**FAST 入门指南**

1. **树结构**

FAST项目在github（https://github.com/FAST-Switch/fast）上的代码树结构如下：

fast

│

├─lib（共享库，包含了所有设计自定义交换机所需要的平台适配代码，硬件流水库和软

│ 件代码库）

│ ├─hardware（硬件有关的所有Verilog代码）

│ │ ├─pipeline（FAST流水线库）

│ │ │ ├─action（动作执行库）

│ │ │ ├─buffer（缓冲区管理库）

│ │ │ ├─keygen（关键字产生库）

│ │ │ ├─match（查表实现库）

│ │ │ ├─parser（协议分析库）

│ │ │ └─other（其他库，如计数器等）

│ │ └─platform（平台相关适配代码）

│ │ ├─NetFPGA（NetFPGA适配器）

│ │ ├─NetMagic08（NetMagic08适配器）

│ │ └─NetMagicPro（NetMagicPro适配器）

│ │ （以NetMagic08为例，它分为三部分：公共数据路径 CDP，用户模块 UM │ │ 和管理逻辑 ML。其中CDP和ML的逻辑已经固定，用户通常只需要改变│ │ UM模块来实现所需要的功能，平台相关代码即是指非UM模块的代码。│ │ 想使用NetMagic08作为研究平台的用户需要下载并阅读UM规范，理解│ │ UM和CDP之间、以及UM和ML之间的数据格式和接口时序关系，再来│ │ 设计UM模块。）

│ │

│ └─software（软件适配有关的所有C代码）

│ ├─channel（南向接口）

│ │ └─openflow（openflow协议适配代码，包括与控制器建立底层连接，openflow│ 消息解析，openflow消息分派等）

│ ├─extension（转发面功能扩展服务，比如有状态防火墙）

│ ├─platform（平台相关通信库）

│ │ ├─NetFPGA

│ │ ├─NetMagic08（主要是NMAC协议）

│ │ └─NetMagicPro（NPE驱动+DMA寄存器读写）

│ ├─services（交换机管理软件提供的各种服务）

│ │ ├─BVLookup（BV算法查找服务）

│ │ ├─CounterManager（统计管理服务）

│ │ ├─TokenbucketManager（令牌桶管理服务）

│ │ ├─TableManager（表管理服务）

│ │ └─TCAMLookup（TCAM算法查找服务）

│ └─UI（Python+Django编写的本地配置管理界面）

└─projects（包括已近实现的用户项目和参考设计项目）

│ ├─NetLabs（NetLabs是NetEXP教学体系的教学资源中一款网络教学软件，用户使用│ │ 本软件配合NetMagic硬件通过观测网络内部的分组处理行为，加深对网络交│ │ 换、 网络协议和网络管理中的基本原理认识，轻松学习到网络报文统计、流│ │ 量统计、令 牌桶原理、ACL原理、TCP原理等）

│ │ ├─hw-src（这里将要放置硬件有关的所有Verilog代码，包括fast流水线库、平台│ 适配代码和工程配置文件）

│ │ ├─sw-src（这里将放置软件有关的所有C代码）

│ │ └─lib-modules.txt （lib-modules.txt文件指定了所设计交换机的所有需要的lib

│ 文件路径）

│ └─SDTS（软件定义隧道交换机（Software-Defined Tunnel Switch），此工程是为用户展│ 示如何利用FAST提供的共享库来快速设计一个基于端口隧道封装转发的SDN交│ 换机的案例工程）

│ ├─hw-src（这里将要放置硬件有关的所有Verilog代码，包括fast流水线库、平台│ 适配代码和工程配置文件）

│ ├─sw-src（这里将放置软件有关的所有C代码，包括南向接口代码，核心管理服│ 务代码以及与平台有关的通信库）

│ └─lib-modules.txt（lib-modules.txt文件指定了所设计交换机的所有需要的lib库文│ 件路径，最终设计的交换机管理软件就是通过这个配置文件进行│ Makefile文件设计，然后编译成可执行文件）

├─docs（fast项目有关文档，包括详细设计文档、用户操作文档、fast白皮书）

├─README（内容包括代码树介绍、设计案例介绍）

└─LICENSE（开源许可证，本项目遵循Apache 2.0开源许可证）

该树的设计,是为了方便用户工程通过选择共享的lib库，快速搭建不同需求的SDN交换机。

1. **设计交换机**

学习如何使用树中的共享lib库来设计自定交换机的最好方式就是参考实例分析，我们通过分析SDTS（Software-Defined Tunnel Switch）交换机的设计，来向用户介绍如何通过配置文件选择lib库快速搭建一个基于端口隧道封装转发的小型SDN交换机。从github获取fast源码:

$git clone https:// github.com/FAST-Switch/fast.git

$cd fast/projexts/SDTS

进入SDTS目录后，你会看到下面现象：

1. hw-src目录：此目录目前为空，当用户开始设计交换机时，通过已经写好的配置文件，提取所需要的Verilog代码，然后在Quartus II上就可直接运行该工程；
2. sw-src目录：此目录中有两个文件Makefile文件和table\_features文件（这个文件建立openflow通道时需要）。这里面没有C代码的原因是，这个工程可以直接在lib库中选择现有的基于平台的通信库和服务源码就能够编译成一个完整功能的交换机管理软件；
3. lib-modules.txt：这是组合成SDTS交换机的配置文件，里面包含了SDTS交换机所需的所有硬件代码和软件代码的路径。

开始设计交换机之前，最好重新创建一个SDTS目录，再在这个目录中按照配置文件添加和修改文件。

**2.1、使用库模块**

lib-modules.txt文件指定了所设计交换机的所有需要的lib库文件路径。当然用户可以根据需求，通过修改lib-modules.txt文件中流水线库、平台或者服务的路径来选择要使用那些流水线库、平台等。

**2.2 添加代码**

用户根据lib-modules.txt文件内容，到lib库中选择对应的软硬件代码库，硬件代码首先去/lib/hardware/pipeline目录下选择对应的流水线库，然后去/lib/hardware/platform目录下选择对应的硬件适配代码；软件代码首先去/lib/software/channel目录下选择通信库，其次到/lib/software/services目录下选择对应服务代码，最后到/lib/software/platform目录下选择硬件平台通信库。另外用户也可以在hw-src目录里添加自己编写的Verilog代码和在sw-src目录中添加自己编写的C代码（注添加的代码最好根据性质放到对应目录下）。这里我们以SDTS为例进行说明，下面是SDTS目录中lib-modules.txt文件的内容：

硬件：

/lib/hardware/pipeline/action;

/lib/hardware/pipeline/keygen;

/lib/hardware/pipeline/match;

/lib/hardware/pipeline/buffer;

/lib/hardware/pipeline/parser;

/lib/hardware/pipeline/fifo;

/lib/hardware/platform/NetMagic08;

软件：

/lib/software/channel/openflow;

/lib/software/services/BVLookup;

/lib/software/services/TableManage;

/lib/software/platform/NetMagic08;

根据上面的内容，分别把对应的代码从lib库中提取到hw-src和sw-src目录下,注意这里只提取上面目录下的代码，并不包括代码所在目录，即比如我们只需提取openflow目录下的所有代码，但不包括openflow这个目录。

**2.3、改写库代码**

当共享库满足不了用户自己的时候，用户也可以根据需求修改库代码，比如目前软件库中的CounterManager，CounterManager默认情况下是定时把报文的前64字节报文通过packet-in消息上报给控制器，如果用户做流量统计或测量，还需要报文的更多信息，则可以修改CounterManager代码，更改默认的64字节配置。

1. **环境配置**
2. Ubuntu 14.04 64位（也可以是其它Linux系统），本次讲解案例在Ubuntu 14.04下测试通过(注，本次测试交换机管理软件和控制器同时部署在这台Ubuntu系统上)；
3. 安装开发库：libpacp、libnet
4. 配置Java开发环境，主要是为了运行floodlight控制器
5. 配置IP地址（注 ，下面的所有终端命令操作都是在root权限下），这里以测试机为例；

#ifconfig eth0 inet6 add 2001:1:10/64 up

1. Win7系统+Quartus II 15.0（64位）
2. NetMagic08设备
3. **编译运行**

软件编译步骤：（在Ubuntu环境下操作）

1. 下载fast源码

$git clone https:// https:// github.com/FAST-Switch/fast.git

1. 进入sw-src目录
2. $cd fast/projects/SDTS/sw-src
3. 按照2.2步骤添加代码
4. 执行Makefile文件，在sw-src目录下生成nms可执行文件

$sudo make clean

$sudo make

1. 运行floodlight控制器（这里假设用户已经下载安装好floodlight）

$java jar target/floodlight.jar

1. 运行SDTS交换机管理软件，即执行nms可执行文件

$ cd sw-src

$./nms 2001:1::10

1. 打开浏览器，输入地址<http://[2001:1::10]:8080/ui/index.html>

当您在floodlight提供的UI界面里发现交换机，则说明我们自行设计的SDTS交换机管理软件已近成功运行。

硬件编译：（在Windows 7 操作系统下操作）

1. 下载fast源码

到此网址（https:// github.com/FAST-Switch/fast.zip）下载源码并解压到当前目录

1. 进入fast/projects/SDTS/hw-src目录
2. 按照2.2步骤添加代码
3. 运行Quartus II 软件
4. 编译硬件代码
5. 烧录代码
6. **联系**

Email：netmagic\_fast@outlook.com

Web：https://fast-switch.github.io