



# P4 Programming

Function blocks and interfaces

Top-down control and design

***NobleProg***



**HITESH KUMAR SHARMA**  
*Technical Instructor & Consultant*

# FUNCTION BLOCKS AND INTERFACES

In P4 programming, function blocks and interfaces play a crucial role in defining the behavior of network devices and how packets are processed. Here's an explanation of function blocks and interfaces in P4:

# FUNCTION BLOCKS

Function blocks in P4 define specific stages of packet processing and encapsulate related operations. They are integral parts of the pipeline through which packets traverse in a P4 program. Each function block focuses on a particular aspect of packet processing, such as parsing, matching, or forwarding.

In P4, function blocks are defined using the control construct. A control block contains a series of operations that are applied to incoming packets as they traverse through the pipeline. The order of control blocks in a P4 program defines the sequence of packet processing stages.

# FUNCTION BLOCKS

In this example, the ingress control block represents a packet's entry point into the pipeline. It applies the `parse_ethernet` function block and then, if a match occurs in the `ipv4_forward` table, applies the associated actions.

## Example:

```
p4
control ingress {
    apply(parse_ethernet);
    if (ipv4_forward.hit) {
        apply(ipv4_forward.apply);
    }
}
```

[Copy code](#)

# INTERFACES

Interfaces in P4 define the communication between different components of a P4 program, such as controls and tables, or between different pipelines. Interfaces define the methods through which data is exchanged between these components.

# INTERFACES

Two common types of interfaces in P4 are:

**1. Control Interfaces:** These interfaces provide a way for control blocks to communicate with each other. For example, a match-action control block can communicate with an associated action control block through an interface.


**2. Table Interfaces:** These interfaces allow tables to communicate with control blocks. Tables can send information to control blocks, and control blocks can use this information to determine actions.

# INTERFACES

In this example, the `hit_stats` interface is used to report statistics from the `ipv4_forward` table. The `apply()` function sends information to the interface, which can be collected for monitoring purposes.

## Example:

```
p4
control ingress {
    apply(parse_ethernet);
    if (ipv4_forward.hit) {
        apply(ipv4_forward.apply);
        hit_stats.report();
    }
}
```

 Copy code

# TOP-DOWN CONTROL AND DESIGN

Top-down control and design is an approach used in P4 programming to design and implement network processing pipelines and behavior. This methodology involves breaking down the network processing logic into functional blocks and defining the high-level structure before delving into the details of each block. Here's how top-down control and design are applied in P4 programming



# TOP-DOWN CONTROL

**1. High-Level Design:** Start by defining the overall architecture of your P4 program. This includes identifying the key components of packet processing, such as parsing, match-action tables, and actions.

**2. Pipeline Stages:** Organize the packet processing into pipeline stages. Each stage corresponds to a function block that performs a specific operation, such as parsing headers, performing matches, and executing actions.

# TOP-DOWN CONTROL

**3. Control Blocks:** Create control blocks for each stage of the pipeline. These control blocks encapsulate the logic and operations to be performed at each stage.

**4. Interfaces:** Define interfaces between control blocks and tables, as well as between different pipeline stages. These interfaces enable communication and data exchange between components.

# DESIGN PRINCIPLES

**1. Modularity:** Design each control block to perform a specific task or set of related tasks. This promotes modularity and makes the program easier to understand and maintain.

**2. Abstraction:** Abstract the complexity of individual control blocks. When designing a control block, focus on its high-level purpose and behavior without getting into the details of implementation.

# IMPLEMENTATION STEPS

- 1. Identify Key Stages:** Determine the main stages of packet processing, such as parsing, matching, and forwarding. These will become the control blocks in your P4 program.
- 2. Create Control Blocks:** Develop control blocks for each stage. Define the operations and logic to be performed at each stage.
- 3. Define Interfaces:** Set up interfaces between control blocks, allowing them to communicate and exchange data.

# DESIGN PRINCIPLES

**4. Pipeline Flow:** Arrange the control blocks in the desired order to represent the pipeline flow. The output of one block becomes the input for the next.

**5. Refine Details:** Once the high-level structure is defined, you can dive into the details of each control block, including the parsing of headers, specifying match conditions, and defining actions.

# CONCLUSION

By using the top-down control and design approach in P4 programming, you create a structured and organized framework for developing network processing pipelines. This approach helps you manage complexity, promote code reuse, and efficiently customize the behavior of network devices.

END OF MODULE-1

THANK YOU