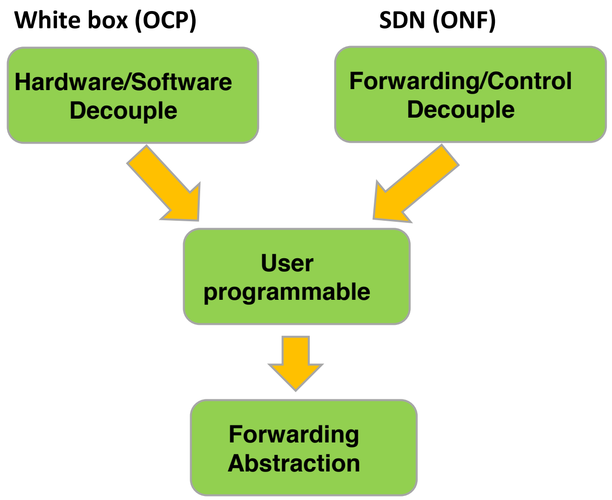
Use P4 to Program NP-based Router through POF Interface

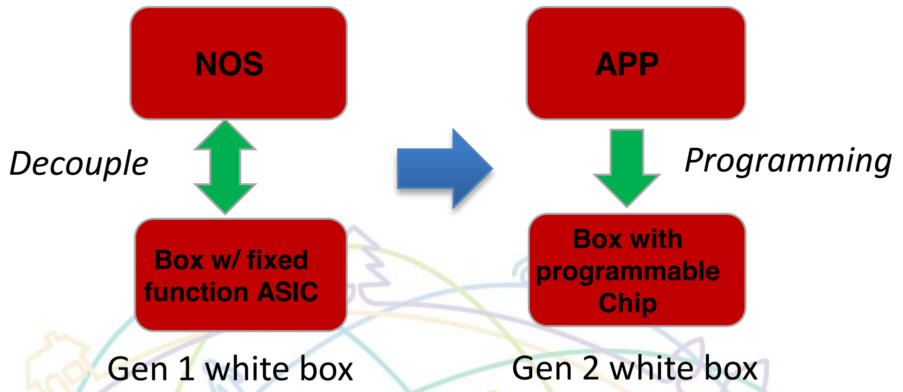
www.huawei.com

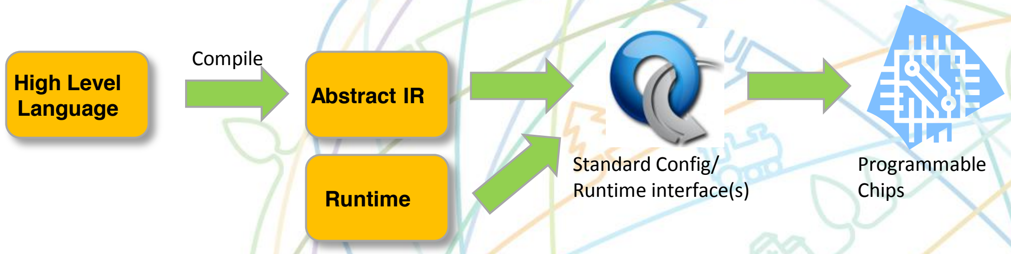
Haoyu Song

HUAWEI

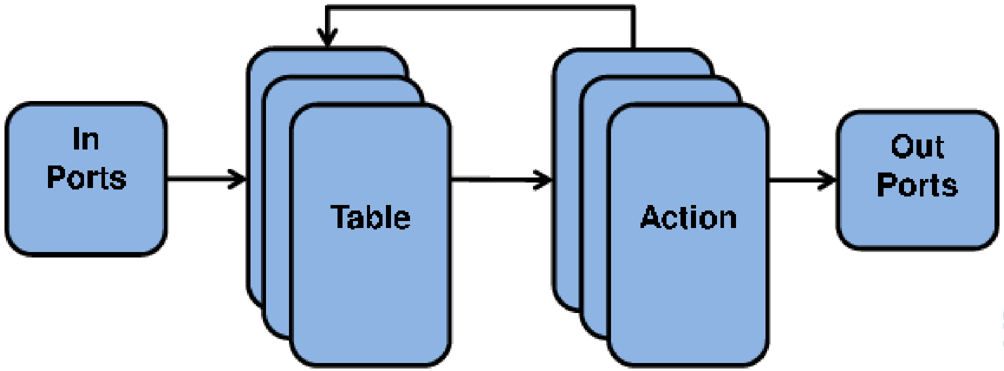
Towards Open Programmable Data Path

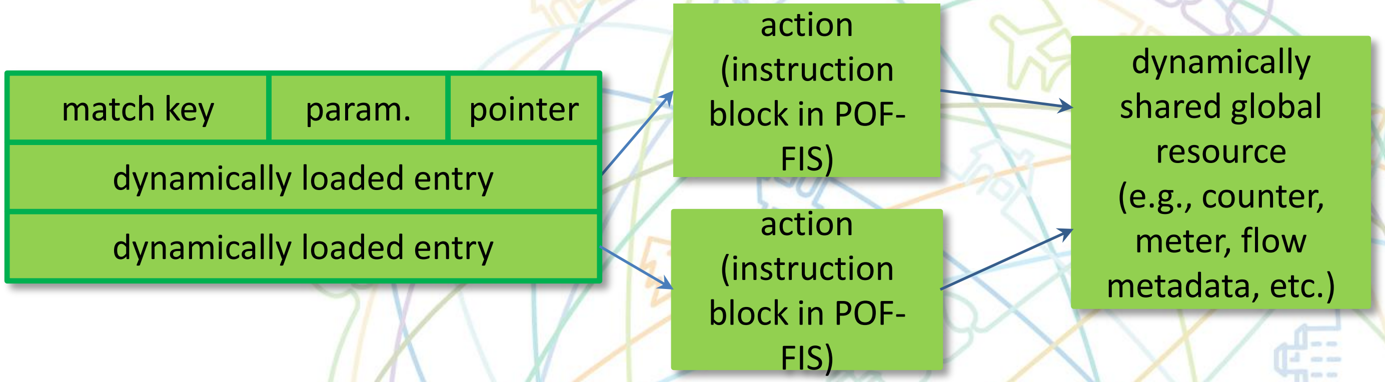




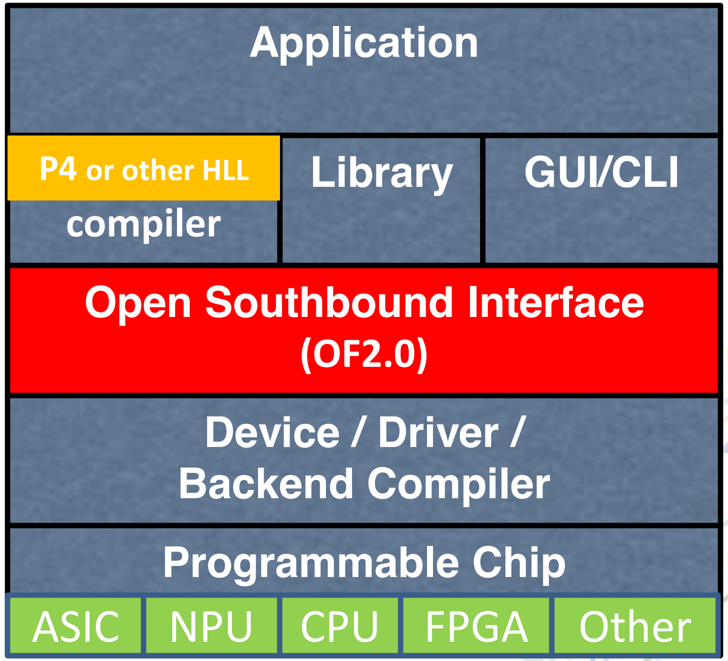


POF Forwarding Abstraction




match key
param.
pointer
dynamically loaded entry
dynamically loaded entry
action (instruction block in POF-FIS)
action (instruction block in POF-FIS)
dynamically shared global resource (e.g., counter, meter, flow metadata, etc.)

POF Programming Architecture



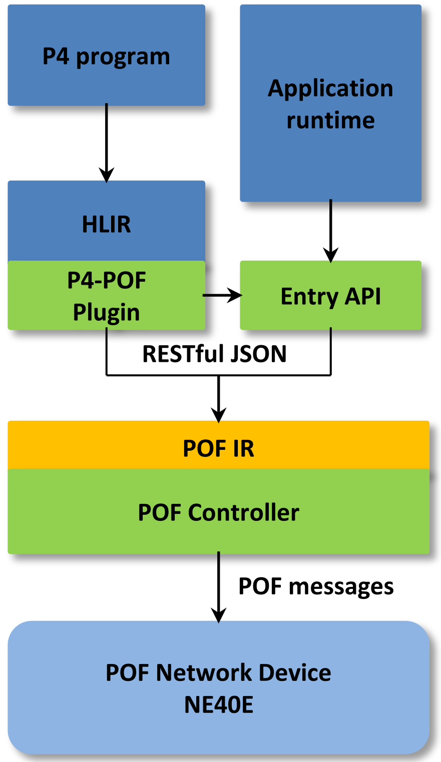
- Configuration and runtime share the same interface
  - Combine static programming and dynamic incremental reconfiguration and control
- POF aims to maximize the flexibility
  - More restrained use is possible

POF IR & SBI: An Extension to OpenFlow

Name	Description
Protocol	Define the protocol header format
Metadata	Define the metadata layout
Table	Define table name, table type, table size, search key, and if the table is a shared table
Table Parameter	Define parameter format for each table that can be accessed by table instruction block (i.e., action)
Instruction Block	Define the actions that will be executed after table matching. Each table entry will point to an instruction block.
Service	Define the table pipeline (similar to control flow in P4)

- Support all existing OF1.4 messages
- Add new POF-specific messages
  - Counter messages
  - Instruction block messages
  - POF Data path enable/disable messages
  - Service messages
- Modify some existing messages to meet POF's requirements
  - table_mod, flow_mod, etc.

Use P4 to Program POF Device



- POF PoC
  - POF Prototype demo in 2013
  - WAN testbed demo in 2015
  - Open source POF controller (www. poforwarding.org)
  - NE40E w/ 240Gbps LC/Slot
  - REST JSON NBI
- P4 Support
  - Reuse open source HLIR compiler
  - P4-POF plugin translates HLIR objects to POF-IR objects, formatted into REST JSON commands
  - POF Controller translates JSON commands to POF messages

Performance & Challenges

- 120G LC → 180Mpps line speed but actual performance falls short of line speed for small packets
  - Architecture implied by P4 spec doesn’t match optimal NP implementation
  - Compiler optimization needed
    - An WIP optimization shows 10~20% improvement
  - Too many TCAM accesses for Parser
    - Combine parser states

Lessons Learnt for Applying P4 on NP

- Decouple the language and the architecture
  - Support incremental "just-in-time" parsing
  - Support modular design
- Design portability is very hard to achieve
  - Reusable library vs. reusable design
  - Standard behavior model but custom code for implementation

HUAWEI

THANK YOU

www.huawei.com