |
| inner_vlan | Inner VLAN ID or tag |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.23 int fsl_vlan_translate_egress_action_add

describe

Add outbound vlan-based vlan translation action.

grammar

int fsl_vlan_translate_egress_action_add(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan, int inner_vlan, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| outer_vlan | Outer VLAN ID or tag |
| inner_vlan | Inner VLAN ID or tag |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.24 int fsl_vlan_translate_egress_action_delete

describe

Delete the outbound vlan-based vlan translation action.

grammar

int fsl_vlan_translate_egress_action_delete(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan, int inner_vlan)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| outer_vlan | Outer VLAN ID or tag |
| inner_vlan | Inner VLAN ID or tag |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.25 int fsl_vlan_translate_egress_action_get

describe

Get outbound vlan-based vlan translation action.

grammar

int fsl_vlan_translate_egress_action_get(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan, int inner_vlan, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |

| | |
|---|---|
| key_mode | Type of key for vlan translation |
| outer_vlan | Outer VLAN ID or tag |
| inner_vlan | Inner VLAN ID or tag |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 1.26 int fsl_vlan_translate_action_range_add

describe

Add inbound vlan translation action based on vlan range.

grammar

int fsl_vlan_translate_action_range_add(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan_lo, int outer_vlan_hi,int inner_vlan_lo, int inner_vlan_hi, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |

outer_vlan_lo outer VLAN ID minimum value

outer_vlan_hi outer vlan id maximum value

inner_vlan_lo inner vlan id minimum value

inner_vlan_hi inner vlan id maximum value

| | |
|---|---|
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.27 int fsl_vlan_translate_action_range_delete

describe

Delete the inbound VLAN translation action based on VLAN range.

grammar

int fsl_vlan_translate_action_range_delete(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan_lo, int outer_vlan_hi, int inner_vlan_lo, int inner_vlan_hi)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |

outer_vlan_lo outer VLAN ID minimum value

outer_vlan_hi outer vlan id maximum value

inner_vlan_lo inner vlan id minimum value

inner_vlan_hi inner vlan id maximum value

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.28 int fsl_vlan_translate_action_range_get

describe

Gets the inbound vlan translation action based on vlan range.

grammar

int fsl_vlan_translate_action_range_get(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, int outer_vlan_lo, int outer_vlan_hi, int inner_vlan_lo, int inner_vlan_hi, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| outer_vlan_lo | outer VLAN ID minimum value |
| outer_vlan_hi | outer vlan id maximum value |
| inner_vlan_lo | inner vlan id minimum value |
| inner_vlan_hi | inner vlan id maximum value |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.29 int fsl_vlan_mac_action_add

describe

Add inbound mac-based vlan translation action.

grammar

int fsl_vlan_mac_action_add(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, fsl_mac_t mac, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |

| | |
|---|---|
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| Mac | VLAN translation matching source MAC address |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.30 int fsl_vlan_mac_action_delete

describe

Delete the inbound mac-based vlan translation action.

grammar

int fsl_vlan_mac_action_delete(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, fsl_mac_t mac)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| Mac | VLAN translation matching source MAC address |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.31 int fsl_vlan_mac_action_get

describe

Get inbound mac-based vlan translation action.

grammar

int fsl_vlan_mac_action_get(int unit, int xlate, int gport, fsl_vlan_translate_key_t key_mode, fsl_mac_t mac, fsl_vlan_action_set_t *action)

parameter

| | |
|---|---|
| unit | Device No |
| xlate | Selection of xlate0 and xlate1 |
| gport | Virtual or physical port number |
| key_mode | Type of key for vlan translation |
| Mac | VLAN translation matching source MAC address |
| action | vlan tag action settings |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.32 int fsl_vlan_ip_action_add

describe

Added ip-based mac binding action.

grammar

int fsl_vlan_ip_action_add(int unit, fsl_vlan_ip_t *vlan_ip, fsl_mac_t mac, int gport)

parameter

| | |
|---|---|
| unit | Device No |
| vlan_ip | IP matching related configuration |
| Mac | Source MAC address bound to IP |

gport The virtual or physical port number bound to the IP

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.33 int fsl_vlan_ip_action_delete

describe

Delete the mac binding action based on ip.

grammar

int fsl_vlan_ip_action_delete(int unit, fsl_vlan_ip_t *vlan_ip)

parameter

unit Device No

vlan_ip IP matching related configuration

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.34 int fsl_vlan_ip_action_get

describe

Get mac binding action based on ip.

grammar

int fsl_vlan_ip_action_get(int unit, fsl_vlan_ip_t *vlan_ip, fsl_mac_t *mac, int *is_lag, int *lport)

parameter

unit Device No

vlan_ip IP matching related configuration

| | |
|---|---|
| Mac | Source MAC address bound to IP |
| is_lag | Port lag information bound to ip |
| lport | The physical port number bound to the IP |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.35 int fsl_mac_ip_bind_miss_action_set

describe

Set the global message processing action when mac or ip binding fails.

grammar

int fsl_mac_ip_bind_miss_action_set(int unit, int bypassen, int trapen, int dropen)

parameter

| | |
|---|---|
| unit | Device No |
| bypassen | Bypass Enable |
| trapen | Trap Enable |
| dropen | drop enable |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.36 int fsl_pdu_option_set

describe

Set l2pdu global processing action.

grammar

int fsl_pdu_option_set(int unit, uint64_t pdu_option)

parameter

| | |
|---|---|
| unit | Device No |
| pdu_option | Global pdu processing behavior configuration |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.37 int fsl_pdu_config_add

describe

Add pdu configuration entry.

grammar

int fsl_pdu_config_add(int unit, int index, fsl_pdu_config_t *pdu_cfg)

parameter

| | |
|---|---|
| unit | Device No |
| index | pdu configuration entry id |
| pdu_cfg | PDU content and mask configuration |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.38 int fsl_pdu_config_delete

describe

Deletes a pdu configuration entry.

grammar

int fsl_pdu_config_delete(int unit, int index)

parameter

| | |
|---|---|
| unit | Device No |
| index | pdu configuration entry id |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 1.39 int fsl_pdu_config_get

describe

Get the specific configuration of a pdu entry.

grammar

int fsl_pdu_config_get(int unit, int index, fsl_pdu_config_t *pdu_cfg)

parameter

| | |
|---|---|
| unit | Device No |
| index | pdu configuration entry id |
| pdu_cfg | PDU content and mask configuration |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 2 Policer Configuration

## 2.1 int fsl_policing_ctl_set

describe

Set the global configuration of policing.

grammar

int fsl_policing_ctl_set(int unit, int direct, fsl_policing_ctl_t *pol_ctl)

parameter

unit Device No

direct Policing direction, inbound and outbound

pol_ctl Global configuration data for policing

```
typedef struct fsl_policing_ctl_s {
    uint16_t    pktLenUsePkt; //Equivalent packet length when policing is based on packets, in bytes
    uint8_t     macroPktBytes; //macro mode
                                    // 0x0: Byte-based policing
                                    // 0x1: Packet-based policing
    uint8_t     flowPktBytes; //flow mode
                                    // 0x0: Byte-based policing
                                    // 0x1: Packet-based policing
    uint8_t     meterGran; // Control granularity, default is 0
    uint8_t     preambleLen; //Frame interval and preamble equivalent packet length, in bytes
} fsl_policing_ctl_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.2 int fsl_policing_ctl_get

describe

Get the global configuration of policing.

grammar

int fsl_policing_ctl_get(int unit, int direct, fsl_policing_ctl_t *pol_ctl)

parameter

| | |
|---|---|
| unit | Device No |
| direct | Policing direction, inbound and outbound |
| pol_ctl | Global configuration data for policing |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.3 int fsl_macro_policing_update_set

describe

Set up global update configuration based on port policing.

grammar

int fsl_macro_policing_update_set(int unit, int direct, fsl_policing_update_t *pol_upd)

parameter

| | |
|---|---|
| unit | Device No |
| direct | Policing direction, inbound and outbound |
| pol_upd | Policing updates configuration data |

typedef struct fsl_policing_update_s {

```
    uint8_t updEn;                  //Fill token bucket enable
    uint16_t updMaxIndex; //Update the maximum number of entries
    uint8_t timer0Num; //Token bucket refresh cycle parameters
    uint8_t timer1Num; //Token bucket refresh cycle parameters
    uint16_t timer0;                //Token bucket refresh cycle parameters
    uint16_t timer1;                //Token bucket refresh cycle parameters
} fsl_policing_update_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.4 int fsl_macro_policing_update_get

describe

Get the global update configuration based on port policing.

grammar

int fsl_macro_policing_update_get(int unit, int direct, fsl_policing_update_t *pol_upd)

parameter

| | |
|---|---|
| unit | Device No |
| direct | Policing direction, inbound and outbound |
| pol_upd | Policing updates configuration data |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.5 int fsl_flow_policing_update_set

describe

Sets the global update configuration for flow-based policing.

grammar

int fsl_flow_policing_update_set(int unit, int direct, fsl_policing_update_t *pol_upd)

parameter

unit Device No

direct Policing direction, inbound and outbound

pol_upd Policing updates configuration data

```
typedef struct fsl_policing_update_s {
    uint8_t updEn;                  //Fill token bucket enable
    uint16_t updMaxIndex; //Update the maximum number of entries
    uint8_t timer0Num; //Token bucket refresh cycle parameters
    uint8_t timer1Num; //Token bucket refresh cycle parameters
    uint16_t timer0;                //Token bucket refresh cycle parameters
    uint16_t timer1;                //Token bucket refresh cycle parameters
} fsl_policing_update_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.6 int fsl_flow_policing_update_get

describe

Gets the global update configuration based on flow policing.

grammar

int fsl_flow_policing_update_get(int unit, int direct, fsl_policing_update_t *pol_upd)

parameter

| | |
|---|---|
| unit | Device No |
| direct | Policing direction, inbound and outbound |
| pol_upd | Policing updates configuration data |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.7 int fsl_macro_policing_enable_set

describe

Set whether to enable port policing.

grammar

int fsl_macro_policing_enable_set(int unit, int direct, int pol_id, int pol_en)

parameter

| | |
|---|---|
| unit | Device No |
| direct | Policing direction, inbound and outbound |
| pol_id | Policing id, which is the physical port number |
| pol_en | Policing enable flag 0/1 |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.8 int fsl_macro_policer_create

describe

Create a port policing.

grammar

int fsl_macro_policer_create(int unit, int direct, int pol_id, fsl_policer_config_t *pol_cfg)

parameter

unit Device No

direct Policing direction, inbound and outbound

pol_id Policing id, which is the physical port number

pol_cfg Policing configuration data

```
typedef struct fsl_policer_config_s {
    fsl_policer_mode_t          mode; //fslPolicerModeTrTcm,                /* RFC 2698 */
                                      //fslPolicerModeTrTcmDs,              /* RFC 4115 */
                                      //fslPolicerModeSrTcm                 /* RFC 2697 */
    fsl_color_sense_t           colorSense; //Color blindness and color sensitivity
    uint8_t                     globalCFlag; //Global C bucket to E bucket coupling flag
    fsl_meter_sharing_mode_t sharingMode; //Sharing mode, 0: No sharing mode 1: FSL_MIN_ONLY,
                                          //2: FSL_MAX_ONLY, 3: FSL_MIN_MAX
    uint32_t                    cir; //c bucket token filling rate kb
    uint32_t                    cirMax; //c bucket maximum addition rate kb
    uint32_t                    cbs; //c barrel depth kb
    uint32_t                    eir; ///e bucket token filling rate kb
    uint32_t                    eirMax; //e bucket maximum addition rate kb
    uint32_t                    ebs; //e barrel depth

    uint8_t                     redPri; //New pri value for red message (color == 2'b00)
```

```
    uint8_t                 yellowPri; //New pri value for yellow message (color == 2'b01)
    uint8_t                 greenPri; //New pri value for green message (color == 2'b10)
    uint8_t                 rChangePri; //Red message update pri enable
    uint8_t                 yChangePri; // Yellow message updates pri enable
    uint8_t                 gChangePri; //Green message update pri enable
    uint8_t                 rChangeDrop; //Red message drop enable
    uint8_t                 yChangeDrop; // Yellow message drop enable
    uint8_t                 gChangeDrop; //Green message drop enable
} fsl_policer_config_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.9 int fsl_macro_policer_delete

describe

Delete port policing.

grammar

int fsl_macro_policer_delete(int unit, int direct, int pol_id)

parameter

| | |
|---|---|
| unit | Device No |
| direct | Policing direction, inbound and outbound |
| pol_id | Policing id, which is the physical port number |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.10 int fsl_flow_policer_create

describe

Create a flow policing.

grammar

int fsl_flow_policer_create(int unit, int direct, int pol_id, fsl_policer_config_t *pol_cfg)

parameter

unit Device No

direct Policing direction, inbound and outbound

pol_id Policing id (-1 means system assigned)

pol_cfg Policing configuration data

```
typedef struct fsl_policer_config_s {
    fsl_policer_mode_t      mode; //fslPolicerModeTrTcm,          /* RFC 2698 */
                                  //fslPolicerModeTrTcmDs,        /* RFC 4115 */
                                  //fslPolicerModeSrTcm           /* RFC 2697 */
    fsl_color_sense_t       colorSense; //Color blindness and color sensitivity
    uint8_t                 globalCFlag; //Global C bucket to E bucket coupling flag
    fsl_meter_sharing_mode_t sharingMode; //Sharing mode, 0: No sharing mode 1: FSL_MIN_ONLY,
                                          //2: FSL_MAX_ONLY, 3: FSL_MIN_MAX
    uint32_t                cir; //c bucket token filling rate kb
    uint32_t                cirMax; //c bucket maximum addition rate kb
    uint32_t                cbs; //c barrel depth kb
    uint32_t                eir; ///e bucket token filling rate kb
    uint32_t                eirMax; //e bucket maximum addition rate kb
```

```
    uint32_t                    ebs; //e barrel depth


    uint8_t                     redPri; //New pri value for red message (color == 2'b00)
    uint8_t                     yellowPri; //New pri value for yellow message (color == 2'b01)
    uint8_t                     greenPri; //New pri value for green message (color == 2'b10)
    uint8_t                     rChangePri; //Red message update pri enable
    uint8_t                     yChangePri; // Yellow message updates pri enable
    uint8_t                     gChangePri; //Green message update pri enable
    uint8_t                     rChangeDrop; //Red message drop enable
    uint8_t                     yChangeDrop; // Yellow message drop enable
    uint8_t                     gChangeDrop; //Green message drop enable
} fsl_policer_config_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 2.11 int fsl_flow_policer_delete

describe

Delete flow policing.

grammar

```
int fsl_flow_policer_delete(int unit, int direct, int pol_id)
```

parameter

| | |
|---|---|
| unit | Device No |
| direct | Policing direction, inbound and outbound |

pol_id Policing ID

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 3 Quality of Service

## 3.1 int fsl_qos_profile_create

describe

Create a priority mapping template.

grammar

int fsl_qos_profile_create(int unit, int qos_pro_index, fsl_qos_profile_t *qos_profile)

parameter

unit Device No

qos_pro_index mapping template index (-1 means system assigned)

qos_profile Template Configuration

```
typedef struct fsl_qos_profile_s {
    uint8_t     useDefault; //1: Use default priority
                            //0: Others
    uint8_t     useL2Info; //0x0: use L3 header information first
                           //0x1: Use L2 header information
    uint8_t     trustCtag; //0x0: Use STAG first
                           //0x1: Use CTAG
    uint8_t     phbPtr; //PHB table index pointer high 6 bits, available logical port number
    uint8_t     use_flag; //Default is 0, no configuration required
} fsl_qos_profile_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.2 int fsl_qos_profile_delete

describe

Delete a priority mapping template.

grammar

int fsl_qos_profile_delete(int unit, int qos_pro_index)

parameter

unit Device No

qos_pro_index mapping template serial number

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.3 int fsl_vlan_priority_map_set

describe

Set the mapping of VLAN priority to internal priority.

grammar

int fsl_vlan_priority_map_set(int unit, int qos_pro_index, int pkt_pri, int cfi, int internal_pri, fsl_color_t color)

parameter

unit Device No

qos_pro_index mapping template serial number

pkt_pri Message priority

cfi Message CFI

internal_pri Internal Priority

color color

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.4 int fsl_vlan_priority_map_get

describe

Get vlan priority mapping configuration.

grammar

int fsl_vlan_priority_map_get(int unit, int qos_pro_index, int pkt_pri, int cfi, int *internal_pri, fsl_color_t *color)

parameter

unit Device No

qos_pro_index mapping template serial number

pkt_pri Message priority

cfi Message CFI

internal_pri Internal Priority

color color

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.5 int fsl_dscp_map_set

describe

Sets the mapping of dscp to internal priorities.

grammar

int fsl_dscp_map_set(int unit, int qos_pro_index, int dscp, int internal_pri, fsl_color_t color)

  

parameter

unit Device No

qos_pro_index mapping template serial number

pkt_pri Message dscp

internal_pri Internal Priority

color color

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.6 int fsl_vlan_priority_map_get

describe

Gets the configuration of a dscp map.

grammar

int fsl_vlan_priority_map_get(int unit, int qos_pro_index, int pkt_pri, int cfi, int *internal_pri, fsl_color_t *color)

parameter

unit Device No

qos_pro_index mapping template serial number

pkt_pri Message dscp

internal_pri Internal Priority

color color

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.7 int fsl_pri_remark_enable_set

describe

Enable the port internal priority re-marking function.

grammar

int fsl_pri_remark_enable_set(int unit, int port, int rmk_en)

parameter

| | |
|---|---|
| unit | Device No |
| port | Physical port number |
| rmk_en | Re-marking enable flag 0/1 |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.8 int fsl_remark_profile_create

describe

Create a re-labeling template.

grammar

int fsl_remark_profile_create(int unit, int rmkPriPtr, fsl_rmk_info_t *rmk_info)

parameter

| | |
|---|---|
| unit | Device No |
| rmkPriPtr | Re-marking template sequence number |
| rmk_info | Re-marking configuration information |

typedef struct fsl_rmk_info_s {

uint8_t index; //Remapped index

```
    uint8_t scosRmkEn; //Remap scos enable
    uint8_t ccosRmkEn; //Remap ccos enable
    uint8_t     brgChgTos; //Remap tos enable
    uint8_t     onlyChgDscp; //Only modify dscp
    uint8_t     use_flag; //Default is 0, no configuration required
} fsl_rmk_info_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.9 int fsl_remark_profile_delete

describe

Delete the re-labeling template.

grammar

```
int fsl_remark_profile_delete(int unit, int rmkPriPtr)
```

parameter

| | |
|---|---|
| unit | Device No |
| rmkPriPtr | Re-marking template sequence number |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.10 int fsl_dscp_unmap_set

describe

Set the packet dscp heavy marking.

grammar

int fsl_dscp_unmap_set(int unit, int rmkPriPtr, int internal_pri, fsl_color_t color, int pkt_dscp)

parameter

| | |
|---|---|
| unit | Device No |
| internal_pri | Internal Priority |
| color | color |
| pkt_dscp | Message dscp |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.11 int fsl_vlanpri_unmap_set

describe

Set packet VLAN priority re-marking

grammar

int fsl_vlanpri_unmap_set(int unit, int rmkPriPtr, int internal_pri, fsl_color_t color, int cos, int cfi)

parameter

| | |
|---|---|
| unit | Device No |
| internal_pri | Internal Priority |
| color | color |
| cos | Packet vlan pri |
| cfi | Packet vlan cfi |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.12 int fsl_dscp_unmap_get

describe

Get the packet dscp re-marking configuration.

grammar

int fsl_dscp_unmap_get(int unit, int rmkPriPtr, int internal_pri, fsl_color_t color, int *pkt_dscp)

parameter

| | |
|---|---|
| unit | Device No |
| internal_pri | Internal Priority |
| color | color |
| pkt_dscp | Message dscp |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 3.13 int fsl_vlanpri_unmap_get

describe

Get the packet VLAN priority re-marking configuration.

grammar

int fsl_vlanpri_unmap_get(int unit, int rmkPriPtr, int internal_pri, fsl_color_t color, int *cos, int *cfi)

parameter

| | |
|---|---|
| unit | Device No |
| internal_pri | Internal Priority |
| color | color |

| | |
|---|---|
| cos | Packet vlan pri |
| cfi | Packet vlan cfi |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 4 Strom Control

## 4.1 int fsl_storm_control_enable_set

describe

Set whether to enable storm control.

grammar

int fsl_storm_control_enable_set(int unit, fsl_storm_control_mode_t mode, int arg, fsl_forward_type_t fwd_type, int enable)

parameter

| | |
|---|---|
| unit | Device No |
| mode | Storm control mode (3 types: system, port, forward id) |
| arg | Storm control id, corresponding to three modes |
| fwd_type | Forwarding type (unicast, multicast, broadcast) |
| enable | Enable flag |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 4.2 int fsl_storm_control_enable_get

describe

Get the enable flag of storm control.

grammar

int fsl_storm_control_enable_get(int unit, fsl_storm_control_mode_t mode, int arg, fsl_forward_type_t fwd_type, int *enable)

parameter

| | |
|---|---|
| unit | Device No |

| | |
|---|---|
| mode | Storm control mode (3 types: system, port, forward id) |
| arg | Storm control id, corresponding to three modes |
| fwd_type | Forwarding type (unicast, multicast, broadcast) |
| enable | Enable flag |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 4.3 int fsl_storm_control_global_set

describe

Set the global configuration of Storm Control.

grammar

int fsl_storm_control_global_set(int unit, uint8_t meter_gran, uint8_t preamble_len)

parameter

| | |
|---|---|
| unit | Device No |
| meter_gran | Control granularity |

preamble_len The equivalent packet length of the preamble and frame gap

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 4.4 int fsl_storm_control_global_get

describe

Get the global configuration of Storm Control.

grammar

int fsl_storm_control_global_get(int unit, uint8_t *meter_gran, uint8_t *preamble_len)

parameter

unit Device No

meter_gran Control granularity

preamble_len The equivalent packet length of the preamble and frame gap

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 4.5 int fsl_storm_control_update_set

describe

Sets the update configuration for Storm Control.

grammar

int fsl_storm_control_update_set(int unit, fsl_storm_control_mode_t mode, fsl_storm_ctl_global_t *global_ctl)

parameter

unit Device No

mode Storm Control Mode

global_ctl Updated configuration for Storm Control

```
typedef struct fsl_storm_ctl_global_s {
    uint32_t pktLenUsePkt; //Equivalent packet length when metering based on packets
    uint8_t updEn; //fill token bucket enable
    uint32_t maxUpdIdx; //The maximum index to fill the token bucket
    uint32_t delayInterval; //Filling token bucket period
} fsl_storm_ctl_global_t;
```

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 4.6 int fsl_storm_control_update_get

describe

Get updated configuration for Storm Control.

grammar

int fsl_storm_control_update_get(int unit, fsl_storm_control_mode_t mode, fsl_storm_ctl_global_t *global_ctl)

parameter

| | |
|---|---|
| unit | Device No |
| mode | Storm Control Mode |
| global_ctl | Updated configuration for Storm Control |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 4.7 int fsl_storm_control_set

describe

Set the storm control type, rate limit, and burst size.

grammar

int fsl_storm_control_set(int unit, fsl_storm_control_mode_t mode, int arg, fsl_forward_type_t fwd_type, fsl_storm_policing_type_t pol_type, uint32_t limit, uint32_t burst_size)

parameter

| | |
|---|---|
| unit | Device No |

  

| | |
|---|---|
| mode | Storm Control Mode |
| arg | Storm control id, corresponding to three modes |
| fwd_type | Forwarding Type |
| pol_type | Policing type (packets and bytes) |
| limit | Speed limit |
| burst_size | Burst Size |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 4.8 int fsl_storm_control_get

describe

Get the type, rate limit, and burst size of storm control.

grammar

int fsl_storm_control_get(int unit, fsl_storm_control_mode_t mode, int arg, fsl_forward_type_t fwd_type, fsl_storm_policing_type_t *pol_type, uint32_t *limit, uint32_t *burst_size)

parameter

| | |
|---|---|
| unit | Device No |
| mode | Storm Control Mode |
| arg | Storm control id, corresponding to three modes |
| fwd_type | Forwarding Type |
| pol_type | Policing type (packets and bytes) |
| limit | Speed limit |
| burst_size | Burst Size |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 5 Field Processor

## 5.1 int fsl_field_init

describe

fp is initialized and must be called first.

grammar

int fsl_field_init(int unit)

parameter

unit Device No

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 5.2 int fsl_field_detach

describe

Clear all fp related configurations, including software resources and hardware configurations.

grammar

int fsl_field_detach(int unit)

parameter

unit Device No

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 5.3 int fsl_field_group_create_mode_id

describe

Create a field group.

grammar

int fsl_field_group_create_mode_id(int unit, fsl_field_qset_t qset, int pri, fsl_field_group_mode_t mode,uint16_t entry_num,fsl_field_key_tp_t key_tp,fsl_field_group_t group)

parameter

unit Device No

qset Specifies which module to configure. Only supports fslFieldQualifyStageIngress, fslFieldQualifyStageEgress, fslFieldQualifyStageLookup, fslFieldQualifyStageLookupEgress

pri mode, entry_num, Key_tp are not used yet

group Group ID

return value

FSLRAL_E_NONE success

FSLRAL_E_INIT Uninitialized

FSLRAL_E_RESOURCE Hardware resource not available

FSLRAL_E_PARAM Parameter error

FSLRAL_E_EXISTS group ID or key tp already exists

FSLRAL_E_MEMORY Failed to allocate memory

FSLRAL_E_XXX Other Errors

# 5.4 int fsl_field_group_destroy

describe

Destroys a field group. Before calling this function, all entries under the group must be destroyed.

grammar

int fsl_field_group_destroy(int unit, fsl_field_group_t group)

parameter

  

| | |
|---|---|
| unit | Device No |
| group | Group ID |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | The group does not exist |
| FSLRAL_E_BUSY | There is entry in this group |
| FSLRAL_E_XXX | Other Errors |

# 5.5 int fsl_field_entry_create_id

describe

Create an entry.

grammar

int fsl_field_entry_create_id(int unit,fsl_field_group_t group,fsl_field_entry_t entry)

parameter

| | |
|---|---|
| unit | Device No |
| group | Group ID |
| entry | entry ID |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | Group does not exist |
| FSLRAL_E_EXISTS | entry ID already exists |
| FSLRAL_E_MEMORY | Failed to allocate memory |

FSLRAL_E_RESOURCE entry entry is full

## 5.6 int fsl_field_entry_install

describe

Install entry to the hardware table. Before installing entry, use fsl_field_qualify_xxx to add qualification. fsl_field_action_add adds actions.

grammar

int fsl_field_entry_install(int unit, fsl_field_entry_t entry)

parameter

unit Device No

entry entry ID

return value

FSLRAL_E_NONE success

FSLRAL_E_INIT Not initialized

FSLRAL_E_NOT_FOUND entry does not exist

FSLRAL_E_XXX Other Errors

## 5.7 int fsl_field_entry_remove

describe

Delete the entry from the hardware table.

grammar

int fsl_field_entry_remove(int unit, fsl_field_entry_t entry)

parameter

unit Device No

entry entry ID

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_XXX | Other Errors |

## 5.8 int fsl_field_entry_destroy

describe

Destroy entry. Destroy the software entry related resources. If the entry has been installed to the hardware table, the function will call The fsl_field_entry_remove interface clears the entry in the hardware table and releases resources.

grammar

int fsl_field_entry_destroy(int unit, fsl_field_entry_t entry)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_XXX | Other Errors |

## 5.9 int fsl_field_entry_destroy_all(int unit)

describe

Destroy all entries.

grammar

int fsl_field_entry_destroy_all(int unit)

parameter

| | |
|---|---|
| unit | Device No |

return value

| | |
|---|---|
| FSLRAL_E_NONE | Success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_XXX | Other Errors |

# 5.10 int fsl_field_action_add

describe

Add action to entry. Multiple actions can be added to one entry.

grammar

int fsl_field_action_add(int unit,fsl_field_entry_t entry,fsl_field_action_t action,uint32_t param0, uint32_t param1)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| action | Action Types |
| param0 | action value. Set to 0 when not used. |
| param1 | action value. Set to 0 when not used. |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_MEMORY | Failed to allocate memory |
| FSLRAL_E_UNAVAIL | The action type is not supported |
| FSLRAL_E_CONFIG | Action type conflicts |

| | |
|---|---|
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

# 5.11 int fsl_field_action_remove

describe

Delete action from entry.

grammar

int fsl_field_action_remove(int unit, fsl_field_entry_t entry,fsl_field_action_t action)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| action | Action Types |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Not initialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

# 5.12 int fsl_field_qualify_DstIp

describe

Add qualification to the entry to match the IPv4 address of the message.

grammar

int fsl_field_qualify_DstIp(int unit, fsl_field_entry_t entry,fsl_ip_t data,fsl_ip_t mask)

parameter

  

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| data | data |
| mask | |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 5.13 int fsl_field_qualify_DstIp6

describe

Add qualification to the entry to match the IPv6 address of the message.

grammar

int fsl_field_qualify_DstIp6(int unit, fsl_field_entry_t entry,fsl_ip6_t data,fsl_ip6_t mask)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| data | data |
| mask | |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |

| | |
|---|---|
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 5.14 int fsl_field_qualify_DstIpRange

describe

Add qualification to entry. Range matching message ipv4 address, supports 4 range configurations, and the configured range values cannot overlap.

grammar

int fsl_field_qualify_DstIpRange(int unit, fsl_field_entry_t entry,fsl_ip_t ipL, fsl_ip_t ipH)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| iP | Minimum ipv4 address |
| i | Maximum ipv4 address |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 5.15 int fsl_field_qualify_DstIp6Range

describe

Add qualification to entry. Range matching message ipv6 address, supports 2 range configurations, and the configured range values cannot overlap.

grammar

int fsl_field_qualify_DstIp6Range(int unit, fsl_field_entry_t entry,fsl_ip6_t ipL, fsl_ip6_t ipH)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| iP | Minimum ipv6 address |
| i | Maximum ipv6 address |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 5.16 int fsl_field_qualify_DstMac

describe

Add qualification to the entry to match the destination Mac address of the message.

grammar

int fsl_field_qualify_DstMac(int unit, fsl_field_entry_t entry,fsl_mac_t data,fsl_mac_t mask)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| data | data |
| mask | Mask |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

# 5.17 int fsl_field_qualify_OuterVlanId

describe

Add qualification to the entry to match the outer VLAN ID of the message.

grammar

int fsl_field_qualify_OuterVlanId(int unit, fsl_field_entry_t entry, fsl_vlan_t data,fsl_vlan_t mask)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| data | data |
| mask | Mask |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

# 5.18 int fsl_field_qualifier_delete

describe

Removes qualification from the specified entry.

grammar

int fsl_field_qualifier_delete(int unit, fsl_field_entry_t entry, fsl_field_qualify_t qual)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |
| qual | qualifier ID |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

# 5.19 int fsl_field_qualify_clear

describe

Remove all qualifications from the specified entry.

grammar

int fsl_field_qualify_clear(int unit, fsl_field_entry_t entry)

parameter

| | |
|---|---|
| unit | Device No |
| entry | entry ID |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |

  

| | |
|---|---|
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | entry does not exist |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 5.20 int fsl_field_qualify_data

describe

Add data qualification to entry to match the udf value.

grammar

int fsl_field_qualify_data(int unit,fsl_field_entry_t eid, int qual_id,uint8_t *data,uint8_t *mask, uint16_t length)

parameter

| | |
|---|---|
| unit | Device No |
| eid | entry ID |
| qual_id | Same as qual_id of fsl_field_data_qualifier_t structure |
| data | data |
| mask | Mask |
| length | The length of the matching data |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | qual ID not created |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 5.21 void fsl_field_data_qualifier_init

describe

Initialize the fsl_field_data_qualifier_t structure.

grammar

void fsl_field_data_qualifier_init(fsl_field_data_qualifier_t *data_qual)

parameter

data_qual structure pointer

return value

No return value.

# 5.22 int fsl_field_data_qualifier_create

describe

Create a data qualifier. Only software resources are created.

grammar

int fsl_field_data_qualifier_create(int unit, fsl_field_data_qualifier_t *data_qualifier)

parameter

unit Device No

data_qualifier structure pointer

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_EXISTS | qual ID exists |
| FSLRAL_E_FULL | The number of supported data qualifiers is full |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 5.23 int fsl_field_data_qualifier_destroy

describe

Destroys a data qualifier.

grammar

int fsl_field_data_qualifier_destroy(int unit, int qual_id)

parameter

| | |
|---|---|
| unit | Device No |
| qual_id | Qual ID |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND | qual ID not created |
| FSLRAL_E_XXX | Other Errors |

## 5.24 int fsl_field_data_qualifier_destroy_all

describe

Destroys all data qualifiers.

grammar

int fsl_field_data_qualifier_destroy_all(int unit)

parameter

| | |
|---|---|
| unit | Device No |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |

| | |
|---|---|
| FSLRAL_E_XXX | Other Errors |

# 5.25 void fsl_field_data_packet_format_t_init

describe

Initialize the _field_data_packet_format_t structure.

grammar

void fsl_field_data_packet_format_t_init (_field_data_packet_format_t *packet_format)

parameter

packet_format supports data qualifier data packet format

return value

No return value.

# 5.26 int fsl_field_data_qualifier_packet_format_add

describe

Write the software data corresponding to the qual id into the hardware table entry.

grammar

int fsl_field_data_qualifier_packet_format_add(int unit, int qual_id,_field_data_packet_format_t *packet_format)

parameter

| | |
|---|---|
| unit | Device No |
| qual_id | Qual ID |

packet_format Not used yet (cannot be NULL)

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |

FSLRAL_E_NOT_FOUND qual ID not created

  

| | |
|---|---|
| FSLRAL_E_XXX | Other Errors |

# 5.27 int fsl_field_data_qualifier_packet_format_delete

describe

Delete the hardware entry corresponding to the qual id.

grammar

int fsl_field_data_qualifier_packet_format_delete(int unit,int qual_id,_field_data_packet_format_t *packet_format)

parameter

| | |
|---|---|
| unit | Device No |
| qual_id | Qual ID |
| packet_format Not used yet (cannot be NULL) | |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | Uninitialized |
| FSLRAL_E_NOT_FOUND qual ID not created | |
| FSLRAL_E_XXX | Other Errors |

# 6 Trunking (Link Aggregation)

## 6.1 int fsl_trunk_init

describe

Trunk initialization. Must be called first.

grammar

int fsl_trunk_init(int unit)

parameter

unit Device No

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 6.2 int fsl_trunk_detach

describe

Clear all trunk related configurations, including software resources and hardware configurations.

grammar

int fsl_trunk_detach(int unit)

parameter

unit Device No

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 6.3 int fsl_trunk_create_id

describe

Create a trunk ID.

grammar

int fsl_trunk_create_id(int unit, fsl_trunk_t tid)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |

return value

| | |
|---|---|
| FSLRAL_E_NONE | Success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_EXISTS | trunk ID already exists |
| FSLRAL_E_XXX | Other Errors |

# 6.4 int fsl_trunk_set

describe

Specify the port, hash key and hash algorithm type in the trunk group. The default hash key is the source Mac address and the destination Mac address. If there are already member ports in the trunk group, the original member ports will be replaced.

grammar

int fsl_trunk_set(int unit, fsl_trunk_t tid, fsl_trunk_add_info_t *add_info)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |
| add_info | Structure pointer |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |

| | |
|---|---|
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_NOT_FOUND | trunk ID does not exist |
| FSLRAL_E_XXX | Other Errors |

## 6.5 int fsl_trunk_destroy

describe

Destroy the trunk group.

grammar

int fsl_trunk_destroy(int unit, fsl_trunk_t tid)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_NOT_FOUND | trunk ID does not exist |
| FSLRAL_E_XXX | Other Errors |

## 6.6 int fsl_trunk_get

describe

Get the port members, hash key and hash algorithm type in the trunk group.

grammar

int fsl_trunk_get(int unit, fsl_trunk_t tid,fsl_trunk_add_info_t *t_data)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |
| t_data | Save trunk information |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_XXX | Other Errors |

## 6.7 int fsl_trunk_psc_set

describe

Set the hash key for the trunk group.

grammar

int fsl_trunk_psc_set(int unit, fsl_trunk_t tid, int psc)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |
| psc | hash key |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_NOT_FOUND | trunk ID does not exist |

| | |
|---|---|
| FSLRAL_E_XXX | Other Errors |

## 6.8 int fsl_trunk_psc_get

describe

Get the hash key of the trunk group.

grammar

int fsl_trunk_psc_get(int unit, fsl_trunk_t tid, int *psc)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |
| psc | Obtained hash key |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |

FSLRAL_E_NOT_FOUND trunk ID does not exist

| | |
|---|---|
| FSLRAL_E_XXX | Other Errors |

## 6.9 int fsl_trunk_hash_alg_set

describe

Set the hash algorithm type for the trunk group.

grammar

int fsl_trunk_hash_alg_set(int unit,fsl_trunk_t tid,fsl_trunk_hash_alg_t alg)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |
| alg | Hash algorithm type |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_NOT_FOUND | trunk ID does not exist |
| FSLRAL_E_XXX | Other Errors |

## 6.10 int fsl_trunk_hash_alg_get

describe

Gets the hash algorithm type for the trunk group.

grammar

int fsl_trunk_hash_alg_get(int unit, fsl_trunk_t tid, int *alg)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |
| alg | The hash algorithm type to obtain |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_NOT_FOUND | trunk ID does not exist |

| | |
|---|---|
| FSLRAL_E_XXX | Other Errors |

# 6.11 int fsl_trunk_failover_set

describe

Set the trunk group port failure sharing to enable.

grammar

int fsl_trunk_failover_set(int unit, fsl_trunk_t tid, int able)

parameter

| | |
|---|---|
| unit | Device No |
| tid | trunk ID (0~7) |
| able | 1: Enable 0: Disable |

return value

| | |
|---|---|
| FSLRAL_E_NONE | success |
| FSLRAL_E_INIT | trunk not initialized |
| FSLRAL_E_PARAM | Parameter error |
| FSLRAL_E_NOT_FOUND | trunk ID does not exist |
| FSLRAL_E_EMPTY | The trunk group is not set |
| FSLRAL_E_XXX | Other Errors |

# 7 Layer 2 Address Management

## 7.1 int fsl_l2_addr_add

describe

Add a mac address.

grammar

int fsl_l2_addr_add(int unit, fsl_l2_addr_t *l2addr)

parameter

| | |
|---|---|
| unit | Device No |
| l2addr | l2 Address |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 7.2 int fsl_l2_addr_delete

describe

Delete a mac address.

grammar

int fsl_l2_addr_delete(int unit, fsl_mac_t mac, fsl_vlan_t vid,fslral_mem_t mem,uint32_t index)

parameter

| | |
|---|---|
| unit | Device No |
| Mac | mac address |
| vid | vlan id |
| mem | mac key table type |
| index | mac key table index |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 7.3 int fsl_l2_addr_get

describe

Get the type and index of a mac address table.

grammar

int fsl_l2_addr_get(int unit, fsl_mac_t mac, fsl_vlan_t vid,fsl_l2_addr_t *l2addr,fslral_mem_t *mem,uint32_t *index)

parameter

| | |
|---|---|
| unit | Device No |
| Mac | mac address |
| vid | vlan id |
| l2addr | l2 address, not used yet |
| mem | mac key table type |
| index | mac key table index |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 7.4 int fsl_l2_addr_delete_by_port

describe

Remove mac address based on port in mac address behavior table.

grammar

int fsl_l2_addr_delete_by_port(int unit, fsl_module_t mod, fsl_port_t port, uint32_t flags)

parameter

| | |
|---|---|
| unit | Device No |
| mod | Not used yet |
| port | The port number |
| flags | Not used yet |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 7.5 int fsl_l2_addr_delete_by_vlan

describe

Delete MAC address based on VLAN, delete all MAC addresses under the specified VLAN.

grammar

int fsl_l2_addr_delete_by_vlan(int unit, fsl_vlan_t vid, uint32_t flags)

parameter

| | |
|---|---|
| unit | Device No |
| vid | vlan id |
| flags | Not used yet |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 7.6 int fsl_l2_age_timer_get

describe

Get the normal aging time of the mac address.

grammar

int fsl_l2_age_timer_get(int unit, int *age_seconds)

parameter

unit Device No

age_seconds Normal aging time

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 7.7 int fsl_l2_age_timer_set

describe

Set the normal aging time of the MAC address.

grammar

int fsl_l2_age_timer_set(int unit, int age_seconds)

parameter

unit Device No

age_seconds Normal aging time

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 7.8 int fsl_l2_fast_age_enable_set

describe

Enable setting of mac address fast aging.

grammar

int fsl_l2_fast_age_enable_set(int unit, int value)

parameter

| | |
|---|---|
| unit | Device No |
| value | Fast aging enable switch value |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 8 Layer 2 Multicast Management

## 8.1 int fsl_mcast_create

describe

Create a multicast group id.

grammar

int fsl_mcast_create(int unit, int group_id, fsl_pbmp_t pbmp)

parameter

| | |
|---|---|
| unit | Device No |
| group_id | Multicast group id |
| pbmp | Multicast group port members |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 8.2 int fsl_mcast_delete

describe

Delete a multicast group ID.

grammar

int fsl_mcast_delete(int unit, int group_id)

parameter

| | |
|---|---|
| unit | Device No |
| group_id | Multicast group id |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 8.3 int fsl_mcast_bitmap_del

describe

Delete the port member corresponding to the multicast group id.

grammar

int fsl_mcast_bitmap_del(int unit, uint16_t group_id, fsl_pbmp_t pbmp)

parameter

| | |
|---|---|
| unit | Device No |
| group_id | Multicast group id |
| pbmp | The multicast group member to be deleted, high is a valid indication |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 8.4 int fsl_mcast_addr_add

describe

Add a multicast mac address.

grammar

int fsl_mcast_addr_add(int unit, fsl_mcast_addr_t *mcaddr)

parameter

| | |
|---|---|
| unit | Device No |
| mcaddr | Multicast address information structure |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 8.5 int fsl_mcast_addr_remove

describe

Delete a multicast mac address.

grammar

int fsl_mcast_addr_remove(int unit, sal_mac_addr_t mac, fsl_vlan_t vid)

parameter

| | |
|---|---|
| unit | Device No |
| Mac | mac address |
| vid | vlan id |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 9 Port Configuration

## 9.1 int fsl_port_control_set

describe

Set the value of the corresponding attribute of the port.

grammar

int fsl_port_control_set(int unit, fsl_port_t port, fsl_port_control_t type, int value)

parameter

| | |
|---|---|
| unit | Device No |
| type | port property enumeration type |
| port | The port number |
| value | port attribute value |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.2 int fsl_port_control_get

describe

Get the value of the attribute corresponding to the port.

grammar

int fsl_port_control_get(int unit, fsl_port_t port, fsl_port_control_t type, int *value)

parameter

| | |
|---|---|
| unit | Device No |
| type | port property enumeration type |
| port | The port number |

value port attribute value

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.3 int fsl_port_tpid_init

describe

Initialize and obtain the tpid usage type of each port, initialize the variables, and be called first when tpid is processed.

grammar

int fsl_port_tpid_init(int unit)

parameter

unit Device No

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.4 int fsl_port_tpid_add

describe

The inbound port tpid is added.

grammar

int fsl_port_tpid_add(int unit,fsl_port_t port,uint16_t tpid)

parameter

unit Device No

port The port number

tpid tpid type value (default is 0x88A8)

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.5 int fsl_port_tpid_delete

describe

The inbound port tpid is deleted.

grammar

int fsl_port_tpid_delete(int unit,fsl_port_t port,uint16_t tpid)

parameter

| | |
|---|---|
| unit | Device No |
| port | The port number |
| tpid | Tpid type value |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.6 int fsl_port_tpid_set

describe

The outbound port tpid is added.

grammar

int fsl_port_tpid_set(int unit,fsl_port_t port,uint16_t tpid)

parameter

| | |
|---|---|
| unit | Device No |
| port | The port number |

tpid Tpid type value

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.7 int fsl_ingress_port_stp_set

describe

Set the inbound port stp status.

grammar

int fsl_ingress_port_stp_set(int unit, fsl_port_t gport, fsl_vlan_t vid, int stp_state)

parameter

unit Device No

gport Port number (divided into normal port and trunk port)

stp_state Port STP status (four types in total) FSL_STP_DISABLE = 0x0, FSL_STP_BLOCKING = 0x1, FSL_STP_LEARNING = 0x2, FSL_STP_FORWARDING = 0x3

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.8 int fsl_egress_port_stp_set

describe

Set the outbound direction port stp status.

grammar

int fsl_egress_port_stp_set(int unit, fsl_port_t gport, fsl_vlan_t vid, int stp_state)

parameter

unit Device No

  

gport Port number (divided into normal port and trunk port)

stp_state Port STP status (four types in total) FSL_STP_DISABLE = 0x0, FSL_STP_BLOCKING = 0x1, FSL_STP_LEARNING = 0x2, FSL_STP_FORWARDING = 0x3

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.9 int fsl_egress_port_stp_get

describe

Get the outbound stp status.

grammar

int fsl_egress_port_stp_get(int unit, fsl_port_t gport, fsl_vlan_t vid, int *stp_state)

parameter

unit Device No

gport Port number (divided into normal port and trunk port)

stp_state Port STP status (four types in total) FSL_STP_DISABLE = 0x0, FSL_STP_BLOCKING = 0x1, FSL_STP_LEARNING = 0x2, FSL_STP_FORWARDING = 0x3

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.10 int fsl_ingress_stp_erps_enable_set

describe

Set the stpchk and erpschk enable status.

grammar

int fsl_ingress_stp_erps_enable_set(int unit, fsl_port_t gport, fsl_port_control_t chk_type, int value)

  

parameter

unit Device No

gport Port number (divided into normal port and trunk port)

chk_type stp check erps check type (four types) fslPortControlerpsLkpEn,

fslPortControlStpChEn, fslPortControlEerpsLkEn, fslPortControlOutStpChkEn

value Enable status value

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

## 9.11 int fsl_ingress_stp_erps_enable_get

describe

Get the enable status of stpchk and erpschk.

grammar

int fsl_ingress_stp_erps_enable_get(int unit ,fsl_port_t gport, fsl_port_control_t chk_type, int *value)

parameter

unit Device No

gport Port number (divided into normal port and trunk port)

chk_type stp check erps check type (four types) fslPortControlerpsLkpEn,

fslPortControlStpChEn, fslPortControlEerpsLkEn, fslPortControlOutStpChkEn

value Enable status value

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 9.12 int fslsoc_stat_get

describe

Gets the count of different types of ports.

grammar

int fslsoc_stat_get(int unit, fslsoc_port_t port, fslsoc_stat_val_t type, uint64_t *val)

parameter

| | |
|---|---|
| unit | Device No |
| port | The port number |
| type | Different counting types |
| value | Count value |

return value

FSLRAL_E_NONE Success

FSLRAL_E_XXX Error

# 10 Set filter, send and receive packets

Note: This section is only applicable to FSL91030M.

# 10.1 fsl_rx_filter_create

describe

Create a filter

grammar

int fsl_rx_filter_create(uint8_t pkt_data_offset,uint8_t pkt_data_size,uint8_t *data,uint8_t *mask,
uint8_t priority,char *desc,uint8_t desc_size, uint16_t desc_type)

parameter

pkt_data_offset: message offset.

pkt_data_size: The length of the message field to be filtered, calculated from the message offset pkt_data_offset. The maximum length is KCOM_FILTER_BYTES_MAX (256 bytes)

data: The message field to be filtered by the filter, its length is equal to pkt_data_size.

mask: The mask used for the message field to be filtered. Its length is equal to pkt_data_size

priority: priority. The lower the value, the higher the priority.

desc: filter name.

desc_size: The length of the filter name, the maximum value is KCOM_FILTER_DESC_MAX (32 bytes)

desc_type: filter behavior, KCOM_DEST_T_NULL (no processing), KCOM_DEST_T_NETIF (upload Protocol stack), KCOM_DEST_T_API (upload application layer).

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

# 10.2 fsl_rx_filter_list

describe

Query filter id and quantity

  

grammar

int fsl_rx_filter_list(uint8_t *filter_ids,uint32_t *filter_num)

parameter

filter_ids: the filter id list found

filter_num: The number of filters found

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

## 10.3 fsl_rx_filter_get

describe

Get filter information based on filter id

grammar

int fsl_rx_filter_get(uint8_t id,kcom_filter_t *filter)

parameter

id: The id of the filter to be queried

filter: The information of the filter queried

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

## 10.4 fsl_rx_filter_destroy

describe

Delete a filter by filter id

grammar

  

int fsl_rx_filter_destroy(uint8_t id)

parameter

id: the filter id to be deleted

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

## 10.5 prepare_pkt

describe

Set the header of the packet to be sent.

grammar

int prepare_pkt(fsl_pkt_t *pkt)

parameter

pkt Packet to be sent

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

## 10.6 fsl_common_tx

describe

Package

grammar

int fsl_common_tx(int unit, fsl_pkt_t *pkt, void *cookie)

parameter

unit device number

pkt Packet to be sent

Cookie reserved, not used

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

## 10.7 fsl_common_rx_register

describe

Register the packet receiving function

grammar

int fsl_common_rx_register(int unit, const char *name, fh_rx_cb_f callback, uint8_t priority, void *cookie, uint32_t flags)

parameter

unit device number

name The name of the packet receiving processing function

callback: packet receiving function

priority The priority of the packet receiving processing function

Cookie reserved, not used

flags reserved, not used

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

## 10.8 fsl_common_rx_unregister

describe

Unregister the packet receiving function

  

grammar

int fsl_common_rx_unregister(int unit, fh_rx_cb_f callback, uint8_t priority)

parameter

callback: packet receiving function

priority The priority of the packet receiving processing function

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

# 10.9 fsl_common_rx_start

describe

Start receiving packets. Initialize and create receiving and processing threads.

grammar

int fsl_common_rx_start(int unit)

parameter

unit device number

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

# 10.10 fsl_common_rx_shutdown

describe

Stop receiving packets. Deregister the packet receiving thread and packet processing thread.

grammar

int fsl_common_rx_shutdown(int unit)

parameter

unit device number

return value

FSL_ERR_OK Success

FSL_ERR_XXX Error

## **11Revision** Information

| Revision time | Version | describe |
|---|---|---|
| 2021.4.27 | V1.0 | initial version. |
| 2021.12.23 | V1.4 | Content optimization. |