Embedded Networking Software Development Lab 3 – Networking Layer 2 (Link Layer) and Layer 1 (Physical layer)

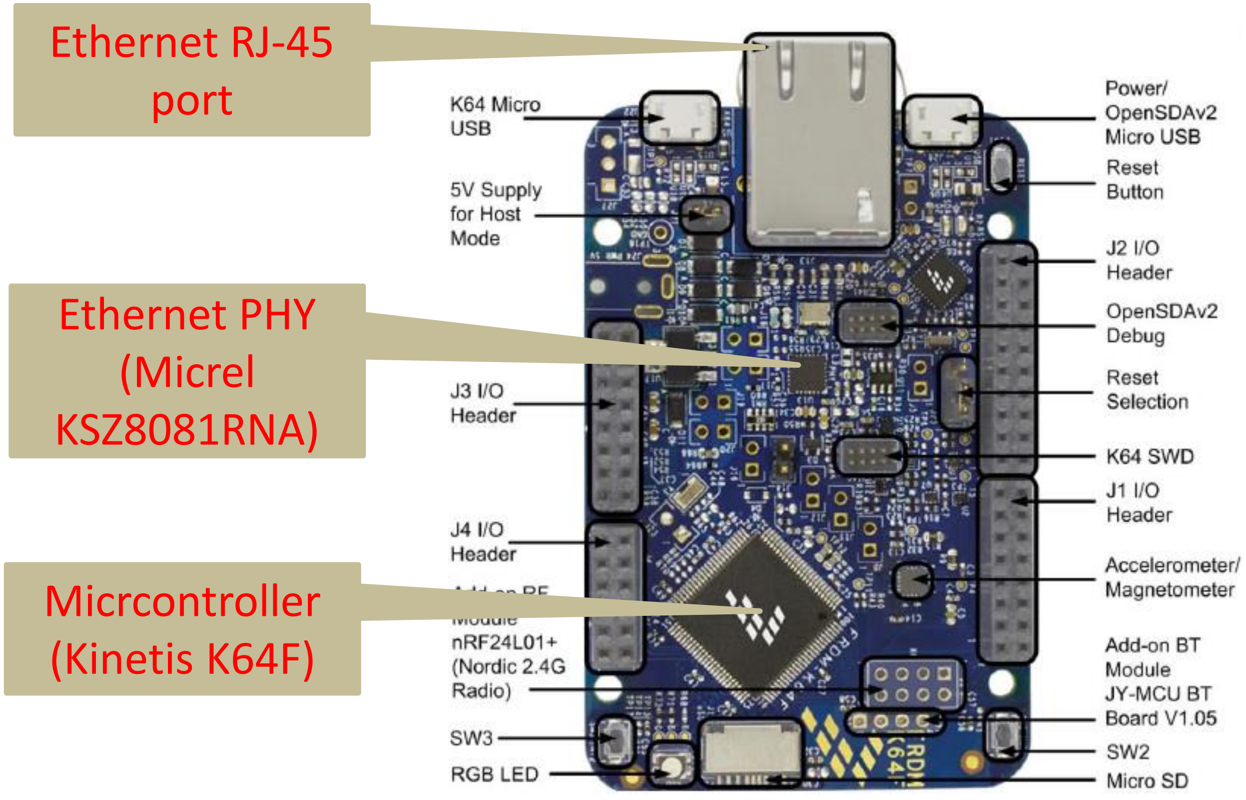
Version 0.1

Overview

- Purpose
  - Learn how the ARP protocol works and to get exposed to Ethernet as a layer-2 networking technology.
- Learning objectives
  - To become familiar with the packet format of the ARP protocol
  - To become familiar with the Ethernet frame format
  - To learn how to detect if there is physical Ethernet connectivity (physical Ethernet link is up or down)

Overview (2)

- The FRDM-K64F board



Overview (3)

- In lab2, we used Wireshark to examine the “ping” ICMP packets (shown in pink below):

When a network interface is “turned on”, a gratuitous ARP request is sent for its local IPv4 address, to detect if someone else is using the same IPv4 address

540 1…	16:04:90:11:00:29	Broadcast	ARP	60	Gratuitous ARP for 192.168.8.2 (Request)
541 1…	16:04:90:11:00:29	Broadcast	ARP	60	Who has 192.168.8.1? Tell 192.168.8.2
542 1…	Dell_01:f1:7d	16:04:90:11:00:29	ARP	42	192.168.8.1 is at f0:1f:af:01:f1:7d
543 1…	192.168.8.2	192.168.8.1	ICMP	60	Echo (ping) request id=0x789a, seq=0/0, ttl=64 (reply in 546)
544 1…	Dell_01:f1:7d	Broadcast	ARP	42	Who has 192.168.8.2? Tell 192.168.8.1
545 1…	16:04:90:11:00:29	Dell_01:f1:7d	ARP	60	192.168.8.2 is at 16:04:90:11:00:29
546 1…	192.168.8.1	192.168.8.2	ICMP	42	Echo (ping) reply id=0x789a, seq=0/0, ttl=128 (request in 543)
547 1…	192.168.8.2	192.168.8.1	ICMP	60	Echo (ping) request id=0x789a, seq=256/1, ttl=64 (reply in 548)
548 1…	192.168.8.1	192.168.8.2	ICMP	42	Echo (ping) reply id=0x789a, seq=256/1, ttl=128 (request in 547)
549 1…	[illegible]	[illegible]	[illegible]	[illegible]	Echo (ping) request id[illegible]
550 1…	[illegible]	[illegible]	[illegible]	[illegible]	Echo (ping) reply id[illegible]
551 1…	[illegible]	[illegible]	[illegible]	[illegible]	Echo (ping) request id[illegible]
552 1…	[illegible]	[illegible]	[illegible]	[illegible]	Echo (ping) reply id[illegible]
553 1…	192.168.8.2	192.168.8.1	ICMP	60	Echo (ping) request id=0x789a, seq=1024/4, ttl=64 (reply in 554)
554 1…	192.168.8.1	192.168.8.2	ICMP	42	Echo (ping) reply id=0x789a, seq=1024/4, ttl=128 (request in 553)
555 1…	192.168.8.2	192.168.8.1	ICMP	60	Echo (ping) request id=0x789a, seq=1280/5, ttl=64 (reply in 556)
556 1…	192.168.8.1	192.168.8.2	ICMP	42	Echo (ping) reply id=0x789a, seq=1280/5, ttl=128 (request in 555)
557 1…	192.168.8.2	192.168.8.1	ICMP	60	Echo (ping) request id=0x789a, seq=1536/6, ttl=64 (reply in 558)
558 1…	192.168.8.1	192.168.8.2	ICMP	42	Echo (ping) reply id=0x789a, seq=1536/6, ttl=128 (request in 557)
559 1…	192.168.8.2	192.168.8.1	ICMP	60	Echo (ping) request id=0x789a, seq=1792/7, ttl=64 (reply in 560)
560 1…	192.168.8.1	192.168.8.2	ICMP	42	Echo (ping) reply id=0x789a, seq=1792/7, ttl=128 (request in 559)

When sending a packet for the first time to a new node, an ARP request is sent to all nodes in the same subnet, to see who has the target IP address

The target node replies with an “ARP reply” message to the requesting node. The ARP reply contains the local MAC address of the node sending the reply.

Background Information

- To be transmitted over the network, IPv4 (and IPv6) packets are encapsulated in layer-2 packets.
  - In the case of Ethernet, layer-2 packets are known as *Ethernet frames*
- Also, layer-3 node addresses need to be mapped to layer-2 node addresses.
  - In the case of Ethernet, layer-2 addresses are known as *MAC (Media Access Control) addresses*, and nodes as known as Ethernet stations.
  - The Address Resolution Protocol (ARP) maps IPv4 addresses to layer-2 addresses.
    - It is a layer 2.5 protocol, as it has “one foot on layer 3 and one foot on layer 2”.

Background Information (2)

- The ARP Protocol has two basic message types:
  - ARP Request
    - Carried in a layer-2 broadcast packet
    - Its purpose is to ask other nodes in the same layer-2 network (layer-3 subnet) “who has a given IPv4” address.
    - There is a special ARP request known as a “Gratuitous ARP request”, used to detect duplicated IPv4 addresses in a subnet.
  - ARP Reply
    - Carried in a layer-2 unicast packet
    - Its purpose is to answer an ARP request to the requesting node, by providing the layer-2 address of the node that has the given IPv4 address (contained in the ARP request)

Background Information (3)

- The ARP Protocol basic flow:
  - When an IPv4 node's network interface is "turned on", or its IP address is changed, the networking stack sends a gratuitous ARP request, to make sure that no other nodes (in the same subnet) have the same IP address.
    - No ARP reply should be received, unless there are two or more nodes with the same IPv4 address, which is a network configuration error.
  - When an IPv4 node sends an IPv4 packet to another node, for which its layer-2 address is not known, the networking stack sends an ARP request, to obtain the target node's layer-2 address, before it can send the IPv4 packet.
  - When an IPv4 node receives an ARP request for its own IPv4 address, the networking stack sends an ARP reply to the requesting node.

Background Information (4)

- The ARP Protocol basic flow (cont.):
  - When the node that sent the ARP request receives an ARP reply for the target IPv4 address, it saves it in a table known as the ARP cache, and sends the pending IPv4 packet(s).
    - Subsequent sends of IPv4 packets will get the target node's layer-2 address from the ARP cache instead of sending ARP requests.
    - Entries in the ARP cache expire after 20 minutes, by default. After that, the next packet send will trigger a new ARP request.

Background Information (5)

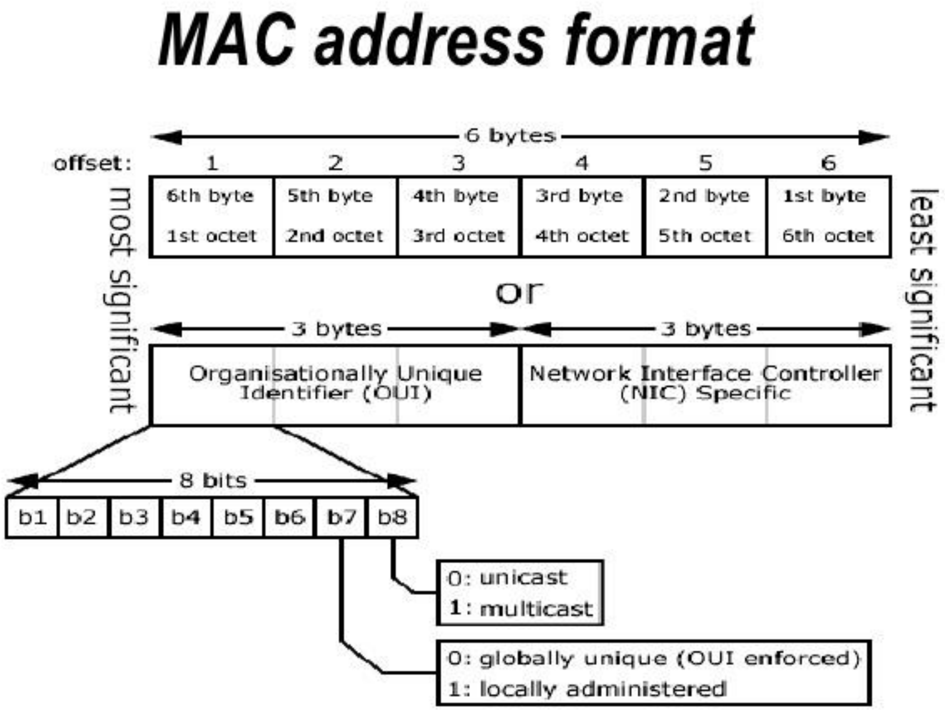
- Fields of an ARP message:
  - Link address type
    - 0x1 for Ethernet MAC addresses
  - Network address type
    - 0x800 for IPv4 addresses
  - Link address size (in bytes)
    - 6 for Ethernet MAC addresses
  - Network address size (in bytes)
    - 4 for IPv4 addresses
  - ARP operation (ARP_REQUEST or ARP_REPLY)
    - 16-bit value in network byte order
    - 0x0001 – ARP request
    - 0x0002 – ARP reply

Background Information (6)

- Fields of an ARP message (cont.):
  - Source (sender) link address
    - Sender’s MAC address in the case of Ethernet
  - Source (sender) network address
    - Sender’s IPv4 address
  - Destination (target) link address
    - Target’s MAC address in the case of Ethernet
  - Destination (target) network address
    - Target’s IPv4 address

Background Information (7)

- Ethernet MAC addresses



There are two special MAC addresses:

- Broadcast MAC address:
  ff:ff:ff:ff:ff:ff
- Null MACV address:
  00:00:00:00:00:00

Background Information (8)

- Ethernet Frames
  - Both IPv4 packets and ARP messages are carried in Ethernet frames
- Fields of an Ethernet frame
  - Destination MAC address
  - Source MAC address
  - Frame type
    - 0x0800 – for IPv4 packets
    - 0x0806 – for ARP messages
    - 0x8100 – for VLAM tagged frame
    - 0x86dd – for IPv6 packets
  - Data payload (e.g., IPv4 packet, ARP message, etc)
  - CRC

Background Information (9)

- Ethernet MAC
  - Media Access Controller
  - Implements a data-link layer (layer 2 of the OSI model).
  - A hardware module inside a microcontroller chip such as the NXP Kinetis K64F
- Ethernet PHY
  - Physical interface transceiver
  - Implements the physical layer (layer 1 of the OSI model)
  - A separate chip such as the Micrel KSZ8081RNA

Background Information (10)

- MII
  - Media Independent Interface
  - Hardware protocol for chip-to-chip communication between the Ethernet MAC (in the microcontroller) and the Ethernet PHY
  - Consists of a parallel data interface and a serial management interface
  - Data interface is full duplex (separate signals for Rx and Tx) and requires 16 pins in total
  - Requires a 25MHz clock
  - Supports data transfer rates up to 100 Mbits/s
  - Serial Management Interface (SMI) also known as MDIO, requires two pins.
    - Multiple PHYs can be connected to the same MDIO bus.
    - Each PHY on a MDIO bus has its own address

Background Information (11)

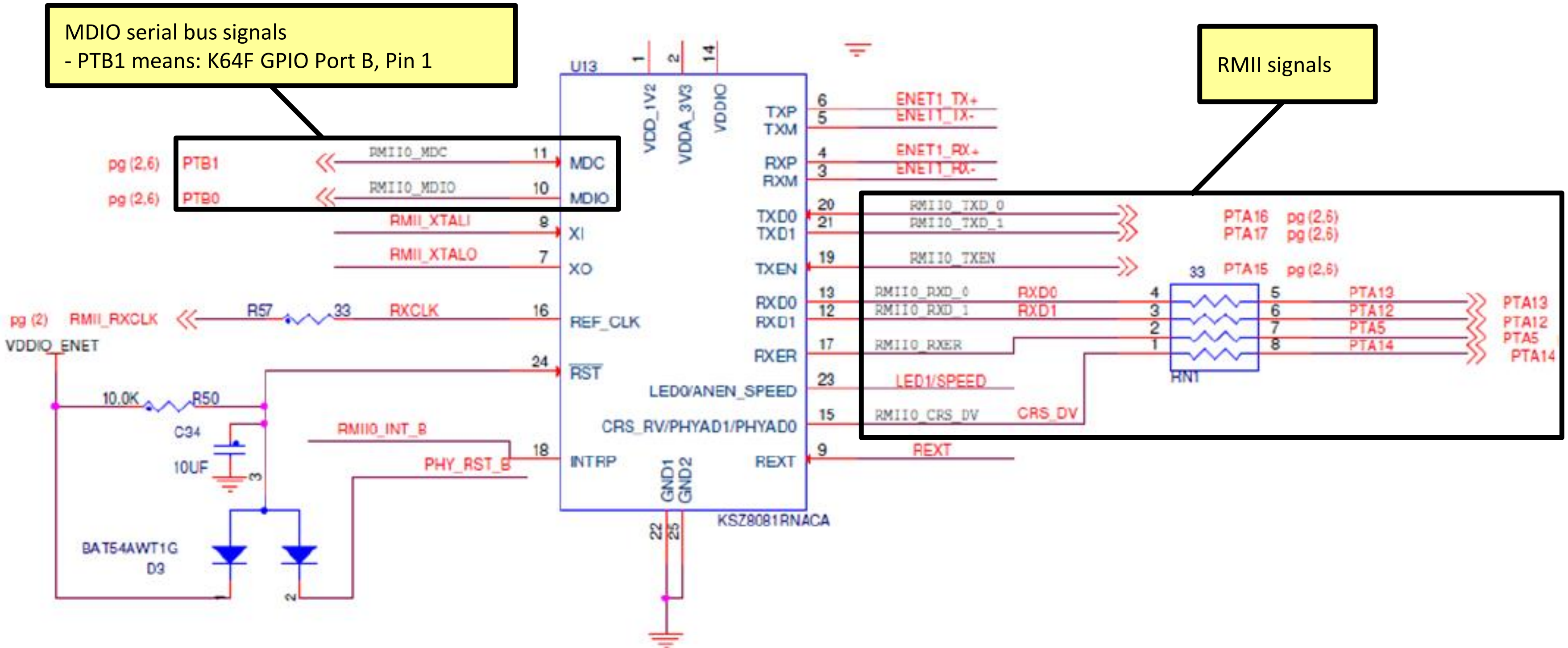
- RMII
  - Reduced pin-count MII
  - Data interface requires only 8 pins
  - Requires a 50MHz clock
- SMII
  - Serial MII
  - Data interface requires only 4 pins
  - Requires a 125MHz clock

Background Information (12)

- Higher speed MAC-to-PHY protocols
  - GMII - Gigabit MII: 1 Gbits/s
  - RGMII - Reduced pin-count GMII
  - SGMII - Serial GMII
  - QSGMII- Quad Serial GMII (4 SGMII lines combined): 5 Gbits/s
  - XGMII - 10 Gigabit MII: 10 Gbits/s

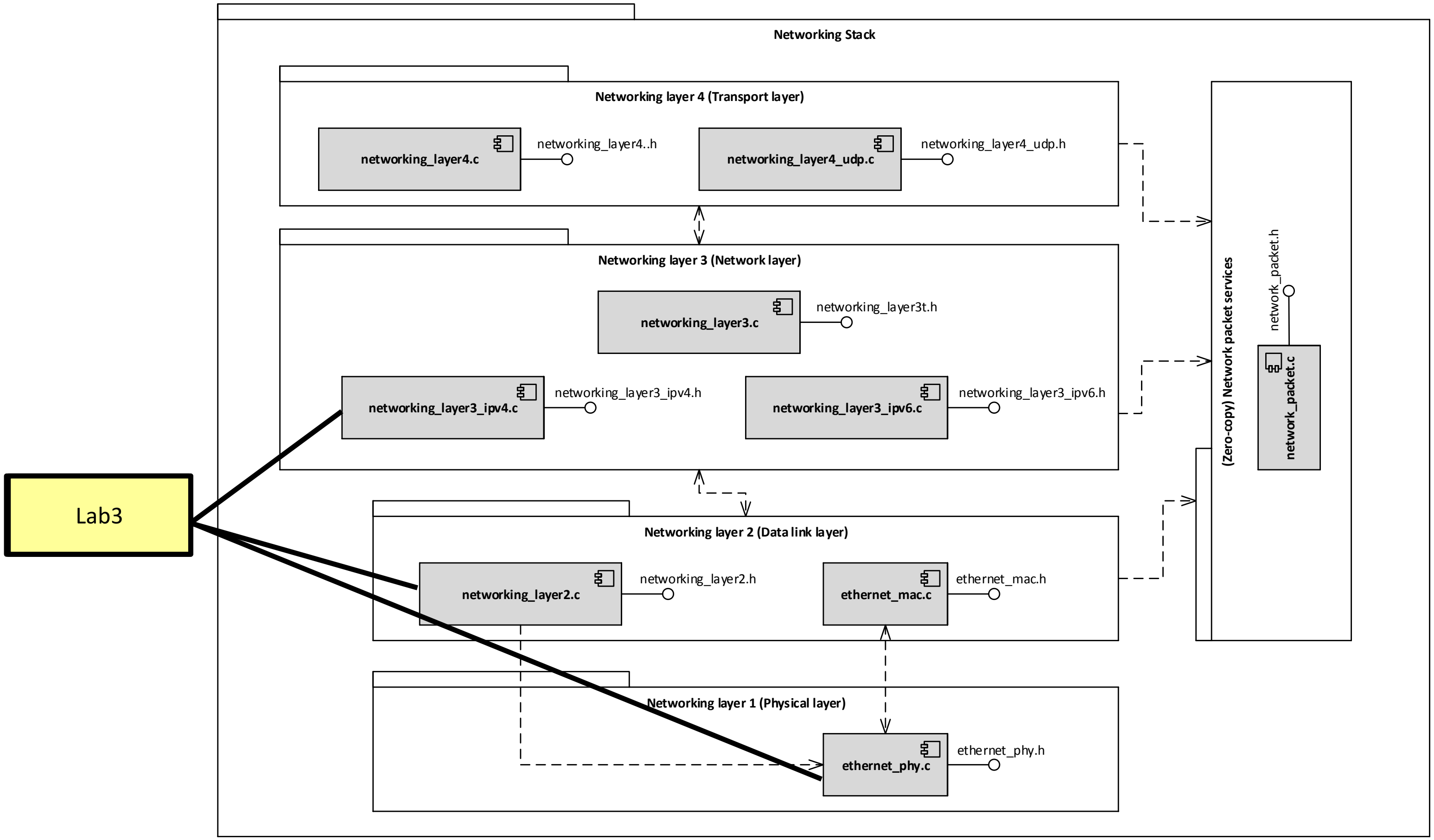
Background Information (13)

- Pin connections between the MCU (K64F chip) and the Micrel Ethernet PHY (KSZ8081RNA chip)



Lab Overview

- Networking Stack Architecture



Lab Overview (2)

- Instructions:
  - Implement the TODOs tagged with ‘TODO-lab3’ in the code of project lab3-layers2and1:
    - Complete function that send an ARP request
    - Complete function that sends an ARP reply
    - Complete function that sends an Ethernet frame
    - Configure the MDIO pins of the Ethernet PHY
    - Implement function to detect if the Ethernet physical link is up/down
    - Complete function that generates local MAC address for the FRDM-K64F’s Ethernet interface.
  - Answer question in the QUESTIONS.txt file

Test Plan

- Test your completed embedded networking stack on the FRDM-K64F board, by using the following test cases:
  - Test case 1: “ping” from the FRDM-K64F board to the PC
  - Test case 2: “ping” from the PC to the FRDM-K64F board
  - Run test cases from lab1 to verify that there are no regressions
- Test Environment Configuration:
  - Connect the Ethernet port of the board to the Ethernet port of your PC
  - Configure a static IPv4 address for the PC’s wired Ethernet interface that is in the 192.168.8.0/24 subnet (e.g., 192.168.8.1)

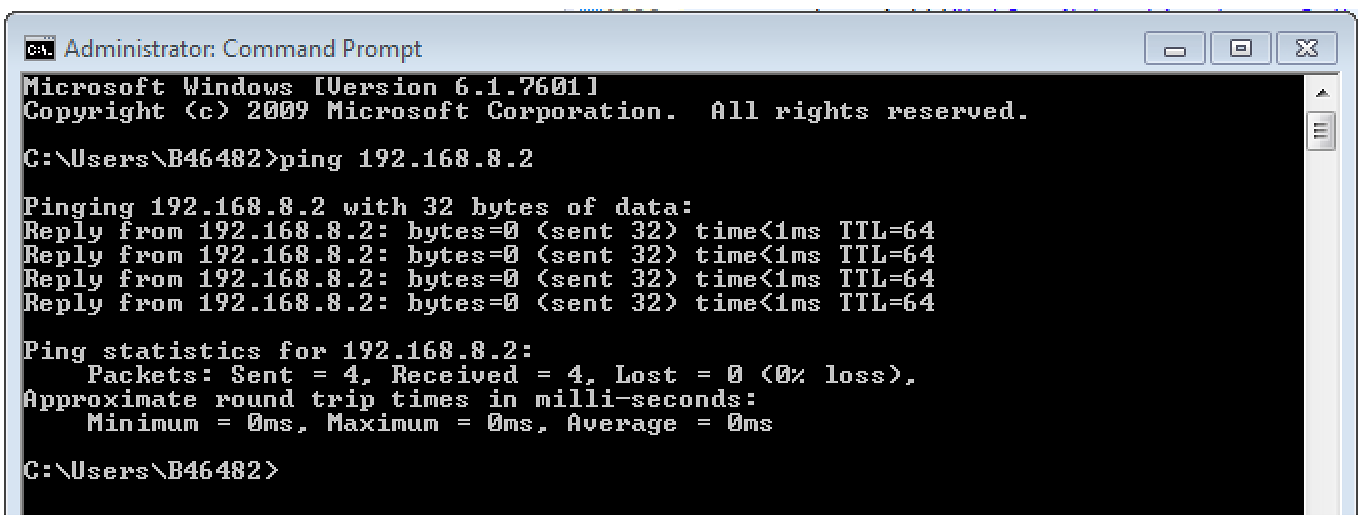
Test Plan (2)

- Test Case 1: Ping from the FRDM-K64F board to the PC
  - Run the “ping” command from the FRDM-K64F board to the PC, from the TeraTerm window:

```
COM4:115200baud - Tera Term VT
File  Edit  Setup  Control  Window  Help
Lab2 - Networking Layer 3 (built Feb 29 2016 09:31:43)
Reference solution

Ethernet link [up]   Ethernet MAC address [16:04:90:11:00:29]

IPv4 address [192.168.8.2]   IPv4 subnet mask [255.255.255.0]

Received packets accepted at layer 2 - Enet [145]  Received packets dropped at layer 2 - Enet [   ]  Sent packets at layer 2 - Enet [11]

Received packets accepted at layer 3 - IPv4 [140]  Received packets dropped at layer 3 - IPv4 [   ]  Sent packets at layer 3 - IPv4 [9]

Received packets accepted at layer 4 - UDP [   ]  Received packets dropped at layer 4 - UDP [132]  Sent packets at layer 4 - UDP [1]

Last UDP message received [   ]

lab2> ping 192.168.8.1

Ping 0 replied by 192.168.8.1
Ping 1 replied by 192.168.8.1
Ping 2 replied by 192.168.8.1
Ping 3 replied by 192.168.8.1
Ping 4 replied by 192.168.8.1
Ping 5 replied by 192.168.8.1
Ping 6 replied by 192.168.8.1
Ping 7 replied by 192.168.8.1
lab2>
```

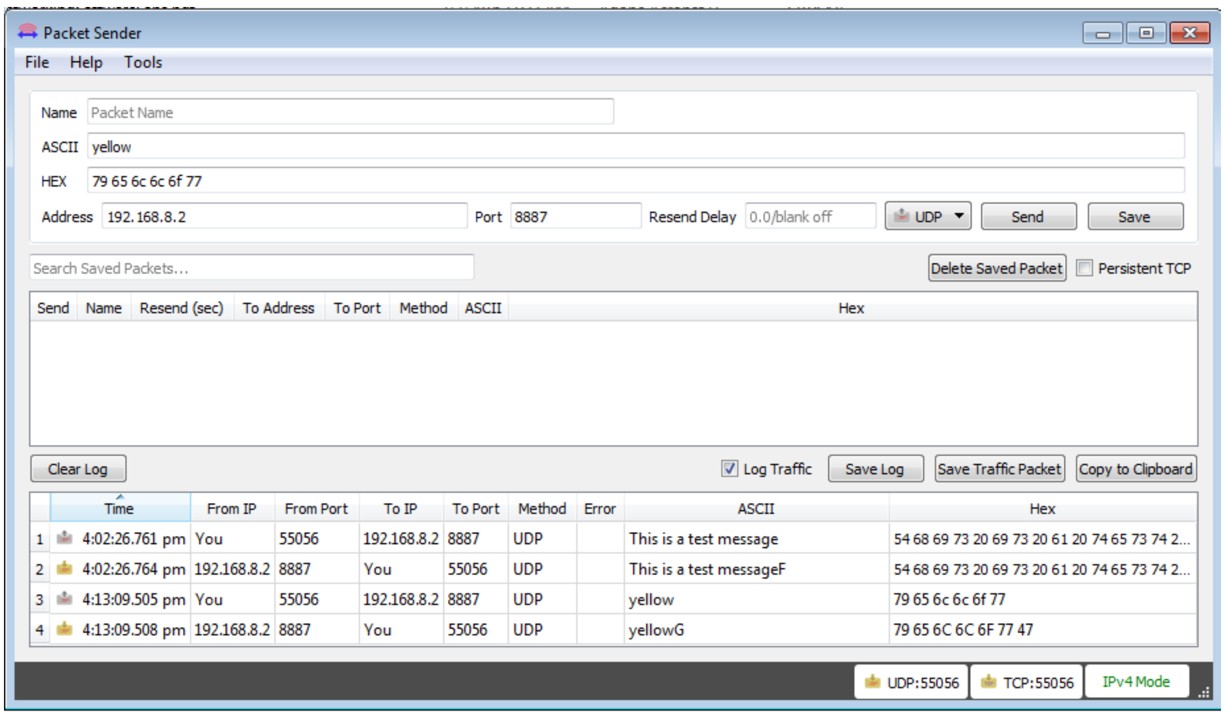
Test Plan (3)

- Test Case 2: Ping from the PC to the FRDM-K64F board
  - Run the “ping” command from the PC to the FRDM-K64F board, from a Command prompt window or a PowerShell window:



Test Plan (4)

- Test Case 3: Command messages
  - Send “command” messages to the board to change the color of the blinking LED. Valid commands are the strings: “red”, “green”, “blue”, “yellow”, “cyan”, “magenta” and “white”.

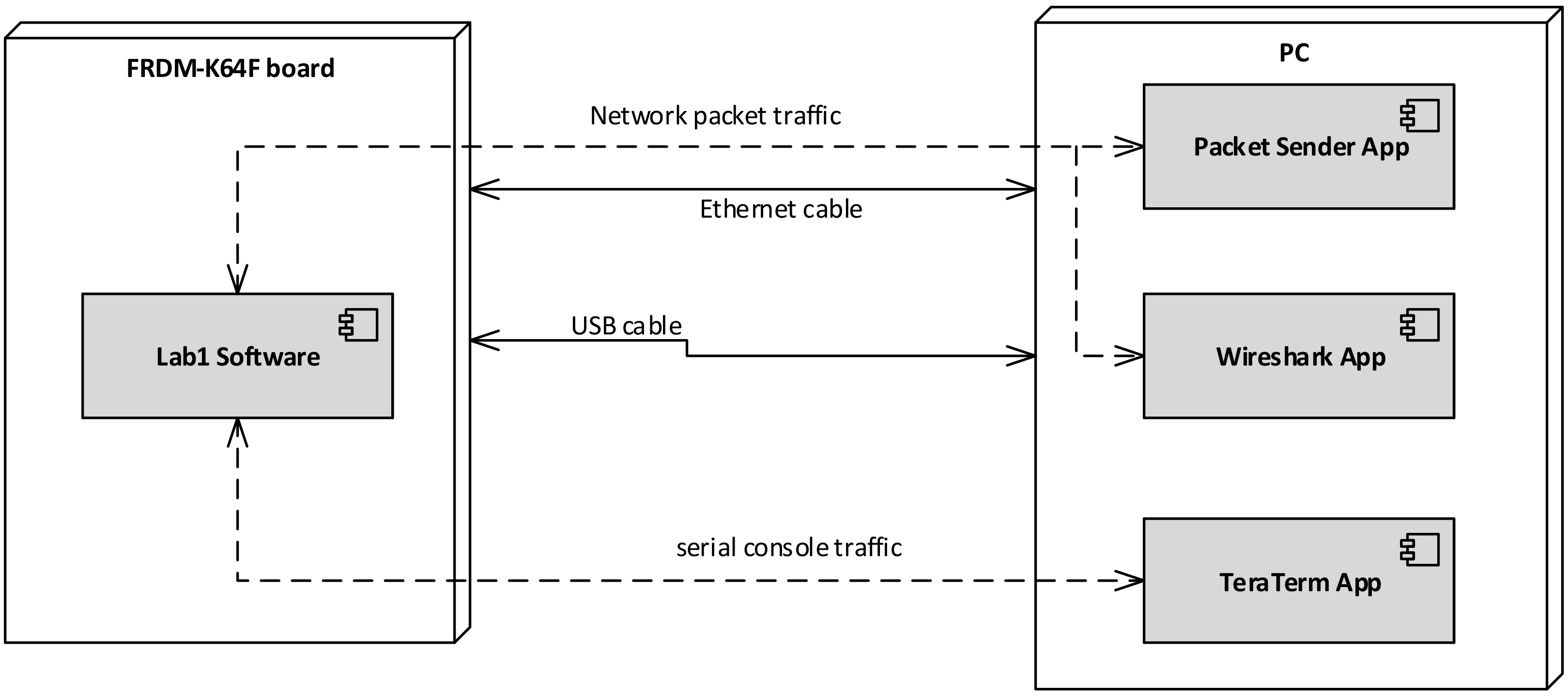


Test Plan (5)

- Test Case 4: Detection of Ethernet link up/down
  - Unplug the Ethernet cable that connects the FRDM-K64F board to the PC, from either end.
    - The board’s RGB LED should blink red
  - Re-plug the unplugged end of the Ethernet cable that connects the FRDM-K64F board to the PC
    - The board’s RGB LED should blink green

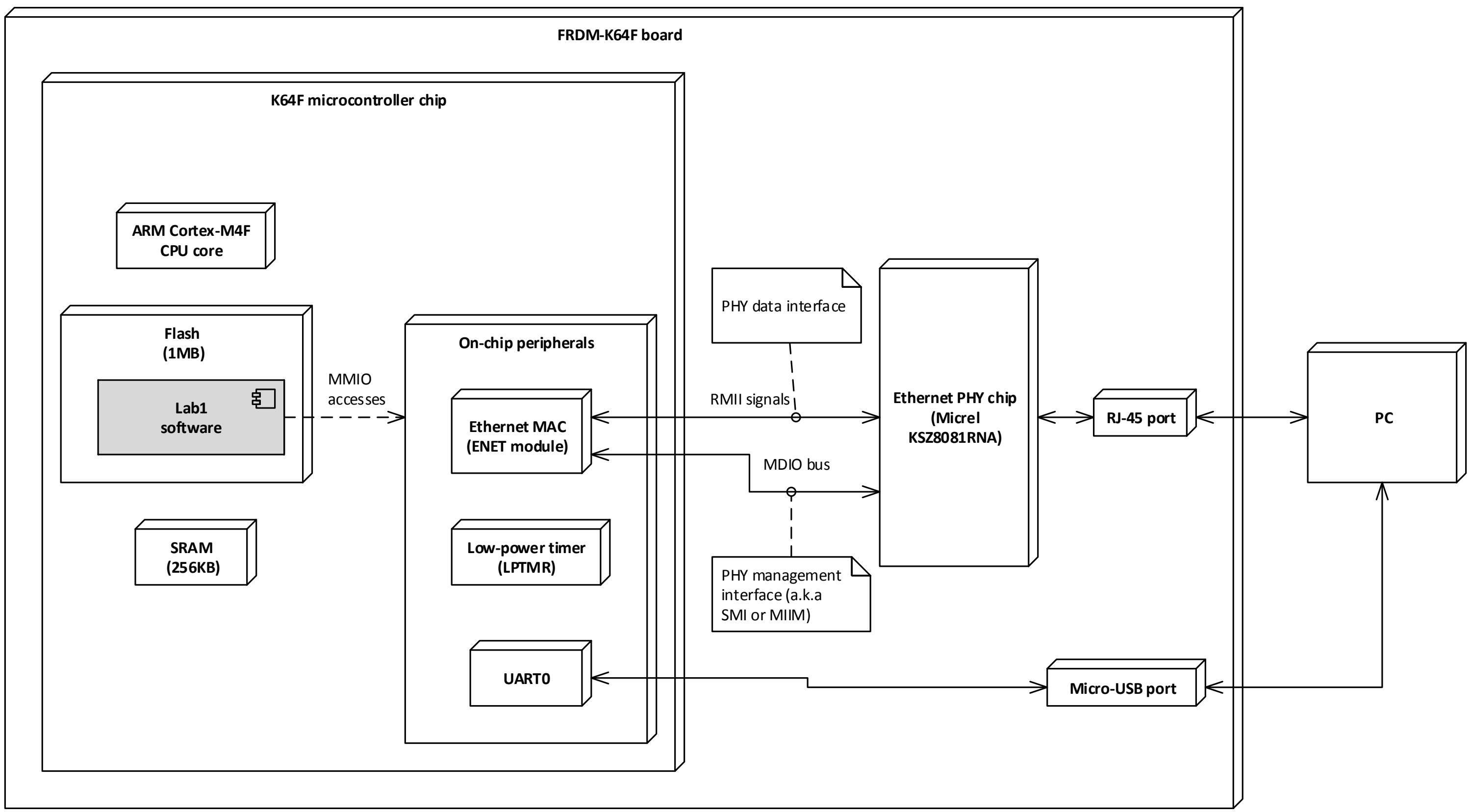
Software Architecture Diagrams

- Hardware/Software Topology



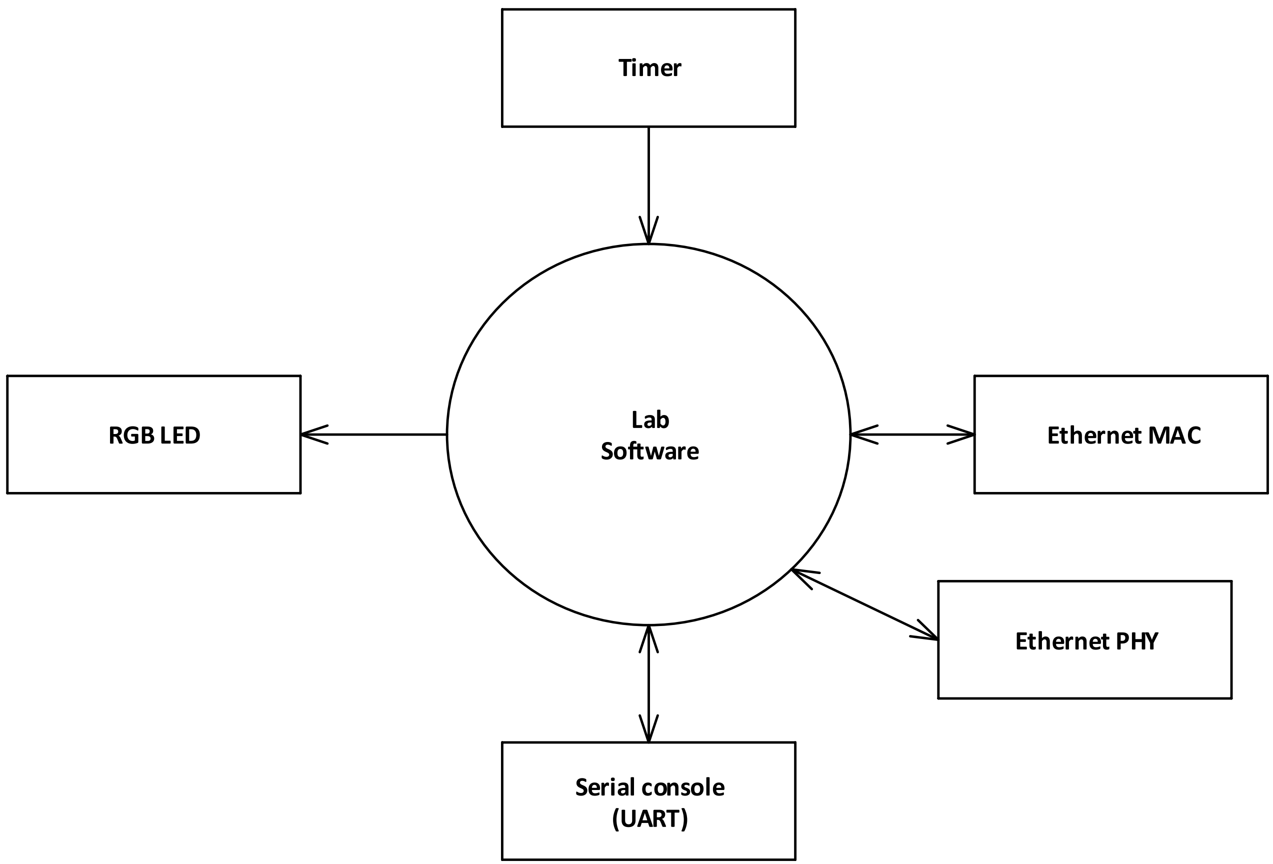
Software Architecture Diagrams (2)

- Hardware Context Diagram



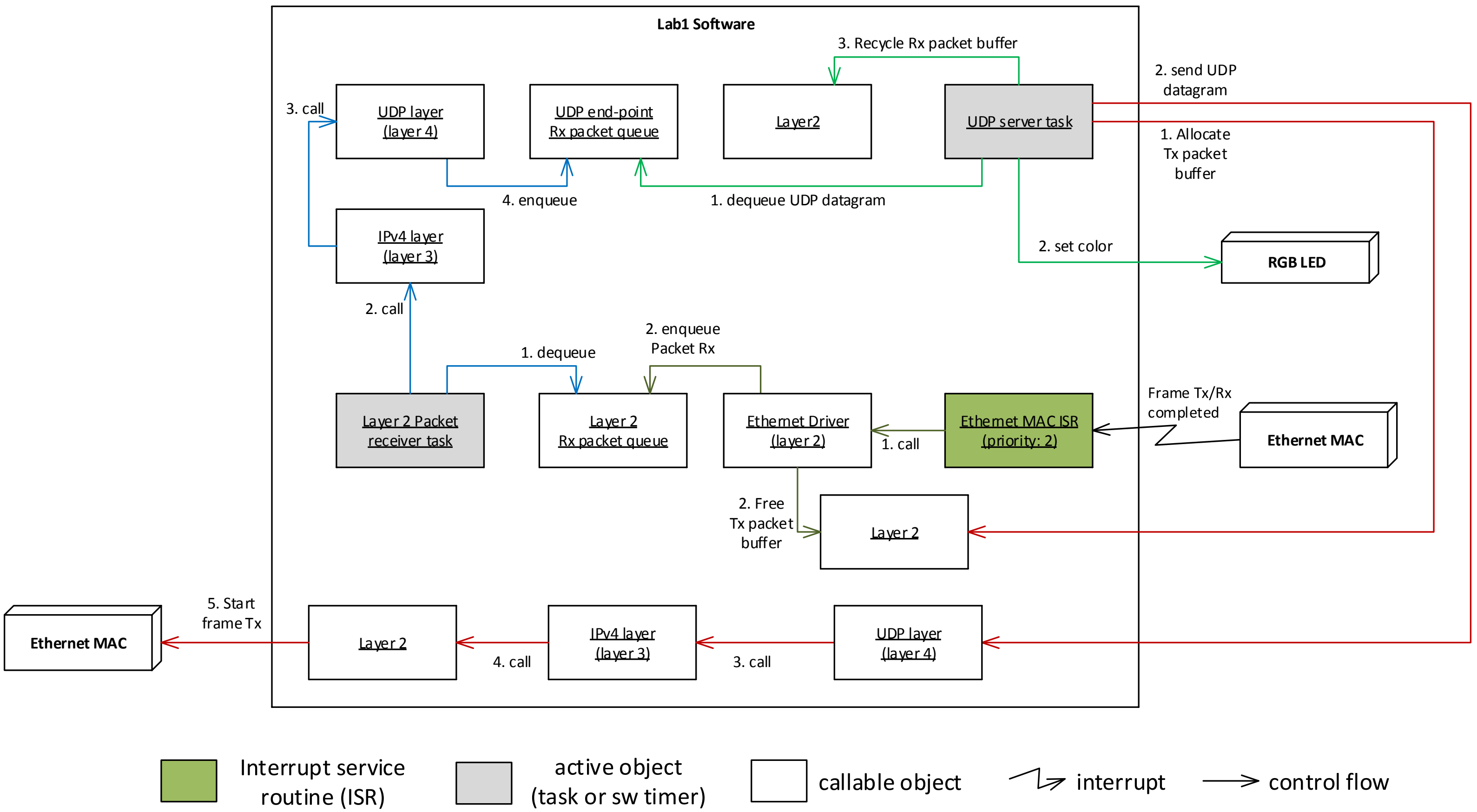
Software Architecture Diagrams (3)

- Software Context Diagram



Software Architecture Diagrams (4)

- Runtime Architecture



Software Architecture Diagrams (5)

- Runtime Architecture (cont.)