# Formula编码规范

## 文件

### 目录命名

目录名称对应芯片的大模块名称，参照《Formula TM 内部互联接口》及《Formula Switch 内部互联接口》 PPT中定义的英文名称命名的小写形式，如NIF - nif，如表1-1 及表1-2

表1‑ TM模块名

|  |  |  |
| --- | --- | --- |
| 中文名 | 英文名 | 目录名 |
| 在线接口 | NIF | nif |
| 拥塞避免 | CAVD | cavd |
| 队列管理 | QCU | qcu |
| 流量统计 | QSTAT | qstat |
| 缓存管理 | BMM | bmm |
| 交换接口 | EXI(IEXI , EEXI) | exi |
| 报文重组 | ROU | rou |
| HBM控制器 | HBM | hbm |
| 拥塞管理 | SCH | sch |
| 流量整形 | SHAP | shap |
| 出端口流量管理 | ETM | etm |

表1‑2 Switch模块名

|  |  |  |
| --- | --- | --- |
| 中文名 | 英文名 | 目录名 |
| 交换MAC | XIF | xif |
| 数据信元复用 | CMX | cmx |
| 前级交换 | IXU | ixu |
| 路由处理 | RFU | rfu |
| 后级交换 | OXU | oxu |
| 数据信元解复用 | CDMX | cdmx |
| 控制信元分发 | CFU | cfu |
| 控制信元交换 | CXU | cxu |
| 系统同步 | SSU | ssu |

### 文件命名

文件名称应与该文件定义的类名相同，对应大模块内的各子模块，命名方式采用大驼峰式命名，每个单词的第一个字母大写，如： “BufferManager.h”，”QueueManager.h”等。

### 文件组织结构

1. 文件组织在基目录下，包含bin、obj、include、src、test等目录，在include、src及test内，一个大模块一个独立目录，除main.c等公共文件外，其他模块必须位于对应的子目录内。结构如下：

ftm

|--------------bin

|--------------obj

| |--------------nif.o //.o文件

|--------------include

| |--------------nif

| |--nif.h //.h文件

|--------------src

| |--------------nif

| |--------------nif.cpp //.cpp文件

|

|-------------- …

|--------------test //testbench目录

| |--------------nif //各大模块单元测试目录

| |--------------cavd

| |--test.o

| |--Makefile

|--------------Makefile //芯片级Makefile

### 文件格式

1. 所有文件应具有如下的文件头说明，包含作者、创建时间、上次编辑时间、上次编辑者、实现功能及相对目录：

/\*

 \* @Author: your name

 \* @Date: 2020-11-07 10:06:20

 \* @LastEditTime: 2020-11-26 16:58:56

 \* @LastEditors: your name

 \* @Description: In User Settings Edit

 \*/

1. 所有方法应具有如下的函数说明，包含描述、输入参数、返回值、上次修改人：

/\*\*

 \* @description:

 \* @param {\*}

 \* @return {\*}

 \* @last editor:

 \*/

## 类

### 2.1 类命名

1. 类名与对应的头文件名相同，采用大驼峰式命名，例如：

class BufferManager{};

2. 除1.1节描述的大模块名外，类名一律不使用缩写；

### 2.2 struct 和 class

仅当只包含数据成员时使用struct，其他情况一律使用class。

### 2.3 构造函数

1. 所有类成员必须在构造函数中初始化；

### 2.4 访问限定

成员变量必须严格限制访问权限，对外仅提供公共的set和get方法设置、获取成员变量，例如：

class Packet class Packet

{ 应修改为 {

Header header; public:

int size; set\_header();

} get\_header();

set\_size();

get\_size();

private:

Header header;

Int size;

}

### 2.5 类成员声明顺序

在类中，依次限定public、protected、private，在每个限定符内，依次声明：

1. typedefs 和 enums；
2. 常量；
3. 构造函数；
4. 析构函数；
5. 成员函数，含静态数据成员；
6. 成员变量，含静态成员变量；

## 变量

### 3.1 变量命名

1. 变量命名采用Linux标准风格，小写+下划线方式；

2. 变量命名要清晰、含义明确，若要使用缩写应使用被广泛认可的缩写；

3. 变量命名采用英文单词或其组合，直观且可拼读，用词准确，禁止使用拼音命名；

### 3.2 命名空间

禁止使用 using name space XXXXXX。

### 3.3 作用域

1. 变量尽可能置于最小作用域，并在声明时初始化；

2. 静态变量必须是原生数据类型(int , char , float等)；

## 函数

### 4.1 函数命名

1. 函数以小写字母+下划线方式命名，一般为懂宾词组，如：

set\_header();

set\_out\_port();

### 4.2 函数参数

1. 函数的参数顺序为：输入参数在前，输出参数在后；(输入参数一般为值参或const引用，输出参数一般为非const指针 )；

### 4.3 函数返回

1. 即便函数返回值为 void ，也必须有一个显示的return语句；

2. ”成功“或”失败”一律不通过返回值表示，采用统一的异常类抛出/处理；

3. 调用有返回值的函数时必须处理其返回值；

## 五、编码风格

### 5.1 换行

1. 一行只写一条语句，不允许把多条语句写在一行中；

错误：

if (…) flag = true;

正确：

If (…)

{

flag = true;

}

1. 一行代码不超过120行，超长代码应换行；
2. 若需要换行，则应按以下原则换行：
3. 在逗号 ” , “ 处换行；
4. 在操作符后换行；
5. 长表达式要在低优先级操作符处换行，操作符放在行尾；
6. 换行后，新的起始位置应和上一行对应级别处相同；
7. If , while , for , defult 等语句自占一行；
8. 多行代码构成的表达式或函数参数，不要在参数末尾加入多余的换行符 “\”；

### 5.2 缩进

1. 在函数的开始、定义及if、for、while、switch、case语句中的程序都要缩进，缩进距离是4空格；

2. 禁止使用”Tab”缩进对齐；

3. 连续赋值时，按赋值运算符”=”对齐；

### 5.3 大括号

1. if、else、for、do、switch、case、default等语句各自占一行，其后执行的语句，一律使用{}括起来，哪怕只一语句；

2. 大括号”{“和”}”自己独占一行，成对的大括号缩进要对齐，并与引用它的语句if、for等左对齐；

### 5.4 空格

1. 关键字 (语句)在使用时，”(”前必须有一个空格，函数名后不带空格，如：

for ( ; ; )

if (flag == true)

2. 在使用逗号分隔参数时，逗号后必须有一个空格，如：

foo(int param\_a, int paramb);

1. 如果”;”不是一行的结束，其后要加空格，如：

for(int i =0; i<100; i++){}

1. 规定强制类型转换后没有空格，如:

param\_a = (int)param\_b;

1. 禁止使用goto;