Vlib Buffer

Michael

2017

The abstract of Dave's mail VPP16.09

Vlib_buffer_t's are embedded in private headroom space, tangent to the packet DMA target [which starts at (struct mbuf *)mb->data_off].

Here's how we create the mempools:

```
rmp = rte_pktmbuf_pool_create ((char *) pool_name,   /* pool name */
            num_mbufs,     /* number of mbufs */
            512, /* cache size */
            VLIB_BUFFER_HDR_SIZE,    /* priv size */
            VLIB_BUFFER_PRE_DATA_SIZE + VLIB_BUFFER_DATA_SIZE,    /* dataroom size */
            socket_id);   /* cpu socket */
```

As Chris wrote, b0->data is an array, not a pointer:

```
u8 data[0]; /**< Packet data. Hardware DMA here */
```

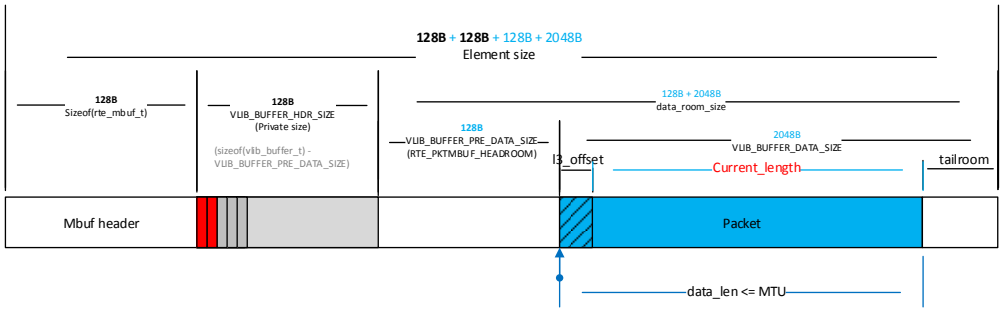
Q: How to manipulate packet payload in VPP?

See vlib_buffer_advance(...). Depending on what you want to do - which you didn?t share - that may be all you need.
In any event, these buffer metadata fields are involved:
current_data, current_length, total_length_not_including_first_buffer, next_buffer, flags

Take a careful look at .../src/vlib/{buffer.h,buffer_funcs.h} to see usage patterns.

DPDK rte_mbuf

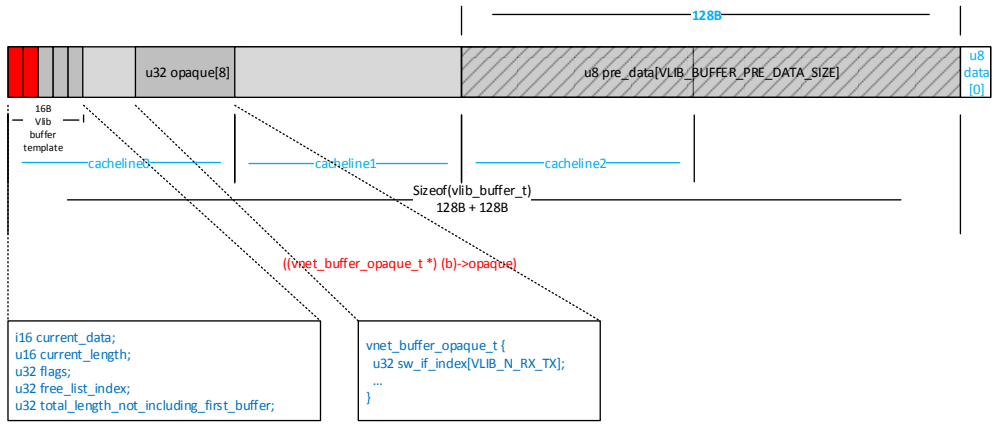
VPP16.09




```
rmp = rte_pktmbuf_pool_create ((char *) pool_name,   /* pool name */
            num_mbufs,    /* number of mbufs */
            512, /* cache size */
            VLIB_BUFFER_HDR_SIZE,    /* priv size */
            VLIB_BUFFER_PRE_DATA_SIZE + VLIB_BUFFER_DATA_SIZE,    /* dataroom size */
            socket_id);   /* cpu socket */
```

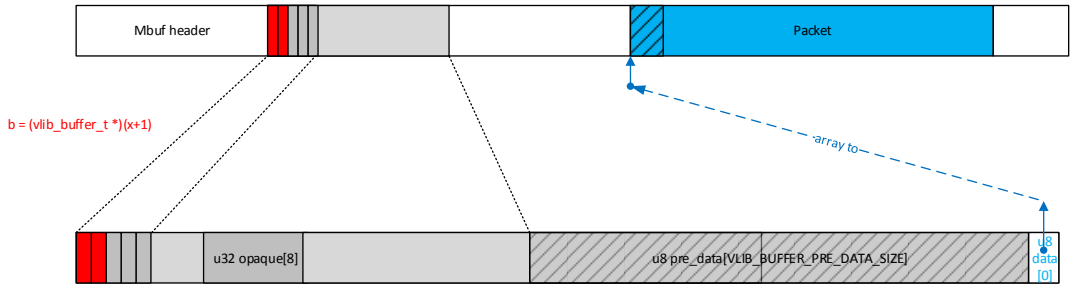

vlib_buffer_t

VPP16.09



vlib_buffer_from_rte_mbuf

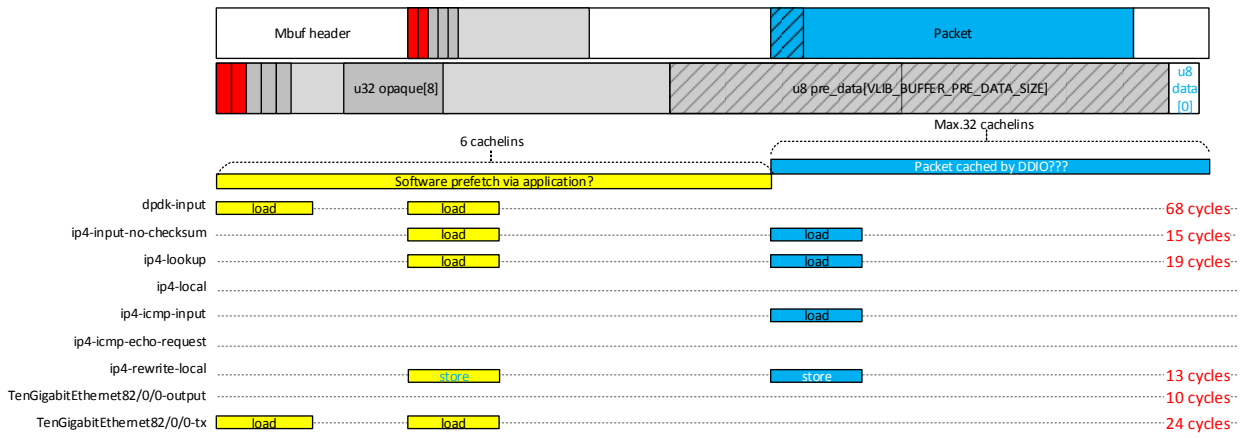
VPP16.09




```
b0->error = node->errors[error0];

l3_offset0 = ((next0 == DPDK_RX_NEXT_IP4_INPUT ||
                        next0 == DPDK_RX_NEXT_IP6_INPUT ||
                        next0 == DPDK_RX_NEXT_MPLS_INPUT) ?
                       sizeof (ethernet_header_t) : 0);

b0->current_data = l3_offset0;
/* Some drivers like fm10k receive frames with
  mb->data_off > RTE_PKTMBUF_HEADROOM */
b0->current_data += mb->data_off - RTE_PKTMBUF_HEADROOM;
b0->current_length = mb->data_len - l3_offset0;

b0->flags = buffer_flags_template;

if (VMWARE_LENGTH_BUG_WORKAROUND)
 b0->current_length -= 4;

vnet_buffer (b0)->sw_if_index[VLIB_RX] = xd->vlib_sw_if_index;
vnet_buffer (b0)->sw_if_index[VLIB_TX] = (u32) ~ 0;
n_rx_bytes += mb->pkt_len;
```


DDIO and Software Prefetch

VPP16.09



Level	size	Latency	IndexBits	Sets	Associativity	LineSize
L1Data	32KiB	4	6	64	8	64B
L1Instructions	32KiB	4	6	64	8	64B
L2	256KiB	7	9	512	8	64B
LLC	≥3MiB	26-31	11	≥4096	12-20	64B

Typical cache parameters in Intel processors