FAST
FPGA-bAsed SDN swiTch

FAST：基于FPGA的SDN交换机开源项目

报告人：李韬

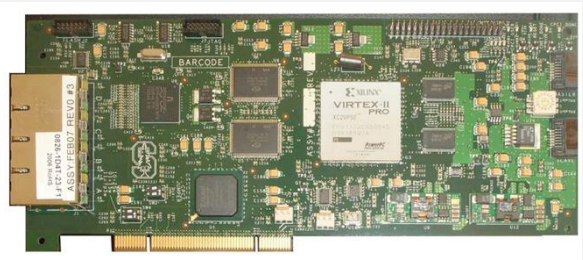
国防科学技术大学计算机学院

网络与信息安全研究所

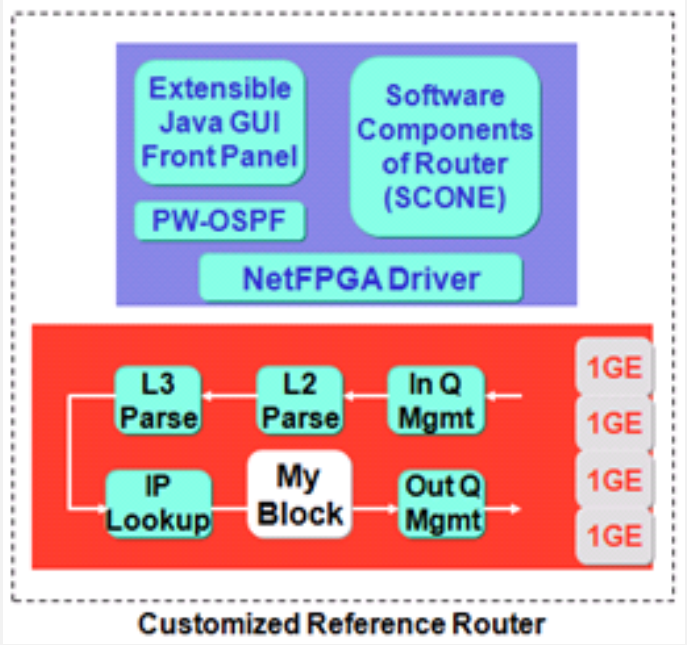
1 项目背景

从NetFPGA说起...

- 对于网络实验平台的关注源于自身**教学**和**科研**的迫切需求
  - 研究生网络实验课程“路由器原理与设计”等
  - 863课题“可重构路由器”等



- NetFPGA优点与痛点
  - 案例丰富、易于获取
  - **复杂度、平台绑定、技术支持**




```
netfpga              {Base directory}
   bin               {Scripts for simulation/synthesis/register gen}
   bitfiles          {Compiled hardware bitfiles}
   lib               {Libraries and software tools}
      C              {C libraries/programs}
      java           {Java libraries}
      Makefiles      {Makefile templates used for sim/synth}
      Perl5          {Perl libraries}
      python         {Python libraries}
      release        {XML files for packaging}
      scripts        {Utility scripts}
      verilog
         contributed {Contributed Verilog modules}
         core        {Official Verilog modules}
      xml            {XML schemas}
   projects          {project directory}
      <project>      {contributed project}
         doc         {documentation}
         include     {project.xml, project specific module XML}
         lib         {Perl and C headers}
         src         {non-library verilog}
         sw          {Project-specific software}
         synth       {Synthesis directory (contains all .xco files)}
         test        {Unified (Hw/Sw) tests}
```

NetFPGA目录结构

自己动手，定制FPGA开源网络实验平台

团队具有十余年高性能网络设备系统工程研制经验，

长期承担大量的网络教学和科研任务

NetMagic开放可编程网络实验平台

基于FPGA+交换芯片的部分可编程交换机

2009

轻量级硬件可编程网络实验平台

NetMagic08

2011

高性能软硬件可编程网络实验平台

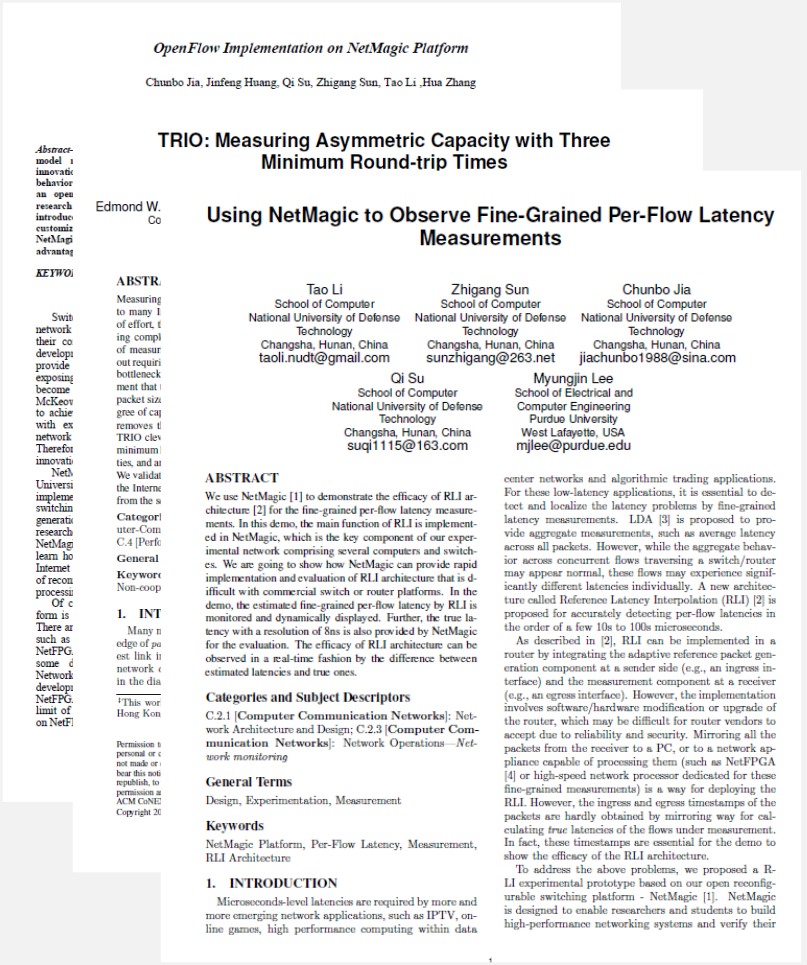
NetMagic-Pro

2013

NetMagic的发展

已在国内外超过20所院校和科研单位应用

基于NetMagic平台已发表科研论文30余篇



OpenFlow Implementation on NetMagic Platform

Chunbo Jia, Jinfeng Huang, Qi Su, Zhigang Sun, Tao Li ,Hua Zhang

TRIO: Measuring Asymmetric Capacity with Three Minimum Round-trip Times

Edmond W.
Co

Using NetMagic to Observe Fine-Grained Per-Flow Latency Measurements

Tao Li
School of Computer
National University of Defense Technology
Changsha, Hunan, China
taoli.nudt@gmail.com

Zhigang Sun
School of Computer
National University of Defense Technology
Changsha, Hunan, China
sunzhigang@263.net

Chunbo Jia
School of Computer
National University of Defense Technology
Changsha, Hunan, China
jiachunbo1988@sina.com

Qi Su
School of Computer
National University of Defense Technology
Changsha, Hunan, China
suqi1115@163.com

Myungjin Lee
School of Electrical and Computer Engineering
Purdue University
West Lafayette, USA
mjlee@purdue.edu

ABSTRACT

We use NetMagic [1] to demonstrate the efficacy of RLI architecture [2] for the fine-grained per-flow latency measurements. In this demo, the main function of RLI is implemented in NetMagic, which is the key component of our experimental network comprising several computers and switches. We are going to show how NetMagic can provide rapid implementation and evaluation of RLI architecture that is difficult with commercial switch or router platforms. In the demo, the estimated fine-grained per-flow latency by RLI is monitored and dynamically displayed. Further, the true latency with a resolution of 8ns is also provided by NetMagic for the evaluation. The efficacy of RLI architecture can be observed in a real-time fashion by the difference between estimated latencies and true ones.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design; C.2.3 [**Computer Communication Networks**]: Network Operations—*Network monitoring*

General Terms

Design, Experimentation, Measurement

Keywords

NetMagic Platform, Per-Flow Latency, Measurement, RLI Architecture

1. INTRODUCTION

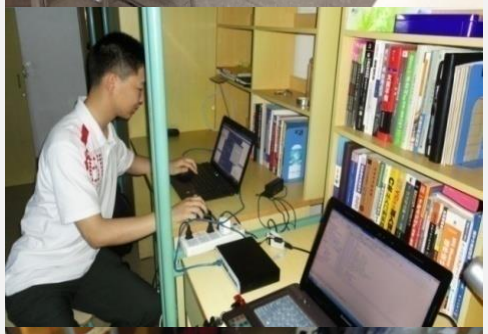
Microseconds-level latencies are required by more and more emerging network applications, such as IPTV, online games, high performance computing within data center networks and algorithmic trading applications. For these low-latency applications, it is essential to detect and localize the latency problems by fine-grained latency measurements. LDA [3] is proposed to provide aggregate measurements, such as average latency across all packets. However, while the aggregate behavior across concurrent flows traversing a switch/router may appear normal, these flows may experience significantly different latencies individually. A new architecture called Reference Latency Interpolation (RLI) [2] is proposed for accurately detecting per-flow latencies in the order of a few 10s to 100s microseconds.

As described in [2], RLI can be implemented in a router by integrating the adaptive reference packet generation component at a sender side (e.g., an ingress interface) and the measurement component at a receiver (e.g., an egress interface). However, the implementation involves software/hardware modification or upgrade of the router, which may be difficult for router vendors to accept due to reliability and security. Mirroring all the packets from the receiver to a PC, or to a network appliance capable of processing them (such as NetFPGA [4] or high-speed network processor dedicated for these fine-grained measurements) is a way for deploying the RLI. However, the ingress and egress timestamps of the packets are hardly obtained by mirroring way for calculating *true* latencies of the flows under measurement. In fact, these timestamps are essential for the demo to show the efficacy of the RLI architecture.

To address the above problems, we proposed a RLI experimental prototype based on our open reconfigurable switching platform - NetMagic [1]. NetMagic is designed to enable researchers and students to build high-performance networking systems and verify their

包括Sigcomm Demo, IEEE Networks, CoNext等

支撑数门本科及研究生网络实验课程



痛点，不仅仅NetMagic...

由于硬件开发、调试的内在复杂度，开源硬件平台的发展面临诸多问题

平台可移植性差

接口电路逻辑、IP核等与平台相关，逻辑设计与平台资源紧耦合，难以跨平台使用和移植

代码可复用度低

缺乏和难以定义统一的模块/部件接口代码和文档质量参差不齐

参与难度大

硬件开发门槛高，投入产出比低

NetMagic项目复用情况

诸多NetMagic项目，除平台相关基础部分外，模块和代码复用率极低。

NetMagic典型项目	项目描述
网络测量	提供端到端高精度测量支持，基于硬件支持Oneprobe网络测量协议，与香港理工合作
细粒度网络流质量监测	提供路由器内部每流传输质量监测，验证细粒度网络流质量监测算法的实现，与普渡大学合作
软件定义硬件交换机SDTS	基于NetMagic08实现的软件定义隧道交换机，实现类LISP的层叠网交换功能
L2Switch-pro	基于NetMagicPro实现的2层以太网交换机
Buffer Bank(Distrbuted buffer management)	类P2P思想实现内容分发在端节点的高效缓存
NetlabSwitch-08	支持网络教学的以太网交换机实现，内置多个观察点，提供十余实验案例
OVS交换机	基于Netmagic08和NetMagicPro平台实现的OVS交换机的扩展和加速
软件定义硬件计数器	NetMagicPro平台实现可灵活定义的硬件计数器验证
LableCast交换机	新的SDN数据平面接口Lablecast的实现与验证
…	….

2 FAST架构

FAST的提出

- 坚持面向**教学**和**科研**，聚焦**网络数据平面**
  - 需求特定利于集中资源，提供技术和发展支撑
  - 领域特定利于接口规范的制定，破解代码可重用问题
- 设计避免**平台依赖**，着力**平台无关功能**部分
  - 解耦平台相关和无关代码，提升功能模块的可移植能力
  - 支持NetMagic、NetFPGA、ONetSwitch、SDNet等平台，最大化凝聚相关力量
- 以**FPGA SDN交换机实现**为突破口
  - 科研教学的热门方向，便于开源社区建设
  - 前期技术积累较多，适合快速推进部署

FAST的提出

- 坚持面向**教学**和**科研**，聚焦**网络数据平面**
  - 需求特定利于集中资源，提供技术和发展支撑
  - 领域特定利于接口规范的制定，破解代码可重用问题
- 设计避免**平台依赖**，着力**平台无关功能**部分
  - 解耦平台相关和无关代码，提升功能模块的可移植能力
  - 支持NetMagic、NetFPGA、ONetSwitch、SDNet等平台，最大化凝聚相关力量
- 以**FPGA SDN交换机实现**为突破口
  - 科研教学的热门方向，便于开源社区建设
  - 前期技术积累较多，适合快速推进部署

FAST技术思路

- 基于模块库的设计
  - 标准规范的基础模块库
  - 模块参数化设计
  - 宏流水线抽象与组合
- 平台可移植性设计
  - 统一的平台数据/控制接口
  - 平台相关及无关逻辑划分
- 支持软硬件协同处理
  - 高速分组IO通道
  - 数据平面可编程扩展

1 灵活高性能

提供可编程、可按需配置的高性能SDN交换处理功能，具有低延迟高带宽的线速转发能力。

2 模块可重构

支持平台无关的模块级可重构功能，尽可能实现模块参数化，有效支持软硬件模块库构建和代码重用。

3 软硬可协同

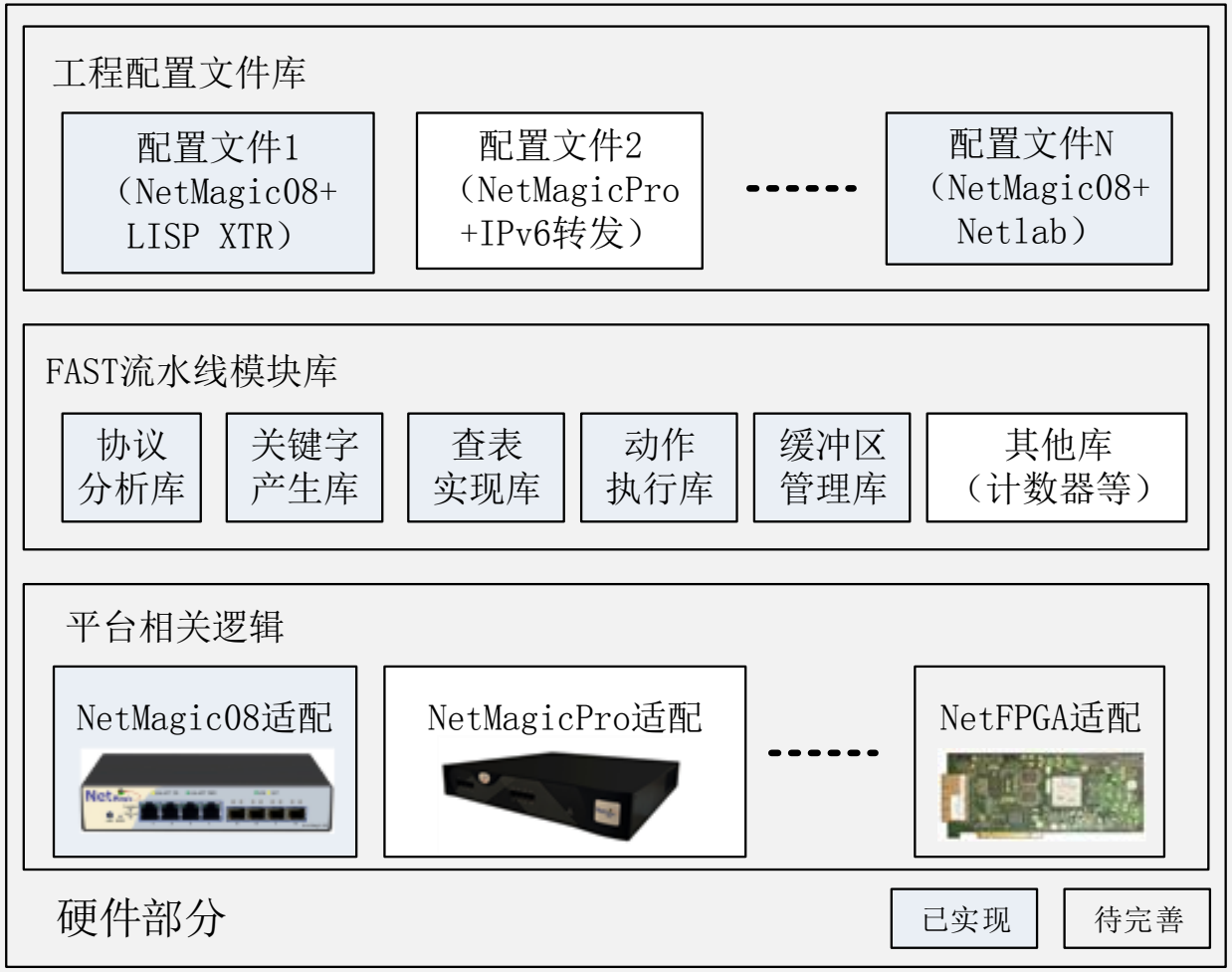
基于软件支持硬件转发平面的扩展，支持更多复杂控制流或状态管理功能的软硬件协同可编程实现和映射

4 开源开放

模块库开源和接口开放，允许第三方模块及代码的无缝集成和替换

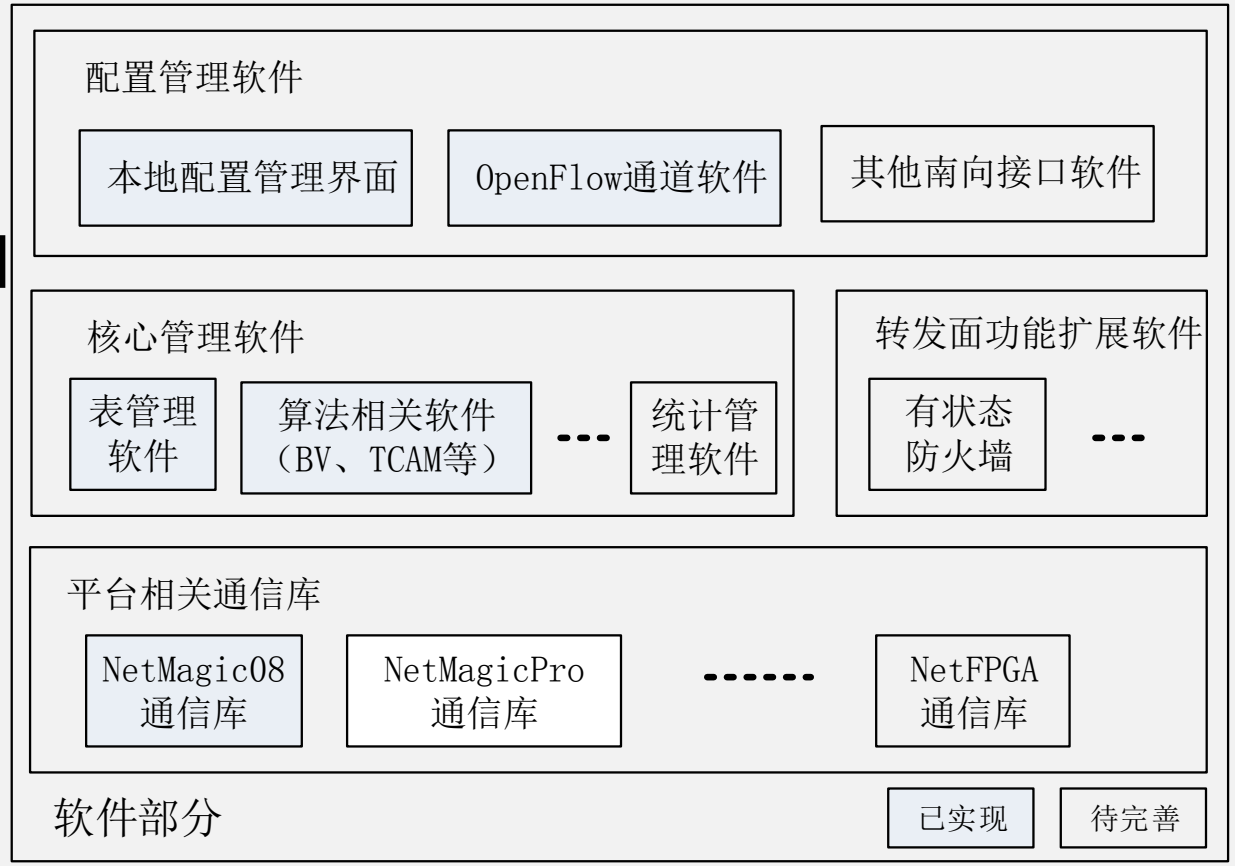
FAST硬件架构

- 平台相关逻辑
  - 适配不同基于FPGA的软硬件平台，提供统一的平台接口
- FAST基础模块库
  - FAST基础模块接口和功能定义不依赖于平台，实现也避免与平台相关
- 工程配置文件库
  - 指定功能模块的连接组合
  - 指定运行平台
  - 指定资源和性能优化约束及目标



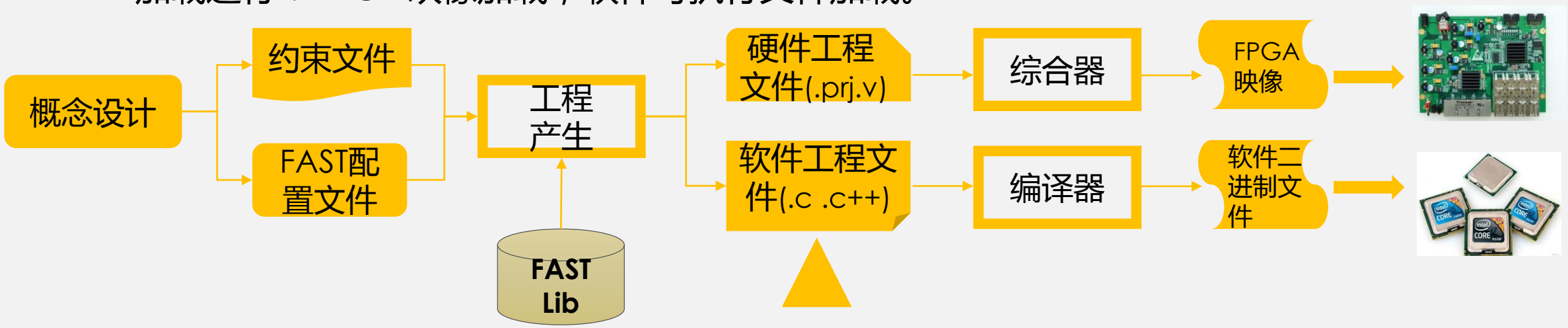
FAST软件架构

- 平台相关通信库
  - FAST软硬件通信接口与平台相关
- FAST核心管理软件
  - 提供硬件功能模块配套管理配置接口
- FAST数据平面功能扩展软件
  - 提供复杂网络处理功能的灵活扩展
- 配置管理软件
  - 配置管理界面
  - 南向接口适配



FAST开发流

- 主要阶段
  - 概念设计：根据功能和应用场景需求形成基于模块的FAST配置文件和平台资源性能约束文件（配置编程语言）
  - 工程产生：在FAST软硬件模块库中搜索组合形成软硬件工程（源代码和配置文件）；必要时需新开发相应模块。
  - 综合编译：将软硬件工程分别通过商用硬件综合器及软件编译器进行综合/编译，形成FPGA映像和软件二进制文件。
  - 加载运行：FPGA映像加载，软件可执行文件加载。

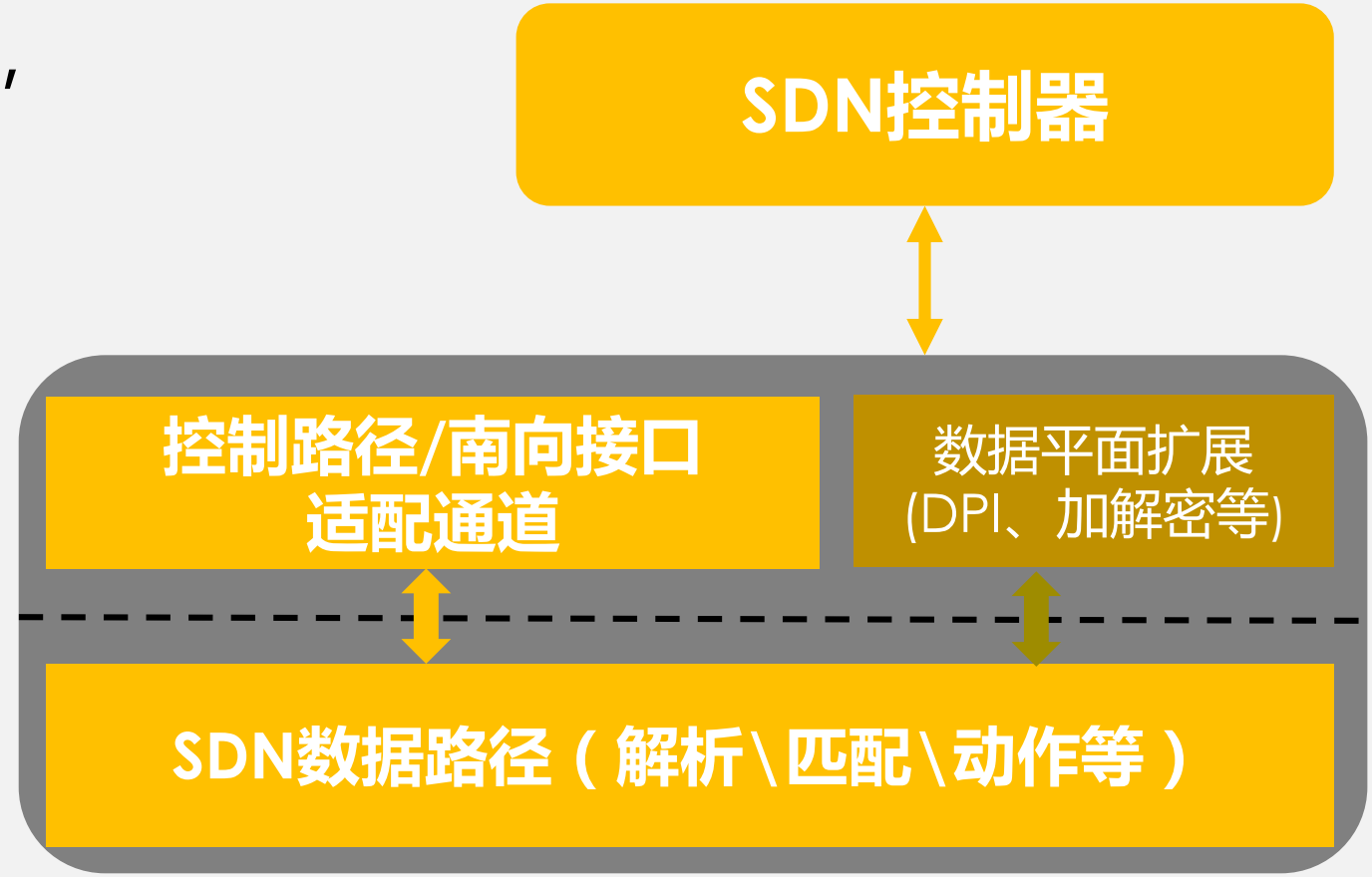


3

FAST规划

FPGA实现SDN交换机关键

- 控制通路：控制通信协议及接口，南向接口适配等
- 数据路径：报文处理流水线，解析、**匹配**、动作等
- 数据平面扩展：支撑通用多核软件线程扩展数据路径处理功能

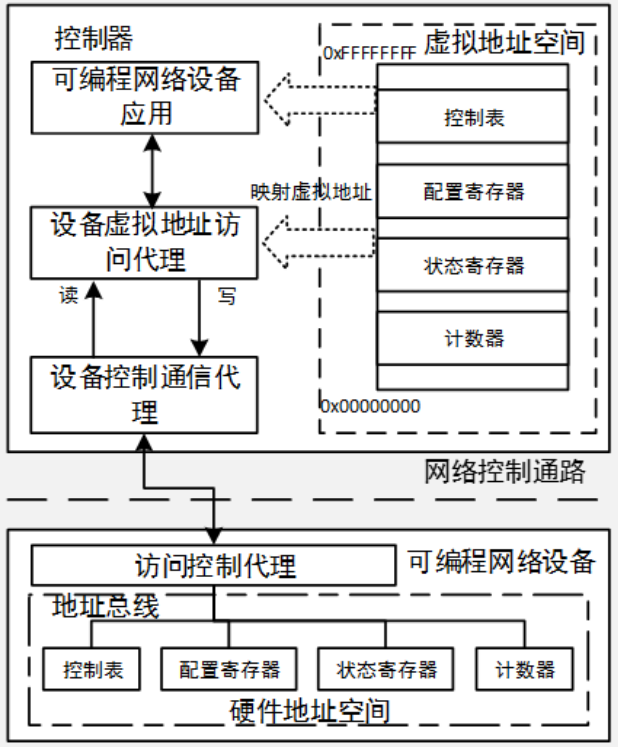


控制通路—控制通信协议及接口

- NMAC软硬件通信协议
  - 轻量级交互开销
  - 基于通用IP报文格式
  - 平台/语义无关API（以太网/PCIE ）

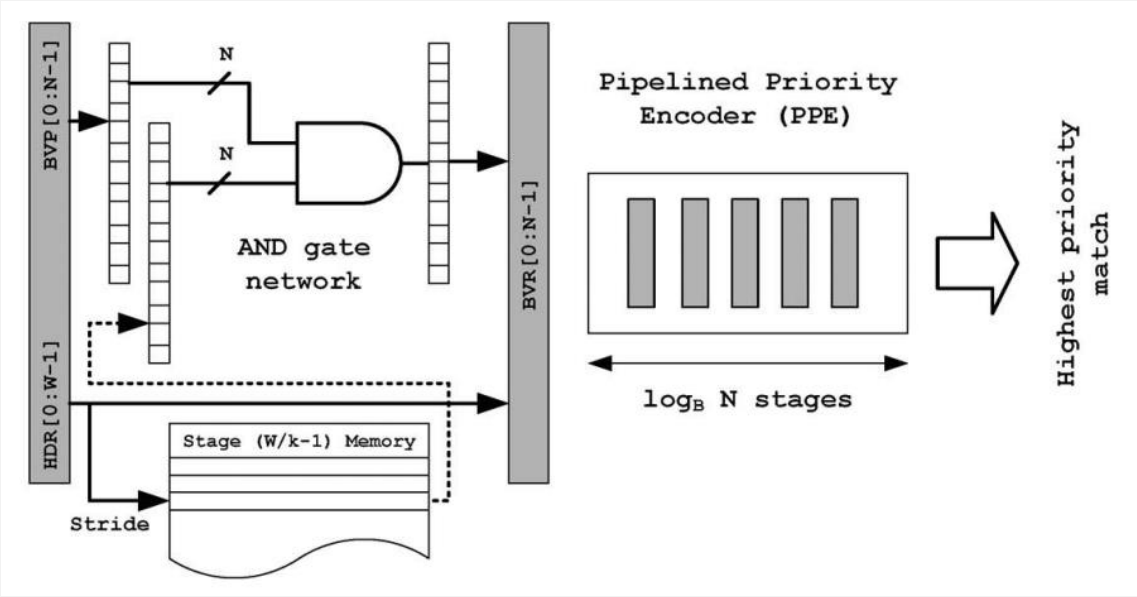
API	描述
int nmac_init(char* NetMagic_ID);	建立控制平面与数据平面的通信连接
void nmac_cleanup();	释放连接
void nmac_write_data(u_int32_t addr, int num, u_int32_t* data);	写数据
u_int32_t* nmac_read_data(int num, u_int32_t addr);	读数据

- 基于虚拟地址空间的访问规范
  - 一致的硬件抽象和约定（寄存器、存储器、计数器等）
  - 支撑软硬件解耦和协同开发



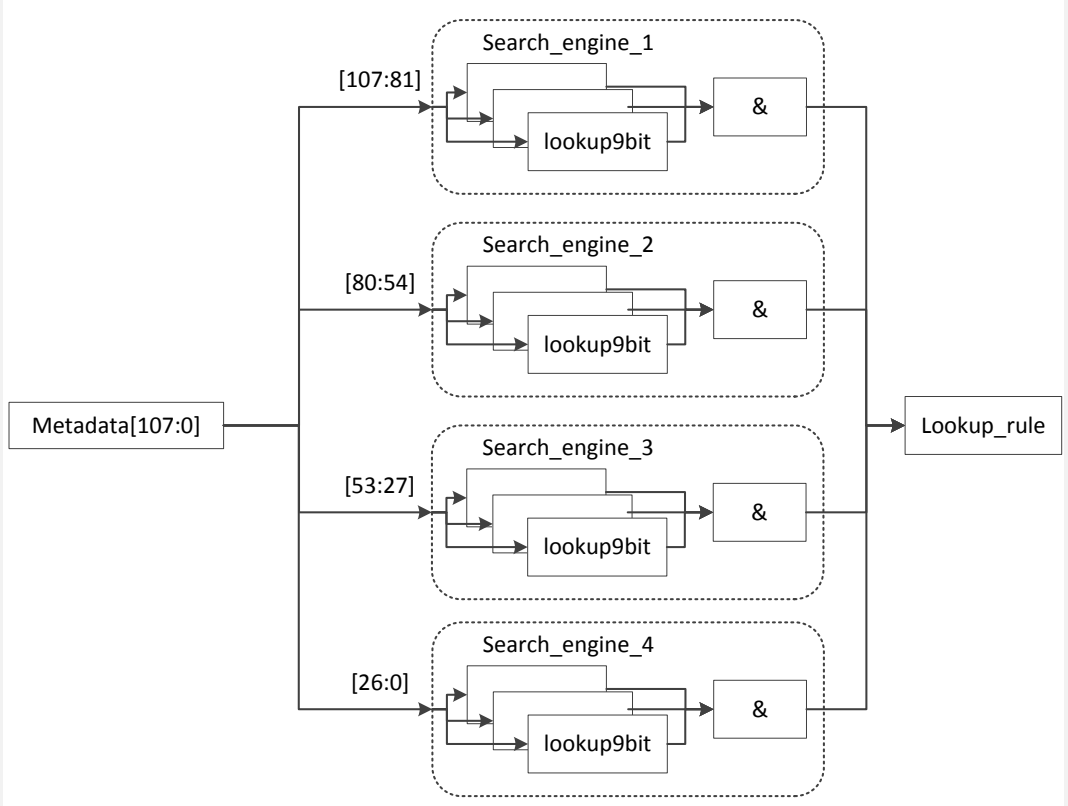
数据路径—StrideBV通配查找算法

- BV&Stride BV查找算法
  - 面向支持通配匹配的高性能报文分类
  - 适合充分利用FPGA逻辑存储资源
  - Virtex7 2000T实现28K规则（160b）



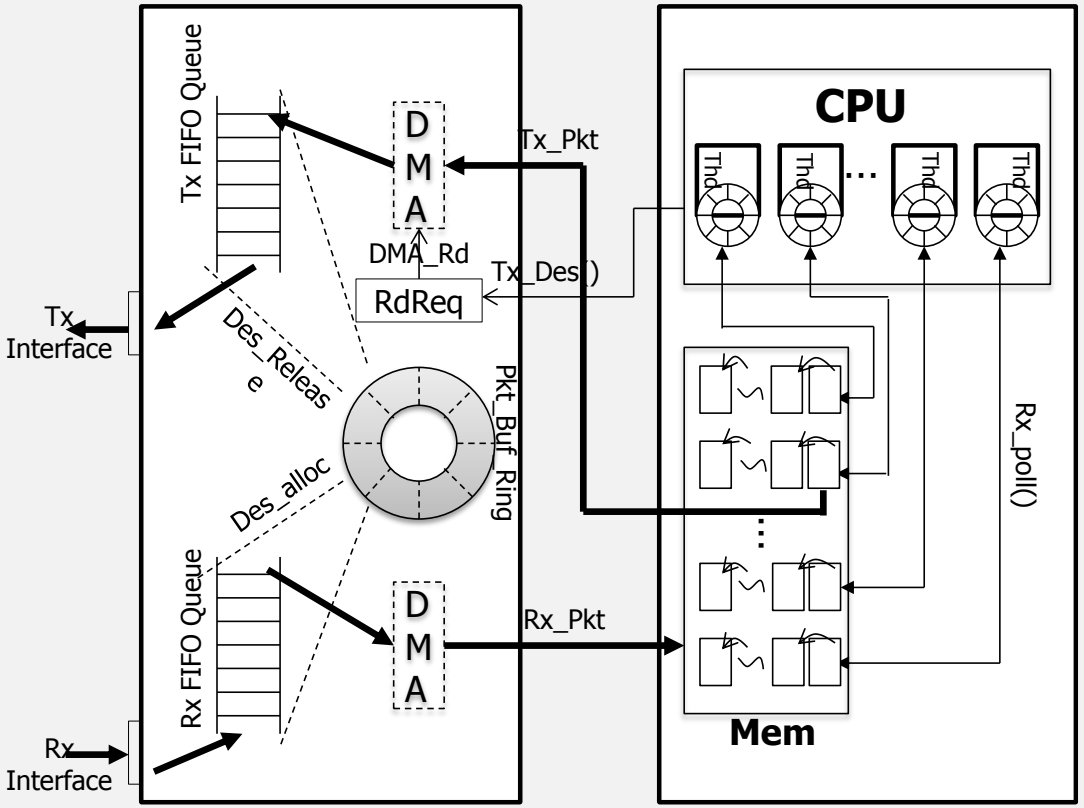
A Scalable and Modular Architecture for High-Performance Packet Classification
TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, 2014

- StrideBV FPGA原型实现
  - 9bits Stride，规则宽度288b，36条
  - 规则数量、关键字长度、Stride可配置

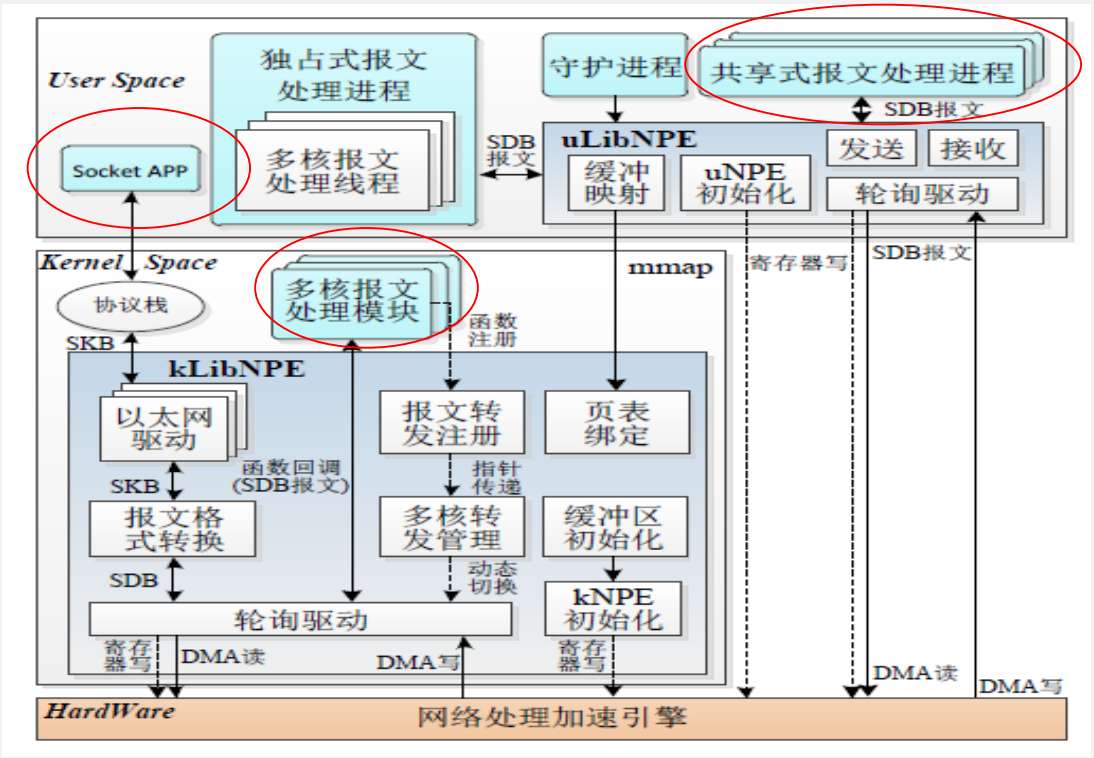


数据平面扩展—数据通信优化方法及接口

- 报文I/O优化技术
  - 缓冲区管理卸载
  - 轮询无中断
  - 多队列支持



- NPDK软件数据平面扩展框架
  - 用户和内核空间高性能分组收发处理接口
  - 提供标准Linux的网络驱动程序接口

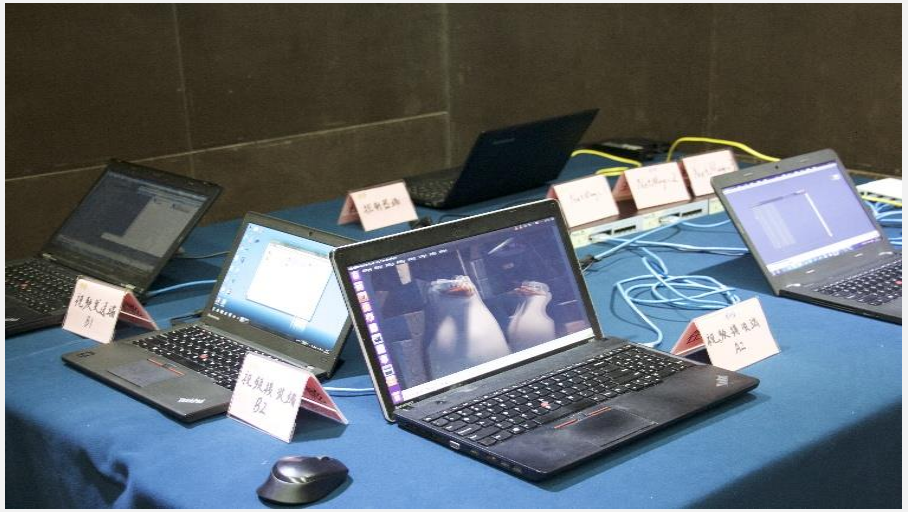


系统实现—SDN交换机原型

- 863课题：面向三网融合的创新网络体系结构
  - 基于NetMagic实现了OpenVSwitch--OFS-Pro，包括分组IO加速、端口扩展、支持OVS数据路径的加速卸载。

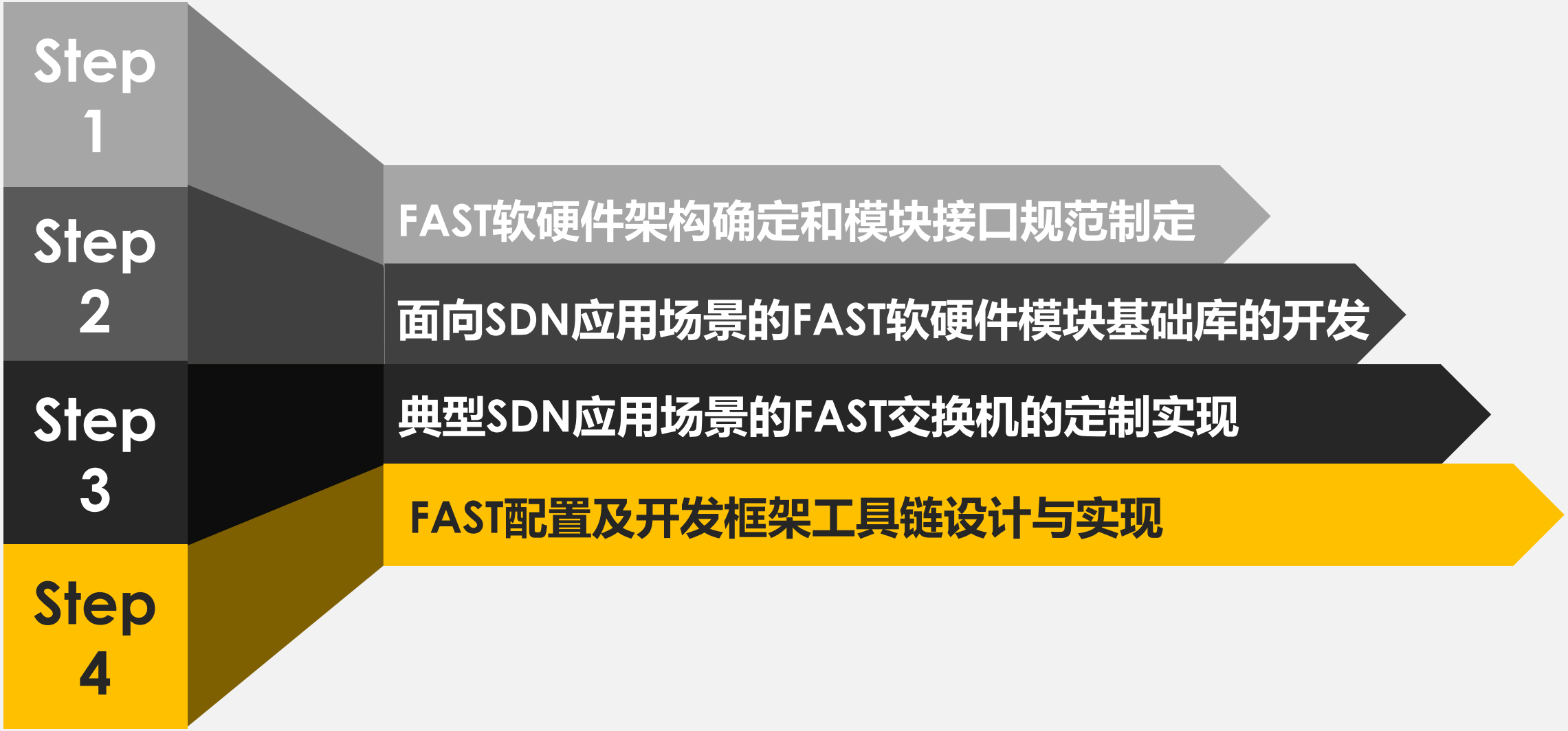
OFS-Pro现场演示

- 863课题：IPv6大规模编址与路由关键技术验证
  - 基于NetMagic平台实现了一个SDN隧道交换机SDTS，以LISP隧道封装和基于端口转发的模式，验证了BV算法对硬件匹配性能效果的提升，配合开源控制器Floodlight搭建了一个低成本、可重构、易部署的SDN实验环境。



SDTS现场演示

FAST发展规划


Step 1
FAST软硬件架构确定和模块接口规范制定
Step 2
面向SDN应用场景的FAST软硬件模块基础库的开发
Step 3
典型SDN应用场景的FAST交换机的定制实现
Step 4
FAST配置及开发框架工具链设计与实现

4 Demo简介

FAST Openflow交换机 Demo拓扑

- 展示基于FAST的Openflow交换机功能
  - FAST交换机：实现与PICA8外部行为类似的基本的SDN交换机，其中FPGA实现Openflow 1.3数据平面，CPU实现交换机适配和配置控制功能
  - SDN控制器：Floodlight控制器，对网络进行控制，并呈现网络拓扑以及流表规则及其他网络相关信息；
  - 视频源端及客户端：发送接收网络流量，制造网络动态变化（链路通断等）。

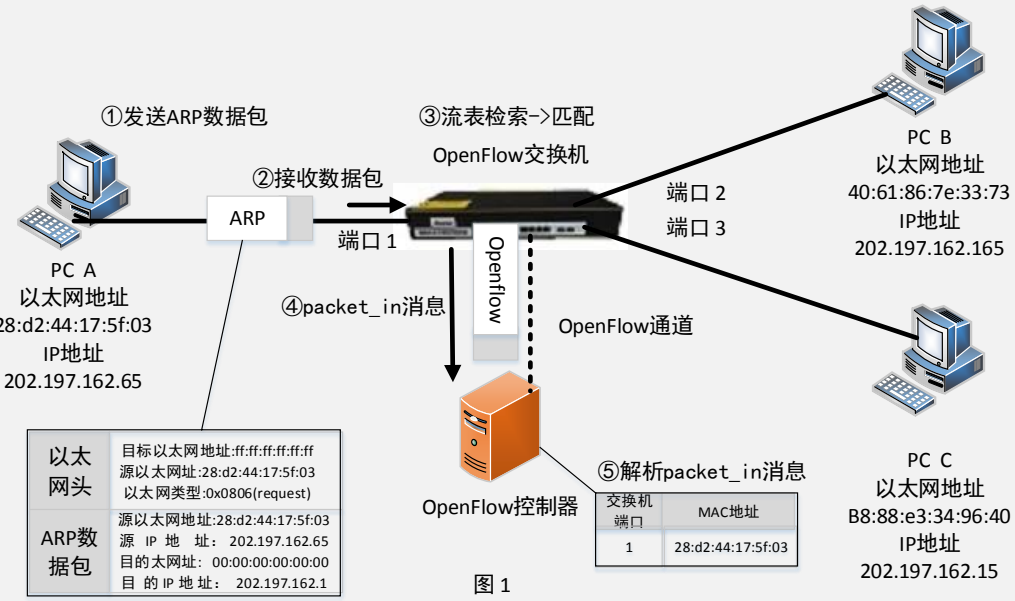
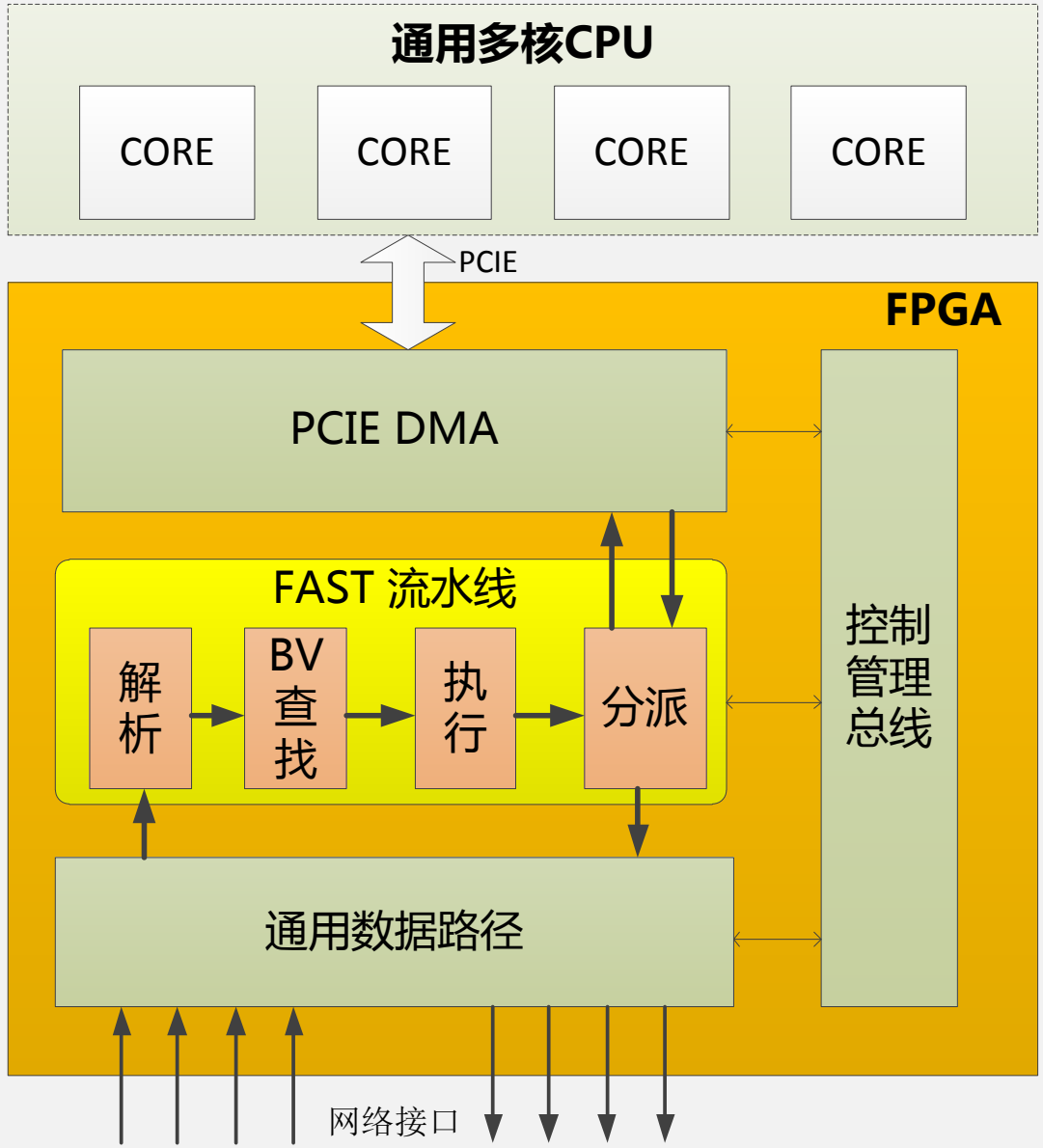


图 1

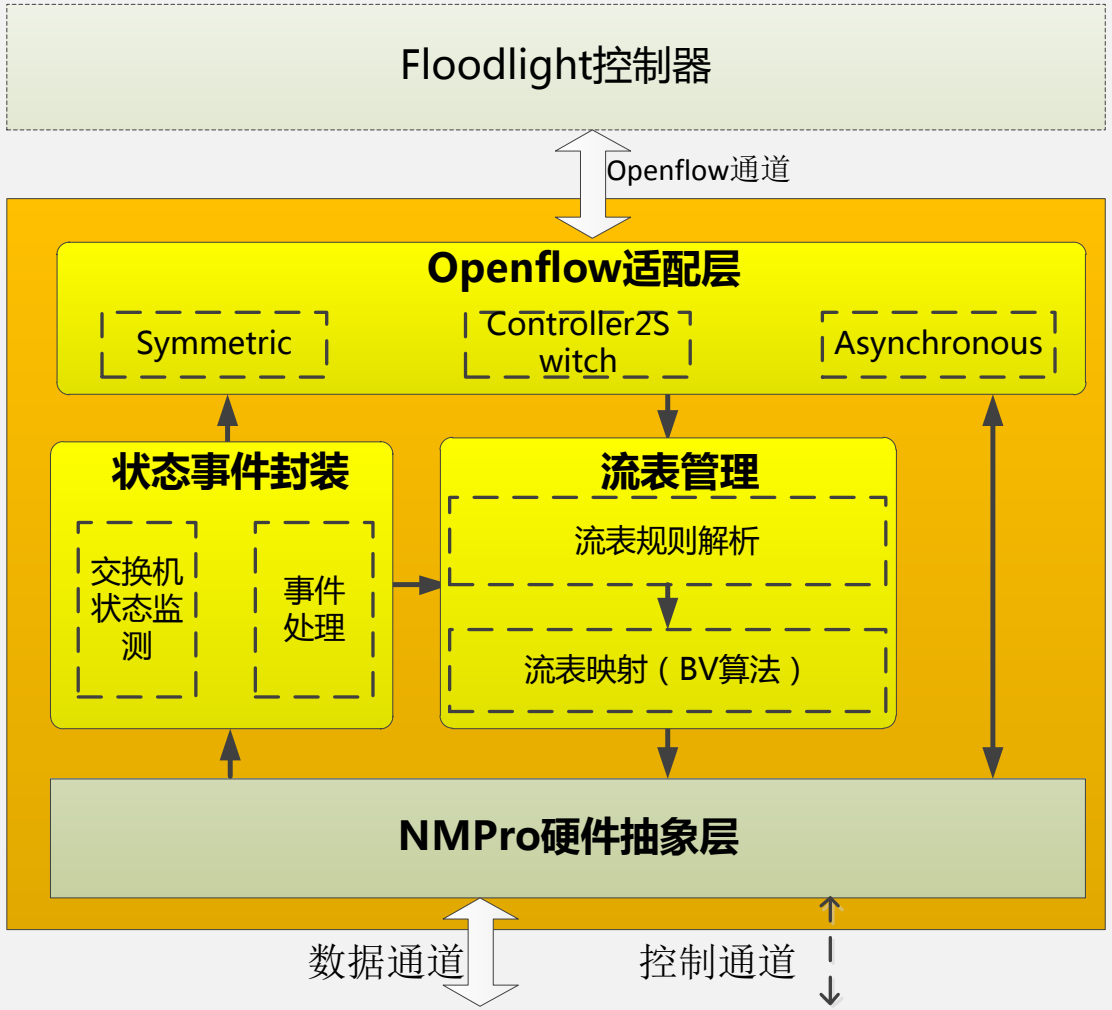
FAST交换机硬件实现

- 基本组成
  - FAST流水线：解析模块（13元组）、BV算法通配查找模块、指令执行模块、报文分派模块
  - 预置通用数据路径：报文输入/输出控制
  - 预置PCIE /DMA部件：提供封装的软件控制及高性能报文数据通信接口
  - 控制管理总线：基于Localbus的读写访问控制总线



FAST交换机软件实现

- 基本组成
  - Openflow适配层：适配SDN南向Openflow接口，支持同步、异步等消息的处理
  - 状态事件封装：监测交换机状态和流量变化，对相应事件进行处理；
  - 流表管理：进行流表规则的解析和到实际硬件表项的映射
  - NMPro硬件抽象层：提供抽象的硬件报文收发接口以及硬件控制管理接口



参与FAST

FAST的开源开放

加速项目的版本迭代

开源可以有效的共享资源，避免不必要的重复劳动，以NetFPGA为例，18个月间就更新了8个版本，大幅度节约了成本和时间。

项目设计开发规范可靠

设计文档在开放平等的基础上供公众审查，在相互借鉴和比较的基础上，能使代码功能设计能够变得有序和规范。例如，NetFPGA 200多份的设计文档就有90余人的贡献。

支撑网络技术创新

由于源码的开放，使得研究人员可以参与并根据需要修改代码，建立相关科研项目，为网络创新技术提供基础支撑

FAST遵循开源原则

OSI开源十条定义

- 自由免费再发布
- 源代码公开
- 允许修改和派生作品发布
- 作者源代码的完整性
- 不能歧视任何个人和团体
- 不能歧视任何领域
- 许可证的发布避免附加条款
- 许可证不能只针对某个产品
- 许可证不能约束其他软件
- 许可证必须独立于技术

FAST开源许可证

- FAST选择Apache 2.0作为项目开源许可证
  - 同样鼓励代码共享和尊重原作者的著作权，同样允许代码修改，再发布
  - 允许修改版本发布不开源，对**商业应用**友好，鼓励业界参与

许可证	类型	要求保留著作权声明	要求对源代码中的修改做出声明或标识	允许再发布收费	允许原作品及其修改版的可执行形式使用其他许可证发型	允许原作品及其修改版发行时不公开源代码	允许被使用其他许可证的软件连接或包含	明示专利授权
GPL v3	强	是	是	是，但不高于发行成本	否	否	否	是
CPL 1.0	弱	是	是	是	是	否	是	是
Apache 2.0	**宽容**	**是**	**是**	**是**	**是**	**是**	**是**	是

FAST开源内容

- FAST开源所有基础框架、软硬件功能模块和相关工具链
  - 软硬件基础模块库：报文解析、关键字提取、掩码匹配查找算法、QoS等HDL源码
  - FAST框架：Openflow控制通道适配、数据平面表管理、BV规则编译下发等软件程序源码
  - FAST工具链：FAST配置文件解析软件、参数化模块产生、脚本文件等

成为FAST一员

初始阶段，任何认可FAST开源开放原则和许可证，并愿意进行贡献或观察的**个人**，可以选择成为FAST“**成员**”或“**观察员**”

FAST成员

权利

- 参加FAST发展目标和规划讨论
- 参加FAST相关规范制定
- 参加FAST技术交流
- 获取其他成员的技术支持

义务

- 优先FAST进行科研教学活动
- 参加FAST需求分析、开发和测试等相关工作
- 为其他成员单位提供力所能及的技术支持

FAST观察员

权利

- 参加FAST技术交流
- 获取其他成员的技术支持

义务

- 为FAST发展提出意见建议

从FAST获得...

无论是否是FAST的一员，都可以从FAST中受益

- **科研院所**：快速构建面向领域应用的SDN实验环境，支撑科研实验数据获取和原型系统实验验证
- **工业部门**：内部培训课程和环境，新技术、算法参考和使用，在改进升级后的产品发布
- **教学单位**：教学案例的构建，基础培训平台的搭建，实验教学的支撑

https://fast-switch.github.io/index.html

FAST

主页 资源 社区 关于FAST

欢迎来到FAST项目

FAST(FPGA bAsed SDN swiTching)是一种以FPGA为转发平面核心的SDN交换机实现架构，其基本思想是可重构交换架构——将报文处理流程拆解成多个独立报文处理阶段，为每个阶段都建立相应的模块库，开发者根据需要自由选择处理模块，用于快速重构报文处理流水线。这种"离线重构"的方式能够满足多样化的SDN交换需求，大幅度降低网络应用服务开发的难度和网络设备的开发周期 。

了解更多

欢迎关注 http://fast-switch.github.io

1 硬件模块开发
硬件模块开发介绍
› 了解更多

2 软件模块开发
软件模块开发介绍
› 了解更多

3 模块组合
选择不同的硬件模块和软件模块，快速搭建不同需求的SDN miniSwitch
› 了解更多

FAST, 软件平面介绍

南向接口软件软件：直接与SDN控制器交互，传输和展示本地配置流表信息，支持OpenFlow协议，未来也可以扩展其他南向接口协议的开发与验证。

核心管理软件：表管理软件用于TTP定义的流表管理维护，配置信息等，与硬件中的信息保持同步；算法相关软件用于实现硬件查表，验证算法的性能和正确性；统计管理软件主要对

谢谢！请批评指正！