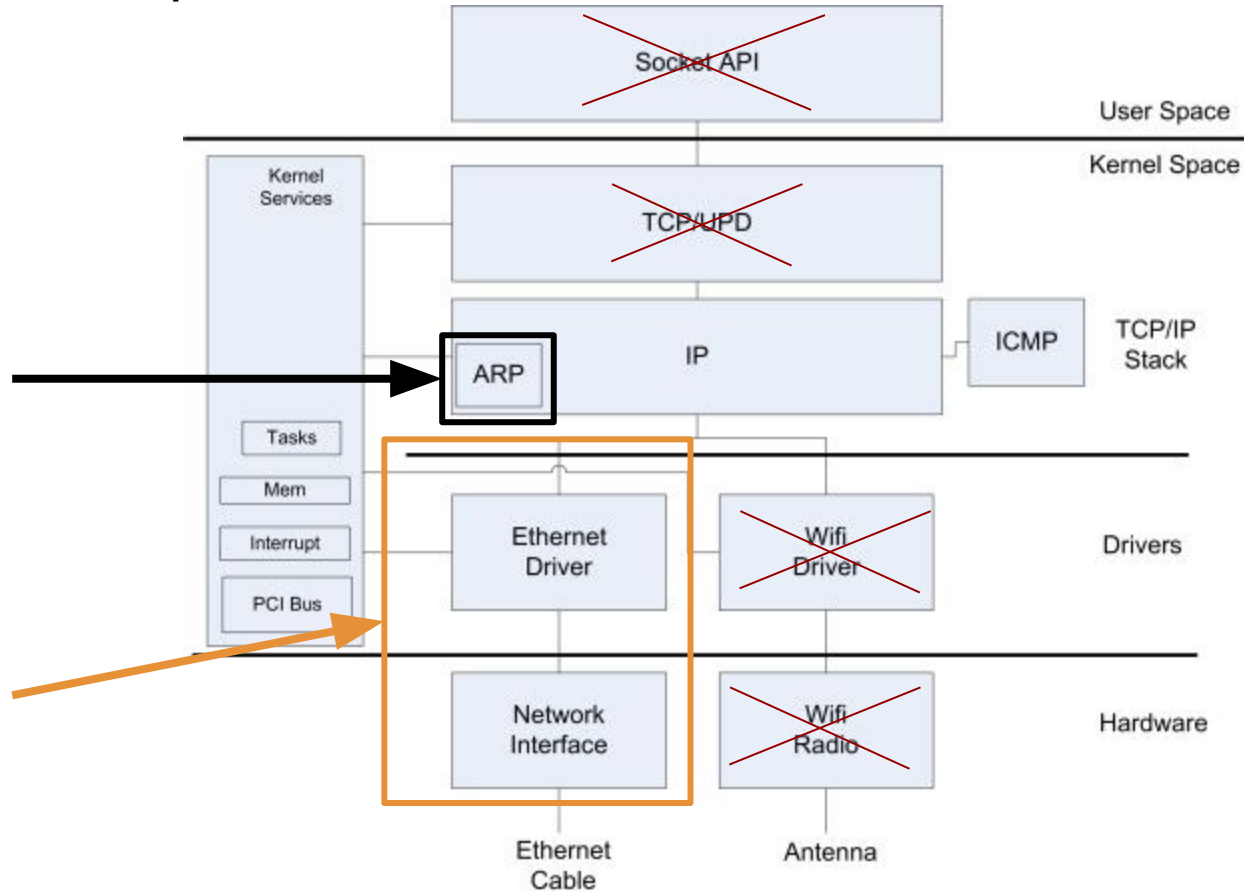


Project Kraken

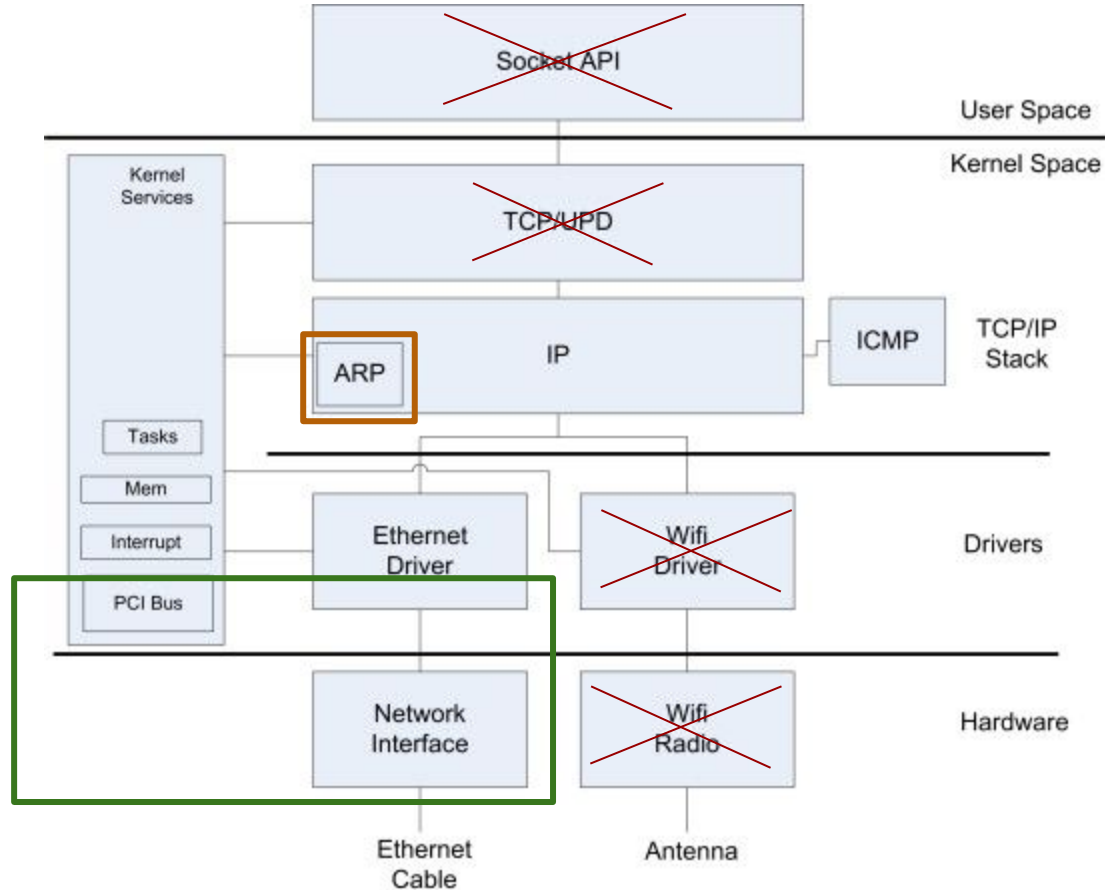
A Network Stack for Ninjastorms OS

Recap - What was the Scope?

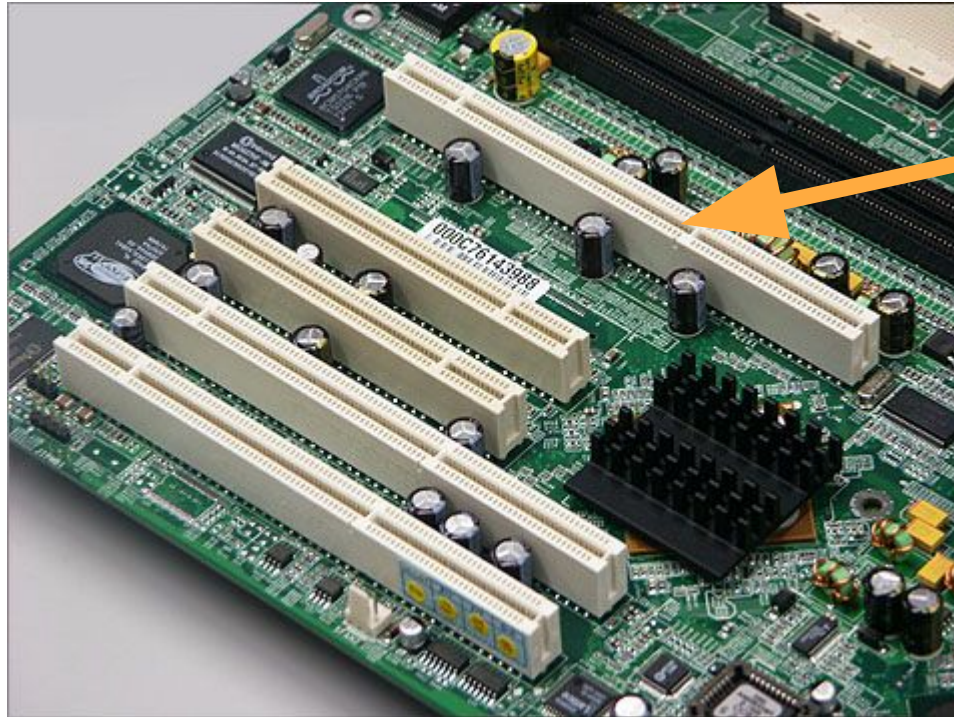


What did we achieve?

- Talk to PCI device
- Talk to network card via PCI
- Give it an address (MAC)
- Send & receive packets via NIC (not yet on layer 2)
- sending rudimentary ARP requests



How to talk to a device?



PCI slot

PCI Enumeration

What are you?



31		16 15		0		
Device ID		Vendor ID		00h		
Status		Command		04h		
Class Code			Revision ID			08h
BIST	Header Type	Lat. Timer	Cache Line S.			0Ch
Base Address Registers						10h
						14h
						18h
						1Ch
						20h
Cardbus CIS Pointer						24h
Subsystem ID			Subsystem Vendor ID			28h
Expansion ROM Base Address						2Ch
Reserved				Cap. Pointer		30h
Reserved						34h
Max Lat.	Min Gnt.	Interrupt Pin	Interrupt Line			38h
						3Ch

PCI Enumeration

What are you?

Who created you?

31		16 15		0	
Device ID		Vendor ID			00h
Status		Command			04h
Class Code			Revision ID		08h
BIST	Header Type	Lat. Timer	Cache Line S.		0Ch
Base Address Registers					10h 14h 18h 1Ch 20h 24h
Cardbus CIS Pointer					28h
Subsystem ID		Subsystem Vendor ID			2Ch
Expansion ROM Base Address					30h
Reserved			Cap. Pointer		34h
Reserved					38h
Max Lat.	Min Gnt.	Interrupt Pin	Interrupt Line		3Ch



PCI Enumeration

What are you?

Who created you?

Tells the size it needs.
Lets us define where
that is.

31	16	15	0	00h
Device ID		Vendor ID		
Status		Command		04h
Class Code			Revision ID	08h
BIST	Header Type	Lat. Timer	Cache Line S.	0Ch
Base Address Registers				10h 14h 18h 1Ch 20h 24h
Cardbus CIS Pointer				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM Base Address				30h
Reserved			Cap. Pointer	34h
Reserved				38h
Max Lat.	Min Gnt.	Interrupt Pin	Interrupt Line	3Ch

How To: Sending a packet

```
int
send_packet(const void * p_data, uint16_t p_len)
{
    printf("[E1000] Sending packet data <%s> with len %i\n", p_data, p_len);
    e1000_tx_desc_t *curr = &e1000→tx_descs[e1000→tx_cur];
    curr→addr = (uint64_t)p_data;
    curr→length = p_len;
    curr→cmd = CMD_EOP | CMD_IFCS | CMD_RS;
    curr→status = 0;
    uint8_t old_cur = e1000→tx_cur;
    e1000→tx_cur = (e1000→tx_cur + 1) % E1000_NUM_TX_DESC;

    writeCommand(REG_TXDESCTAIL, e1000→tx_cur);

    while(!(curr→status));
    printf("[E1000] Packet send!\n");
    return 0;
}
```

1. Build
packet

How To: Sending a packet

```
int
send_packet(const void * p_data, uint16_t p_len)
{
    printf("[E1000] Sending packet data <%s> with len %i\n", p_data, p_len);
    e1000_tx_desc_t *curr = &e1000->tx_descs[e1000->tx_cur];
    curr->addr = (uint64_t)p_data;
    curr->length = p_len;
    curr->cmd = CMD_EOP | CMD_IFCS | CMD_RS;
    curr->status = 0;
    uint8_t old_cur = e1000->tx_cur;
    e1000->tx_cur = (e1000->tx_cur + 1) % E1000_NUM_TX_DESC;

    writeCommand(REG_TXDESCTAIL, e1000->tx_cur);

    while(!(curr->status));
    printf("[E1000] Packet send!\n");
    return 0;
}
```

1. Build
packet

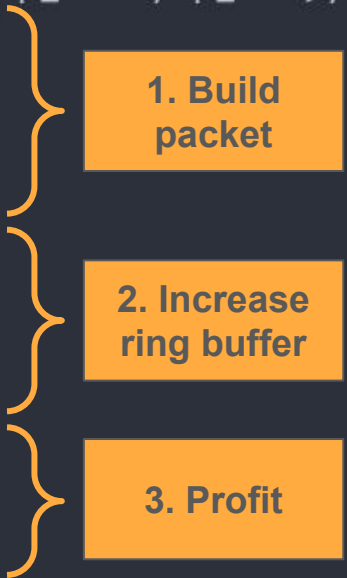
2. Increase
ring buffer

How To: Sending a packet

```
int
send_packet(const void * p_data, uint16_t p_len)
{
    printf("[E1000] Sending packet data <%s> with len %i\n", p_data, p_len);
    e1000_tx_desc_t *curr = &e1000->tx_descs[e1000->tx_cur];
    curr->addr = (uint64_t)p_data;
    curr->length = p_len;
    curr->cmd = CMD_EOP | CMD_IFCS | CMD_RS;
    curr->status = 0;
    uint8_t old_cur = e1000->tx_cur;
    e1000->tx_cur = (e1000->tx_cur + 1) % E1000_NUM_TX_DESC;

    writeCommand(REG_TXDESCTAIL, e1000->tx_cur);

    while(!(curr->status));
    printf("[E1000] Packet send!\n");
    return 0;
}
```



The diagram consists of three orange boxes on the right side of the code block, each connected to a specific section of the code by a yellow curly bracket. The first box, labeled '1. Build packet', is connected to the lines where the packet is prepared (setting address, length, command, and status). The second box, labeled '2. Increase ring buffer', is connected to the lines where the ring buffer index is updated and the hardware register is written. The third box, labeled '3. Profit', is connected to the final while loop and return statement.

1. Build packet
2. Increase ring buffer
3. Profit

First packet send!

```
[E1000] Sending packet data <Hello Ninja!> with len 12
[E1000] Packet send!
All done. ninjastorms out!
qemu-system-arm: terminating on signal 15 from pid 17124 (/bin/zsh)
~/Workspaces/ninjastorms feature/e1000_driver !3 ?3
└─$ cat net_dump.dat
000360^+

Hello Ninja!%
```

First ARP request!

```
static void
network_test(void)
{
    // ARP request: Who has 10.0.2.15? Tell 10.0.2.10
    const char * data = "\xff\xff\xff\xff\xff\xff\x52\x54\x00\x12\x34\x56\x08\x06\x00\x01" \
                        "\x08\x00\x06\x04\x00\x01\x52\x54\x00\x12\x34\x56\x0a\x00\x02\x0a" \
                        "\x00\x00\x00\x00\x00\x00\x0a\x00\x02\x0f";
    for(int i = 0; i < 50; i++)
    {
        send_packet(data, 42);
        // wait some time before next packet
        for (int j = 0; j < 100000000; ++j);
    }
}
```

Sender MAC address
52 : 54: 00: 12: 34: 56

10 . 00 .02 . 15
Target IP

10 . 00 .02 . 10
Sender IP

First ARP request!

```
static void
network_test(void)
{
    // ARP request: Who has 10.0.2.15? Tell 10.0.2.10
    const char * data = "\xff\xff\xff\xff\xff\xff\x52\x54\x00\x12\x34\x56\x08\x06\x00\x01" \
                        "\x08\x00\x06\x04\x00\x01\x52\x54\x00\x12\x34\x56\x0a\x00\x02\x0a" \
                        "\x00\x00\x00\x00\x00\x00\x0a\x00\x02\x0f";
    for(int i = 0; i < 50; i++)
    {
        send_packet(data, 42);
        // wait some time before next packet
        for (int j = 0; j < 100000000; ++j);
    }
}
```

1	0.000000000	RealtekU_12:34:56	Broadcast	ARP	42 Who has 10.0.2.15? Tell 10.0.2.10
2	0.000015835	f6:09:3d:ac:44:46	RealtekU_12:34:56	ARP	42 10.0.2.15 is at f6:09:3d:ac:44:46
3	0.553173395	RealtekU_12:34:56	Broadcast	ARP	42 Who has 10.0.2.15? Tell 10.0.2.10
4	0.553183322	f6:09:3d:ac:44:46	RealtekU_12:34:56	ARP	42 10.0.2.15 is at f6:09:3d:ac:44:46

Next Steps

- Use Ethernet instead of raw PDUs

- Implement ARP with ARP table and extraction of information from frames

