Vlan概要设计

|  |  |  |
| --- | --- | --- |
| 文件状态：  [√] 草稿  [ ] 正式发布  [ ] 正在修改 | 文件标识： |  |
| 当前版本： | 1.0 |
| 作 者： | 严林钢 |
| 审 核： |  |
| 完成日期： |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文档更新记录 | | | | |
| 版本/状态 | 作者 | 审核 | 起止日期 | 备注 |
| 1.0 | 严林钢 |  | 2016/1/22 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

目录

[1 文档说明 4](#_Toc440371302)

[1.1 编写目的 4](#_Toc440371303)

[1.2 阅读建议 4](#_Toc440371304)

[1.3 参考资料 4](#_Toc440371305)

[1.4 缩写术语 4](#_Toc440371306)

[2 综合描述 5](#_Toc440371307)

[2.1 需求描述 5](#_Toc440371308)

[2.2 设计约束 5](#_Toc440371309)

[2.2.1 运行环境 5](#_Toc440371310)

[2.2.2 运行形式 5](#_Toc440371311)

[2.3 结构设计 5](#_Toc440371312)

[2.3.1 模块在系统中的位置 5](#_Toc440371313)

[2.3.2 模块内部结构 5](#_Toc440371314)

[2.3.3 Vlan模块与其他模块的通道 5](#_Toc440371315)

[3 功能设计 7](#_Toc440371316)

[3.1 VLAN管理 7](#_Toc440371317)

[3.1.1 功能说明 7](#_Toc440371318)

[3.1.2 数据结构 7](#_Toc440371319)

[3.1.3 流程说明 7](#_Toc440371320)

[3.1.4 对外接口说明 7](#_Toc440371321)

[3.2 VLAN-端口管理 8](#_Toc440371322)

[3.2.1 功能说明 8](#_Toc440371323)

[3.2.2 数据结构 8](#_Toc440371324)

[3.2.3 流程说明 8](#_Toc440371325)

[3.2.4 对外接口 8](#_Toc440371326)

[3.3 VLAN命令行 8](#_Toc440371327)

[3.3.1 功能说明 8](#_Toc440371328)

[3.3.2 数据结构 8](#_Toc440371329)

[3.3.3 流程说明 8](#_Toc440371330)

[3.3.4 对外接口 8](#_Toc440371331)

[4 调试性设计 9](#_Toc440371332)

[5 遗留问题 10](#_Toc440371333)

[6 附录说明 11](#_Toc440371334)

[6.1 附加文档 11](#_Toc440371335)

# 文档说明

## 编写目的

## 阅读建议

## 参考资料

## 缩写术语

|  |  |  |
| --- | --- | --- |
| **缩写术语** | **英文** | **中文** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 综合描述

## 需求描述

**Vlan 配置**

1. 创建/删除vlan
2. 查看vlan信息

**Vlan-Port配置**

1. 端口类型的设置（access/trunk/hybrid）
2. 端口的Pvid号设置

## 设计约束

### 运行环境

* 硬件要求

CPU：Mips-ar9344、9341

#### 软件环境

|  |  |
| --- | --- |
| OS\Kernel | Linux 2.6.31 |
| 第三方软件 | Bsp/Ethernet/athr\_gmac.ko |

### 运行形式

|  |  |  |
| --- | --- | --- |
|  | 用户态 | 内核态 |
| 运行形式 | 动态链接库：libvlan.so  可执行程序：vlan | 无 |

## 结构设计

### 模块在系统中的位置

用户层和驱动层，在程序运行时，被操作系统载入内存中



### 模块内部结构



模块内部包括以下结构：

* Vlan管理
* Vlan-Port管理
* Vlan命令行

vlan模块分为3个部分

* **VLAN管理**

Vlan 的管理主要内容:

1>Vlan 配置

* **VLAN-Port管理**

Vlan-port管理主要内容：

1>port tag/untag配置

2>port pvid配置

* **VLAN命令行**

vlan，vlan-port的配置命令的解析

### Vlan模块与其他模块的通道

#### Vlan模块与以太网驱动的通道

ioctl调用

#### Vlan模块与Linux device的通道

socket通道、ioctl调用

#### Vlan模块与UI的通道

库函数调用s

# 功能设计

## VLAN管理

### 功能说明

1>Vlan的创建/删除/查询

### 数据结构

### 流程说明

#### 添加vlan

**流程描述**

对cmd解析后，查看vlan端口是否存在，不存在则创建vlan端口；若存在，查看vlan子端口是否存在，若不存在创建vlan子端口；若子端口存在，将子端口放在vlan端口下，返回数据和执行状态

**流程图**



#### 删除vlan

**流程描述**

对cmd解析后，查看vlan端口是否存在，若不存在直接返回，若存在查看子端口是否存在，若不存在，删除vlan端口返回；若存在查看子端口是否在vlan端口下，若在，将其从vlan端口下移除，再删除子端口；若不在vlan端口下，直接删除子端口，再删除vlan端口，返回数据和执行状态。

**流程图**



#### 查看vlan

**流程描述**

对cmd解析后，查看有哪些vlan端口存在，返回数据和执行状态。

**流程图**



### 函数接口说明

#### 添加vlan

* 函数原型

AP\_ERROR\_CODE\_E vlan\_add(int vlan\_id) //创建vlan端口

* 参数说明

vlan\_id：所需要添加的vlan-id

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 失败

* 功能描述

创建vlan 端口

#### 删除vlan

* 函数原型

AP\_ERROR\_CODE\_E vlan\_del(int vlan\_id) // 删除vlan端口

* 参数说明

vlan\_id：所需要删除的vlan-id

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 失败

* 功能描述

删除vlan

#### 查看vlan

* 函数原型

AP\_ERROR\_CODE\_E vlan\_get(int vlan\_id)

* 参数说明
* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 失败。

* 功能描述

查看当前创建的vlan

## VLAN-Port管理

### 功能说明

1. port tag/untag增、删、清除、改、查（add/del/ clean/set/get）；
2. port pvid改、查（set/get）。

### 数据结构

### 流程说明

#### 设置pvid

**流程描述**

对cmd解析后，查看vlan端口是否存在，若不存在直接返回，若存在，查看port 是否存在，若不存在直接返回；若存在，设置pvid，返回数据和执行状态。

**流程图**



#### 查询Pvid

**流程描述**

对cmd解析后，查看vlan端口是否存在，若不存在直接返回，若存在，查看port 是否存在，若不存在直接返回；若存在，查看pvid，返回数据和执行状态。

**流程图**



#### 添加tag

**流程描述**

对cmd解析后，查看vlan是否存在，若不存在，直接返回失败；若存在，再配置交换芯片，添加tag，返回数据和执行状态。

**流程图**



#### 删除tag

**流程描述**

对cmd解析后，查看vlan端口是否存在，若不存在，直接返回失败；若存在，将其子端口从vlan端口下移除，配置交换芯片，删除tag，返回数据和执行状态。

**流程图**



#### 清除tag

**流程描述**

对cmd解析后，查看vlan是否存在，若不存在，直接返回失败；若存在，再配置交换芯片，清除tag，返回数据和执行状态。

**流程图**



#### 修改tag

**流程描述**

对cmd解析后，调用vlan\_tag\_clean接口清除tag，调用vlan\_tag\_add接口添加tag，返回数据和执行状态。

**流程图**



#### 查看tag

**流程描述**

对cmd解析后，查看vlan端口是否存在，若不存在，直接返回失败；若存在，则查看tag，返回数据和执行状态。

**流程图**



#### 添加untag

**流程描述**

对cmd解析后，查看vlan是否存在，若不存在，直接返回失败；若存在，配置交换芯片，向寄存器写值，添加untag，返回数据和执行状态。

**流程图**



#### 删除untag

**流程描述**

对cmd解析后，查看vlan端口是否存在，若不存在，直接返回失败；若存在，将其子端口从vlan端口下移除，配置交换芯片，删除untag，返回数据和执行状态。

**流程图**



#### 清除untag

**流程描述**

对cmd解析后，查看vlan是否存在，若不存在，直接返回失败；若存在，再配置交换芯片，清除untag，返回数据和执行状态。

**流程图**



#### 修改untag

**流程描述**

对cmd解析后，调用vlan\_ untag \_clean接口清除untag，调用vlan\_ untag \_add接口添加untag，返回数据和执行状态。

**流程图**



#### 查看untag

**流程描述**

对cmd解析后，查看vlan端口是否存在，若不存在，直接返回失败；若存在，则查看untag，返回数据和执行状态。

**流程图**



### 对外接口

#### 设置Pvid

* 函数原型
* AP\_ERROR\_CODE\_E port\_pvid\_set(int port\_no，int pvid)
* 参数说明

pvid：需要设置的pvid

port\_no：需要设置pvid的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

给指定端口设置pvid

#### 查看Pvid

* 函数原型
* AP\_ERROR\_CODE\_E port\_pvid\_get(int port\_no)
* 参数说明

port\_no：需要查询pvid的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

查看指定端口的pvid

#### 添加tag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_tag\_add(int port\_no，int vlan\_id)
* 参数说明

port\_no：需要添加tag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

端口添加tag

#### 删除tag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_tag\_del(int port\_no，int vlan\_id)
* 参数说明

port\_no：需要删除tag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

端口删除tag

#### 清除tag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_tag\_clean (int port\_no)
* 参数说明

port\_no：需要清除tag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

清除端口tag

#### 修改tag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_tag\_set(int port\_no，int vlan\_id)
* 参数说明

port\_no：需要修改tag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

修改端口tag

#### 查看tag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_tag\_get(int port\_no)
* 参数说明

port\_no：需要查看tag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

查看端口tag

#### 添加untag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_untag\_add(int port\_no，int vlan\_id)
* 参数说明

port\_no：需要添加untag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

端口添加untag

#### 删除untag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_untag\_del(int port\_no，int vlan\_id)
* 参数说明

port\_no：需要删除untag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

端口删除untag

#### 清除untag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_untag\_clean (int port\_no)
* 参数说明

port\_no：需要清除untag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

清除端口untag

#### 修改untag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_untag\_set(int port\_no，int vlan\_id)
* 参数说明

port\_no：需要修改untag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

修改端口untag

#### 查看untag

* 函数原型
* AP\_ERROR\_CODE\_E vlan\_untag\_get(int port\_no)
* 参数说明

port\_no：需要查看untag的端口

* 返回值

AP\_E\_NONE 成功;

AP\_E\_PARAM 参数为NULL

AP\_E\_DEFAULT\_FAIL 配置失败

* 功能描述

查看端口untag

## VLAN命令行

### 功能说明

实现以下配置命令：

vlan add/del <vlan\_id> //创建/删除vlan

vlan port <port\_no> <link\_type> //设置port为那种类型（access/trunk/hybrid）

vlan port <port\_no> tag/ untag <vlan\_id> //向vlan里添加/删除port

vlan port <port\_no> pvid <pvid\_number> //设置pvid

vlan show //显示vlan（vlan\_id，vlan下有哪些port，port的pvid，port的link-type）

### 数据结构

Nvarm里面需要数据：

对于Port  ：

#define PORT\_LINK\_TYPE “port\_link\_type”

//每一个port对应的link\_type，例如：0；1；2（0：port0的link\_type是access；1：port1的link\_type是trunk；2：port2的link\_type是hybrid）

#define PORT\_PVID             “port\_pvid”

//每一个port对应的pvid，例如：100；200（100：port0的pvid是100；200：port1的pvid是200）

#define PORT\_UNTAG “port\_untag”

//每一个port对应vlan的模式是untag，例如：100,200；300,400；（对于port0针对vlan100，vlan200 的模式是untag；对于port1针对vlan300，vlan400的模式是untag）

#define PORT\_TAG “port\_tag”

//每一个port对应vlan的模式是tag，例如：100,200；300,400；（对于port0针对vlan100，vlan200 的模式是tag；对于port1针对vlan300，vlan400的模式是tag）

对于vlan：

         #define VLAN\_ID “vlan\_id” //表示 创建的vlan 例：vlan\_id = 100，200，…

对于port的结构体：

typedef enum{

                   NULL,

                 ACCESS,

                   TRUNK,

                   HYBRID,

}VLAN\_PORT\_TYPE;

Typedef struct{

UINT32 port\_no;

UINT32 pvid;

VLAN\_PORT\_TYPE port\_link\_type; //端口的类型

list \*untag //port对应vlan的模式是untag

list \*tag //port对应vlan的模式是tag

}PORT\_DATA\_T;

#### 命令 vlan add <vlan\_id> 实现

**流程描述**

1>调用AP\_ERROR\_CODE\_E vlan\_add(int vlan\_id)接口创建的vlan，

2>判断调用是否成功，若成功则向nvarm里写入vlan\_id值，若失败则直接返回

**流程图**



#### 命令vlan del <vlan\_id> 实现

**流程描述**

1. 调用AP\_ERROR\_CODE\_E vlan\_del(int vlan\_id)接口创建的vlan，
2. 判断调用是否成功，若成功则向nvarm里清除vlan\_id值，若失败则直接返回失败。

**流程图**



#### vlan port <port\_no> <link\_type> 实现

##### 设置Access口

**流程描述**

1. 读nvarm里面的pvid，查看是否有pvid，若没有调用port\_pvid\_set接口设置pvid==1
2. 若有pvid值，调用port\_pvid\_set接口设置pvid是否成功，不成功则返回失败
3. 若成功，在调用vlan\_tag\_set设置tag==pvid是否成功，若不成功，返回失败，
4. 若成功，再调用vlan\_untag\_set设置untag==pvid是否成功，若不成功，返回失败，
5. 若成功向nvarm里写入相应的值，返回数据和执行状态。

**流程图**



##### 设置trunk口

**流程描述**

1. 读nvarm里面的pvid，查看是否有pvid，若没有调用port\_pvid\_set接口设置pvid==1
2. 若有pvid值，调用port\_pvid\_set接口设置pvid是否成功，不成功则返回失败
3. 若成功，在调用vlan\_tag\_set设置tag==pvid是否成功，若不成功，返回失败，
4. 若成功，再调用vlan\_untag\_set设置untag==pvid是否成功，若不成功，返回失败，
5. 若成功向nvarm里写入相应的值，返回数据和执行状态。

**流程图**



注：与access相比，配置成trunk的时候，可以调用vlan\_tag\_add接口，但不能调用vlan\_untag\_add接口。

##### 设置hybrid口

**流程描述**

1. 读nvarm里面的pvid，查看是否有pvid，若没有调用port\_pvid\_set接口设置pvid==1
2. 若有pvid值，调用port\_pvid\_set接口设置pvid是否成功，不成功则返回失败
3. 若成功，在调用vlan\_tag\_set设置tag==pvid是否成功，若不成功，返回失败，
4. 若成功，再调用vlan\_untag\_set设置untag==pvid是否成功，若不成功，返回失败，
5. 若成功向nvarm里写入相应的值，返回数据和执行状态。

**流程图**



注：与access相比，配置成hybrid时候，可以调用vlan\_tag\_add接口，也可以调用vlan\_untag\_add接口。

#### 命令vlan port <port\_no> tag <vlan\_id> 实现

**流程描述**

1. 查看port是否存在，若不存在直接返回失败
2. 若存在，调用AP\_ERROR\_CODE\_Evlan\_tag\_add(int port\_no ,int vlan\_id)接口，向tag表中添加vlan
3. 调用vlan\_untag\_add(int port\_no ,int vlan\_id) 向untag表中添加vlan
4. 向nvarm里写入相关的值，返回数据和执行状态

**流程图**



#### 命令vlan port <port\_no> untag<vlan\_id> 实现

**流程描述**

1. 是否能从nvarm里读取相关的值，若不能直接返回失败
2. 若可以，调用AP\_ERROR\_CODE\_Evlan\_tag\_del(int port\_no ,int vlan\_id)接口，向tag表中删除vlan
3. 调用vlan\_untag\_del(int port\_no ,int vlan\_id) 向untag表中删除vlan
4. 向nvarm里写入相关的值，返回数据和执行状态

**流程图**



#### 命令vlan port <port\_no> pvid < pvid\_number> 实现

**流程描述**

1. 查看port是否存在，若不存在直接返回失败，
2. 若存在，调用port\_pvid\_set(int port\_no，int pvid)接口设置pvid
3. 调用vlan\_untag\_set(int port\_no,int vlan)接口修改untag
4. 向nvarm里写入port\_pvid值，返回数据和执行状态

**流程图**



#### 命令vlan show 实现

**流程描述**

读取nvarm里面的值，显示相关的值

**流程图**

### 对外接口

# 调试性设计

日志等调试方式在此描述

# 遗留问题

# 附录说明

## 附加文档