**Packet Buffer**

In the preprocessing unit, every packet it reads will be sent to this buffer without doing any modifications. If a table miss is occurred, the relevant packet will be taken from this buffer. If the packet matches with an entry in the flow table, then there is no need to keep that packet in this buffer any more, therefore in that case, the packet can be removed from the buffer.

I implemented a FIFO queue using an array for the packet buffer. Buffer size can be changed by changing **PACKET\_BUFFER\_LEN**. The first byte before a packet has the length of the packet followed by that byte. Therefore, for every packet an extra byte is kept in front of that packet to identify the packet size. I kept the array that is used as the buffer, global in the header file. So that it can be accessed from anywhere. And there are two other variables **buffer\_full\_flag** and **buffer\_empty\_flag** keeping the status of the buffer.

When implementing this buffer in the code, first it has to be initialized using the **initialize\_packet\_buffer()** function. Then **add\_a\_byte\_to\_the\_packet\_buffer(uint8\_t byte)** function is used to add a byte to the buffer. In the preprocessing unit, this buffer is filled by reading byte by byte. **dqueue\_from\_packet\_buffer()** function returns the firstly inputted byte to the buffer. **print\_buffer()** will print the current buffer.

I have implemented two functions to read a packet from the buffer at once. **read\_a\_packet\_and\_pass\_a\_pointer()** returns the pointer of the oldest packet of the buffer and also, this will remove that packet from the packet buffer. The first byte that this pointer points at, contains the length of the packet. Therefore when reading the packet number of bytes equal to the packet length, starting from the byte pointed by the pointer has to be read.

**read\_a\_packet(uint8\_t \* pk)** function does the same thing as the **read\_a\_packet\_and\_pass\_a\_pointer()** The differenceis for **read\_a\_packet** function we have to give a pointer as a parameter, and the packet will be stored starting at that given address.

**read\_and\_print\_packet()** function will read the oldest packet in the buffer, remove it from the buffer and print that packet. Once the starting address of a received packet (The first byte contains the length) is given to **print\_packet(uint8\_t \* pkt)** function, it will print that packet.