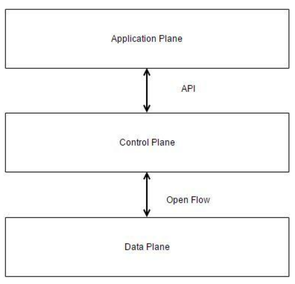
A Design Implementation of a Packet Parser to be used in the SDN switch

Mit Mehta, Morris Jones
Department of Electrical Engineering, San Jose State University, San Jose, California 95192

Introduction

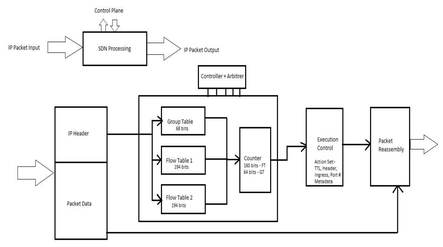
Implementation of a Packet Parser for a 5 port Low Speed SDN Switch.

Software Defined Networking:



SDN [1] is divided into three layers, the Application Layer where the applications for the users to control the SDN is there, the Control Plane where the controller is present that gives instructions and actions sets [2] to the physical device and the Data Plane where our project resides.

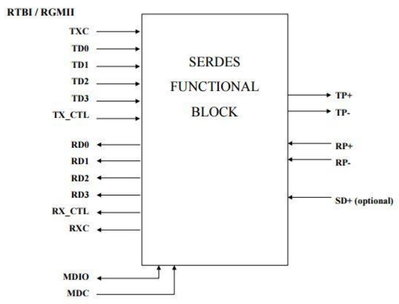
Overview of the working in a Data Plane:



The figure above shows what happens to a packet after it comes in an SDN switch and how it is processed inside the switch design and how it interacts with the Control Plane.

Modeling

RGMII Interface:



RGMII (Reduced Gigabit Media Independent Interface) [3] is a version of a MII interface. It is mainly used when transferring data from the PHY (Physical) device to a MAC (Media Access Controller) device. This interface is mainly used because it has less number of pins to deal with and also because it supports Gigabit speed of data transfer.

Here as shown TD0 - TD3 are the input signals and bring 4 bits of data on every clock cycle.

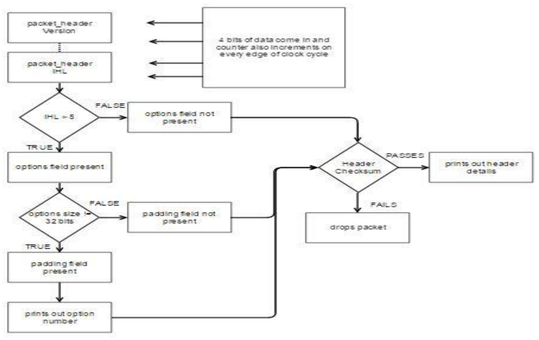
Similarly, RD0 – RD3 are the output signals that also work in the same fashion as the input bits but in the opposite direction. This interface is used to take the input data bits from the outside world.

Design Implementation

Key Points

- Runs on both the posedge and negedge of clock.
- Writing to Structure defined in System Verilog [4].
- Test Data for a data packet from Wireshark.

Design Flow Diagram



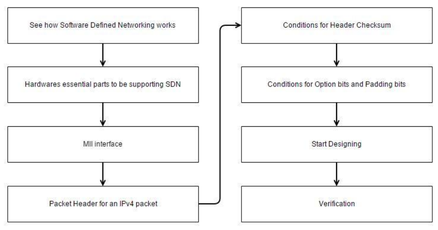
As can be seen, 4 bits if data are taken in every edge of clock and these bits of data are saved inside the fields of the defined structure. Also there is a counter implemented to guide the data bits to their appropriate fields.

Conditions on length of IHL field [5] i.e. whether option field is present or not is provided and depending on the output of this condition, memory for the options field and the padding field is decided. All the header bits are stored in the structure.

Checksum [5] is checked for the value and also the value of Time To Live field in the structure is to be checked. If either of them fail the condition, the packet is immediately dropped. In this condition only the checksum value is displayed.

If they pass these conditions the values of the packet header in binary format are displayed.

Design Approach



As can be seen, I started with researching the topics on Software Defined Networking, Open Flow Protocol that defined the necessary features of a hardware device that supports SDN.

After that research on how data will come in the design depending on the how it is implemented on a general networking switch was done and information on the MII and the RGMII interface were found.

How the packet data is transferred through and what are the essential conditions for a packet parser were determined. Depending on these conditions, code for the checksum was developed and then the design in System Verilog was started. To verify the design, multiple test bench were created each for a different test case.

Results

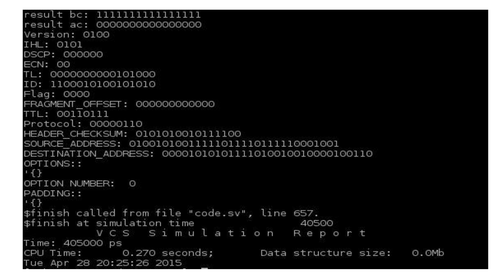
No Options Field

As can be seen from the terminal, when there are no options field present and also the checksum value is correct, the Header Fields are displayed correctly on the screen

Wrong Checksum

As can be seen from the terminal output, when the checksum does not pass the value the packet header fields are not displayed. Only the incorrect checksum value is displayed on screen. In the same way when the Time To Live field goes 0 after decrementing, the output screen only shows the checksum value and does not print the Network Packet Header fields.

Options Field present



If the options field is present and it passes checksum and TTL then the option number will be displayed.

Conclusion

We have presented a design implementation of the Networking Switch's Packet Parser for an SDN Switch using System Verilog. And the output displays consistent behavior depending on the packet format and the error bits. This design supports IPv4 packet format only.

Future Modifications

This design can be further extended to support IPv6 packets as it will be essential in the near future. Conditions depending on the fields of the IPv6 packet header can be added and design can be made to work in either IPv4 or IPv6 formats.

Key References

[1] Exploring Software Defined Networking with Brocade white paper.

[2] Open Flow Switch Specification Version 1.4.0.

[3] Reduced Gigabit Media Independent Interface (RGMII) Version 1.3.

[4] System Verilog 3.1a Language Reference Manual.

[5] Computer Networks, 5th Edition, Andrew S. Tanenbaum, Prentice-Hall, 2011.

Acknowledgments

I thank Professor Morris Jones for providing invaluable expertise on the topic and his precious time.

For further information

Please contact mitmehta15*@gmail.com* for the System Verilog code, Verification Test Benches and Simulation Results. Available upon request. For expertise on the topic contact me on (408) 813-6463.