Embedded Networking Software Development
Lab 2 – Networking Layer 3 (Network Layer)

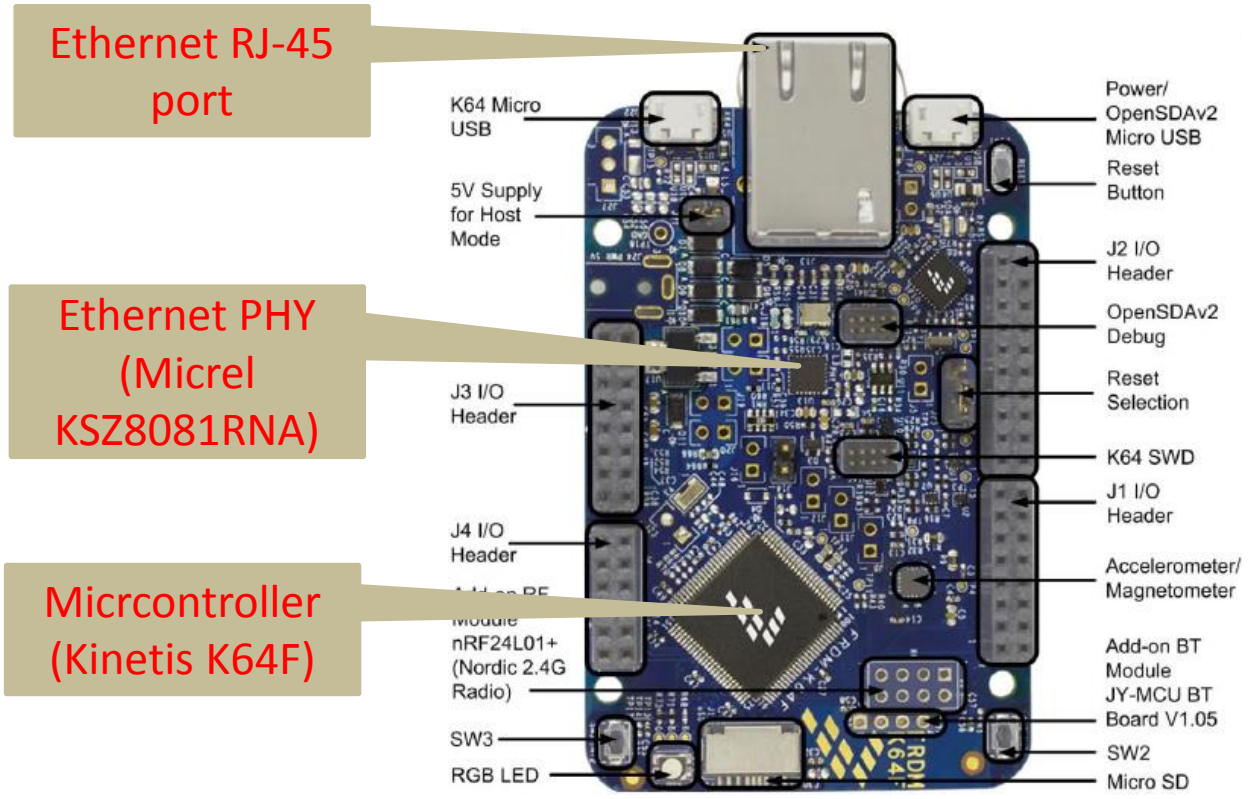
Version 0.1

Overview

- Purpose
  - Learn how the “ping” command is implemented in an IPv4-based networking stack.
- Learning objectives
  - To become familiar with the packet formats of the IPv4 and ICMP protocols
  - To learn how the “ping” command is implemented in an embedded networking stack for the FRDM-K64F board.

Overview (2)

- The FRDM-K64F board



Background Information

- The IPv4 protocol (Internet Protocol version 4)
  - Layer 3 protocol in the OSI model

Table 45-107. IPv4 header format

31 30 29 28	27 26 25 24	23 22 21 20	19 18 17 16	15 14 13 12	11 10 9 8	7 6 5 4	3 2 1 0
Version	IHL	TOS		Length			
Fragment ID				Flags	Fragment offset		
TTL		Protocol		Header checksum			
Source address							
Destination address							
Options							

Field name	Description
Version	4-bit IP version information. 0x4 for IPv4 frames.
IHL	4-bit Internet header length information. Determines number of 32-bit words found within the IP header. If no options are present, the default value is 0x5.
TOS	Type of service/DiffServ field.
Length	Total length of the datagram in bytes, including all octets of header and payload.
Fragment ID, flags, fragment offset	Fields used for IP fragmentation.
TTL	Time-to-live. In effect, is decremented at each router arrival. If zero, datagram must be discarded.
Protocol	Identifier of protocol that follows in the datagram.
Header checksum	Checksum of IP header. For computational purposes, this field's value is zero.
Source address	Source IP address.
Destination address	Destination IP address.

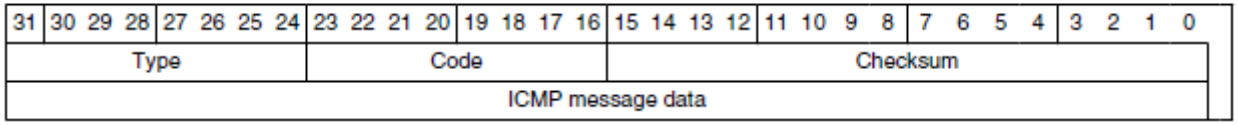
Background Information

- The IPv4 protocol (2)
  - Carries layer 4 packets
    - UDP datagrams
    - TCP segments
  - Carries layer 3 control packets
    - ICMP messages
    - IGMP messages
  - Carried in a layer 2 protocol such as Ethernet
    - An IPv4 packet is encapsulated in an Ethernet frame, using the value 0x0800 for the frame type field

Background Information (2)

- The ICMP protocol
  - Internet Control Message Protocol

Table 45-110. ICMP header format



Field name	Description
Type	8-bit type information
Code	8-bit code that is related to the message type
Checksum	16-bit one's complement checksum over the complete ICMP datagram

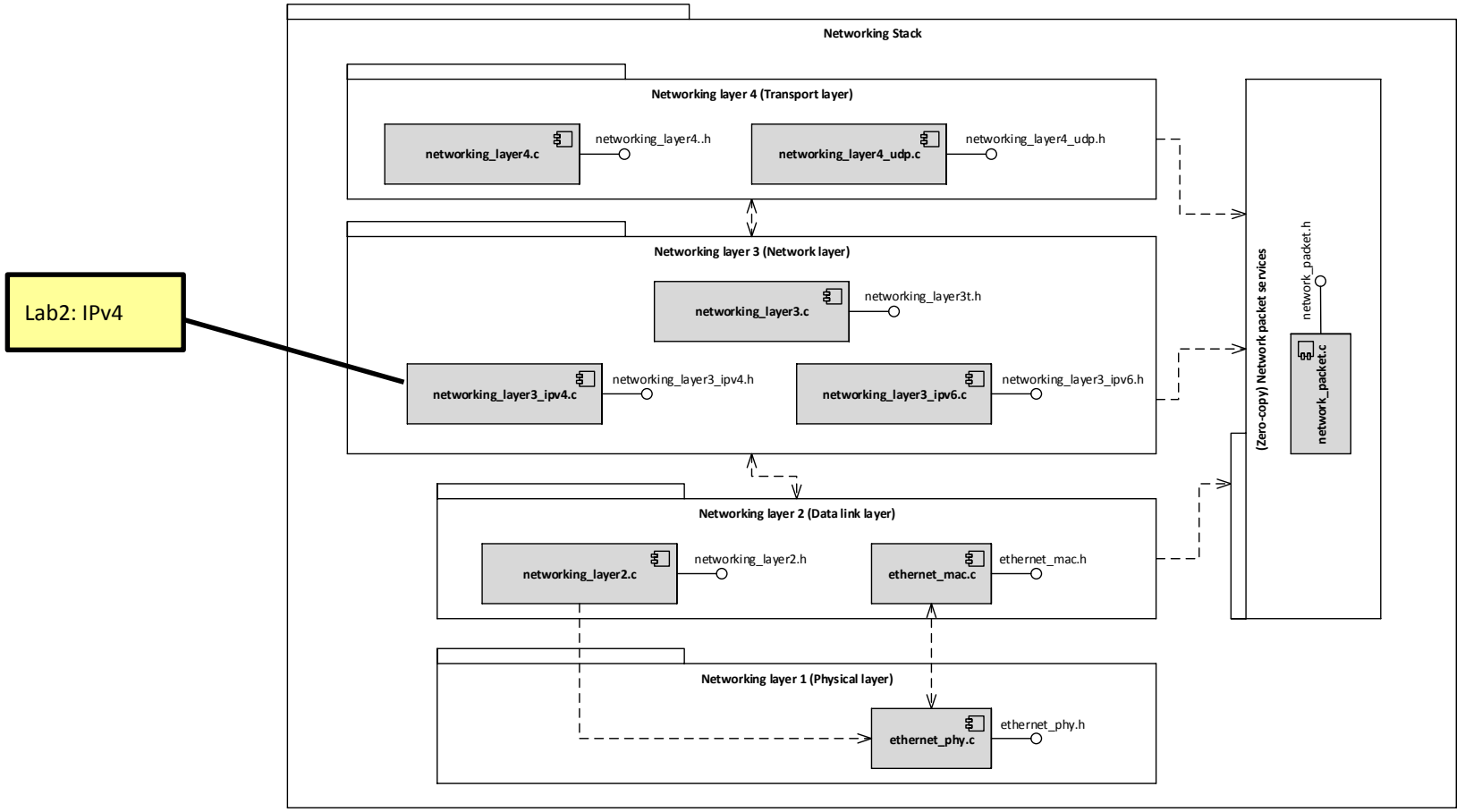
  - ICMPv4 messages are encapsulated in IPv4 packets
    - Protocol field of the IPv4 header is set to the value 0x01

Background Information (3)

- The “ping” command
  - Ping uses ICMP messages
  - It sends an ICMP “echo request” message to the target host and waits for an ICMP “echo reply” message from the target host.
- Ping request ICMP message
  - Type = 8, Code = 0
- Ping reply ICMP message
  - Type = 0, Code = 0

Lab Overview

- Networking Stack Architecture



Lab Overview (2)

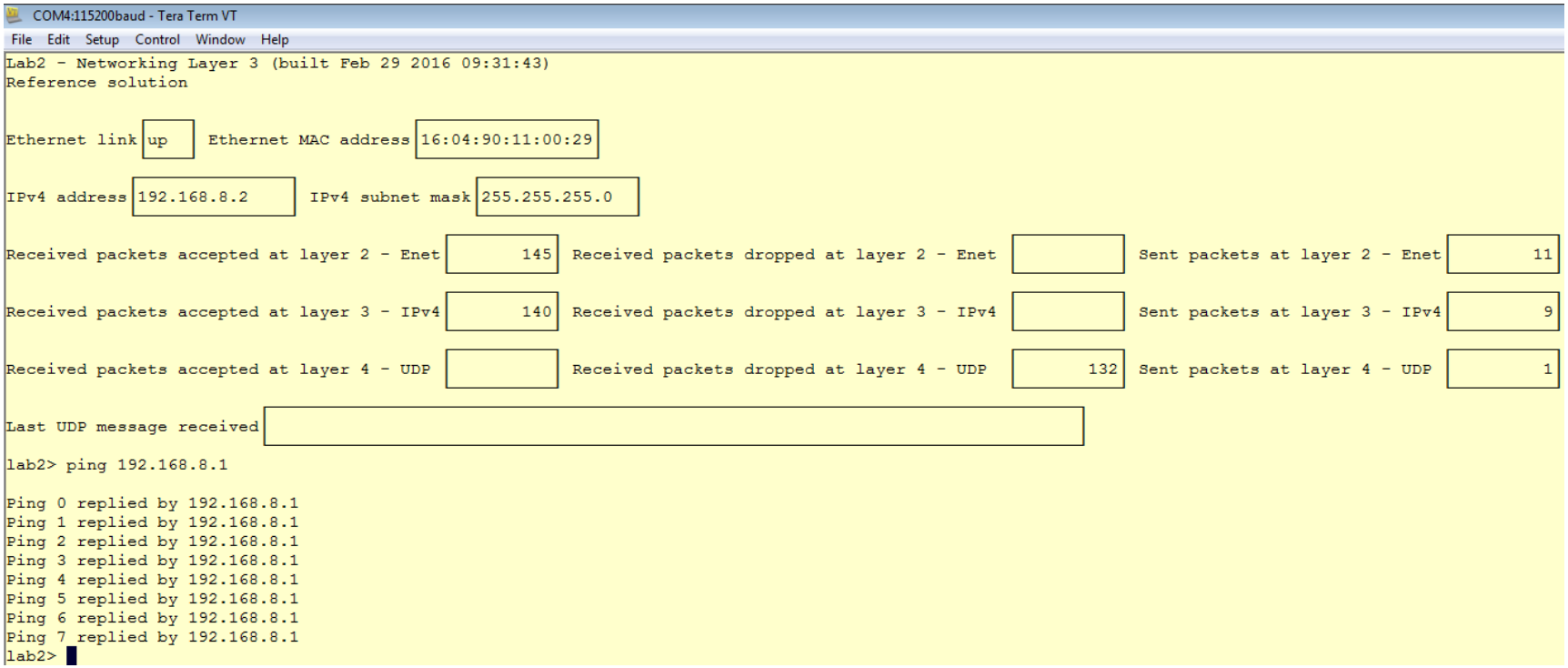
- Instructions:
  - Complete the code that implements the minimum support for the ICMP protocol in the embedded networking stack for the FRDM-K64F board, to be able to send and reply to “pings”:
    - Solve the 7 TODOs in the networking_layer3_ipv4.c source file.

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)

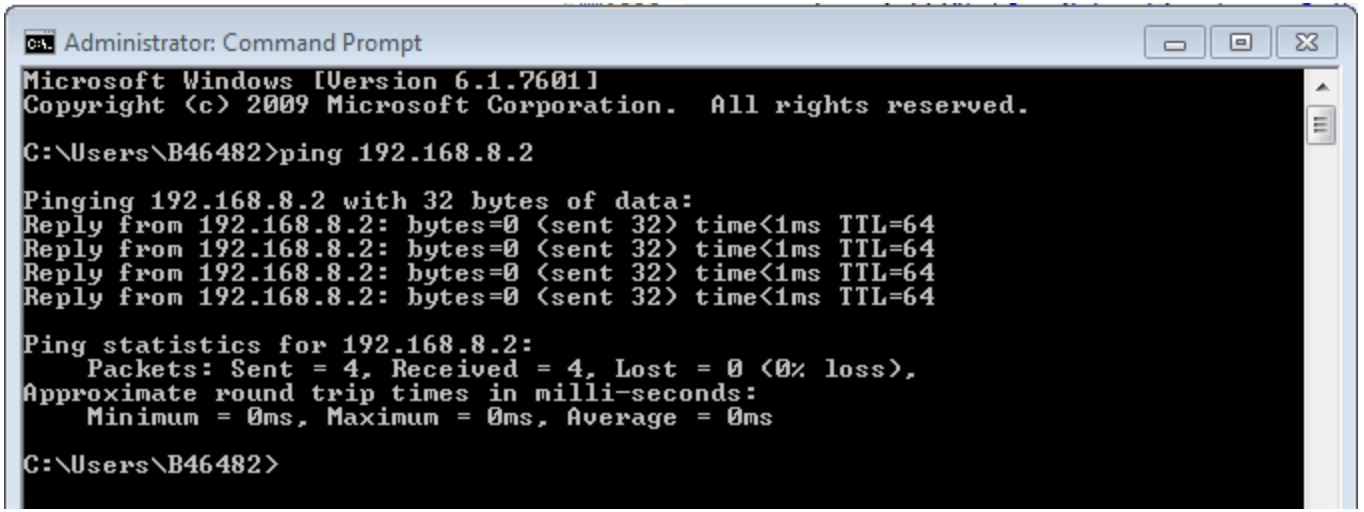
Test Plan (2)

- 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:



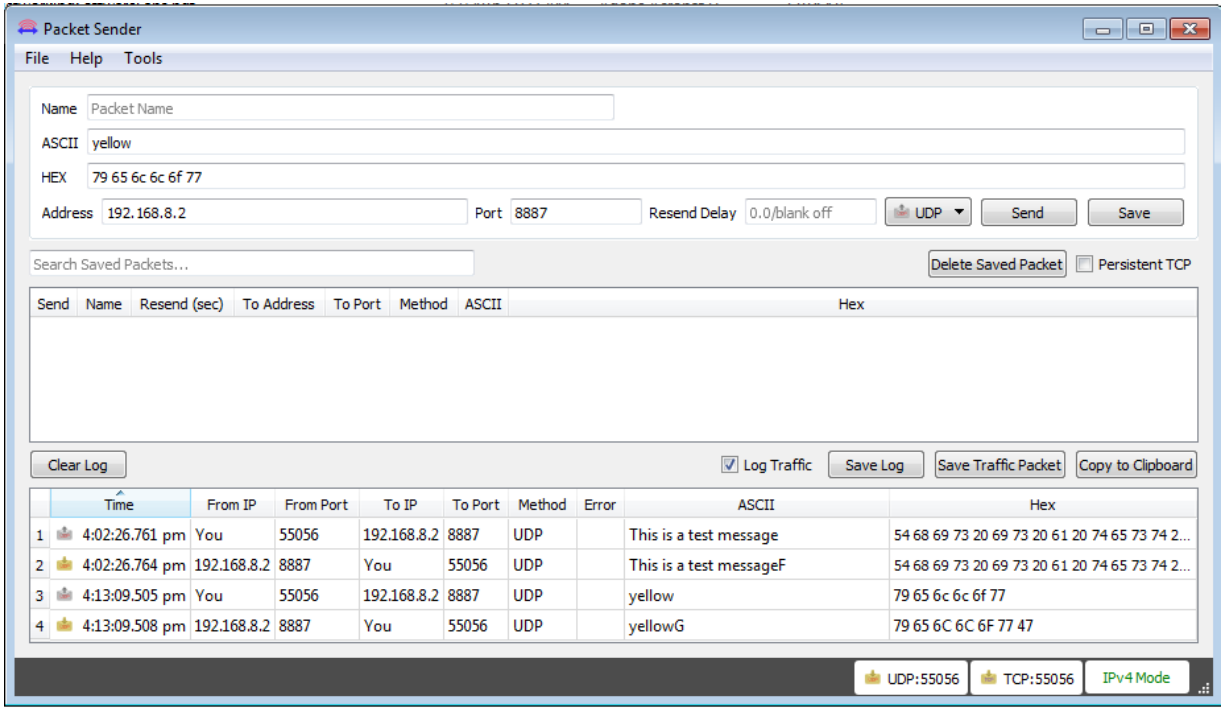
Test Plan (3)

- 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)

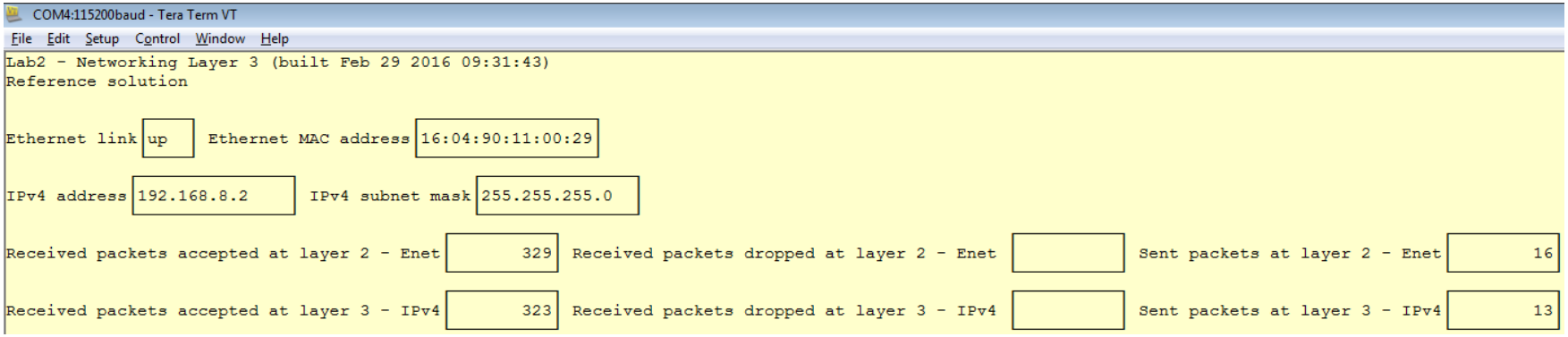
- Test Case 4: Wireshark Analysis
  - Use Wireshark to examine the “ping” ICMP packets:

	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…	192.168.8.2	192.168.8.1	ICMP	60 Echo (ping) request id=0x789a, seq=512/2, ttl=64 (reply in 550)
	550 1…	192.168.8.1	192.168.8.2	ICMP	42 Echo (ping) reply id=0x789a, seq=512/2, ttl=128 (request in 549)
	551 1…	192.168.8.2	192.168.8.1	ICMP	60 Echo (ping) request id=0x789a, seq=768/3, ttl=64 (reply in 552)
	552 1…	192.168.8.1	192.168.8.2	ICMP	42 Echo (ping) reply id=0x789a, seq=768/3, ttl=128 (request in 551)
	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)

	569 1…	192.168.8.1	192.168.8.2	ICMP	74 Echo (ping) request id=0x0001, seq=765/64770, ttl=128
	570 1…	192.168.8.2	192.168.8.1	ICMP	60 Echo (ping) reply id=0x0001, seq=765/64770, ttl=64
	571 1…	192.168.8.1	192.168.8.2	ICMP	74 Echo (ping) request id=0x0001, seq=766/65026, ttl=128
	572 1…	192.168.8.2	192.168.8.1	ICMP	60 Echo (ping) reply id=0x0001, seq=766/65026, ttl=64
	573 1…	192.168.8.1	192.168.8.2	ICMP	74 Echo (ping) request id=0x0001, seq=767/65282, ttl=128
	574 1…	192.168.8.2	192.168.8.1	ICMP	60 Echo (ping) reply id=0x0001, seq=767/65282, ttl=64
	575 1…	192.168.8.1	192.168.8.2	ICMP	74 Echo (ping) request id=0x0001, seq=768/3, ttl=128 (no
	576 1…	192.168.8.2	192.168.8.1	ICMP	60 Echo (ping) reply id=0x0001, seq=768/3, ttl=64

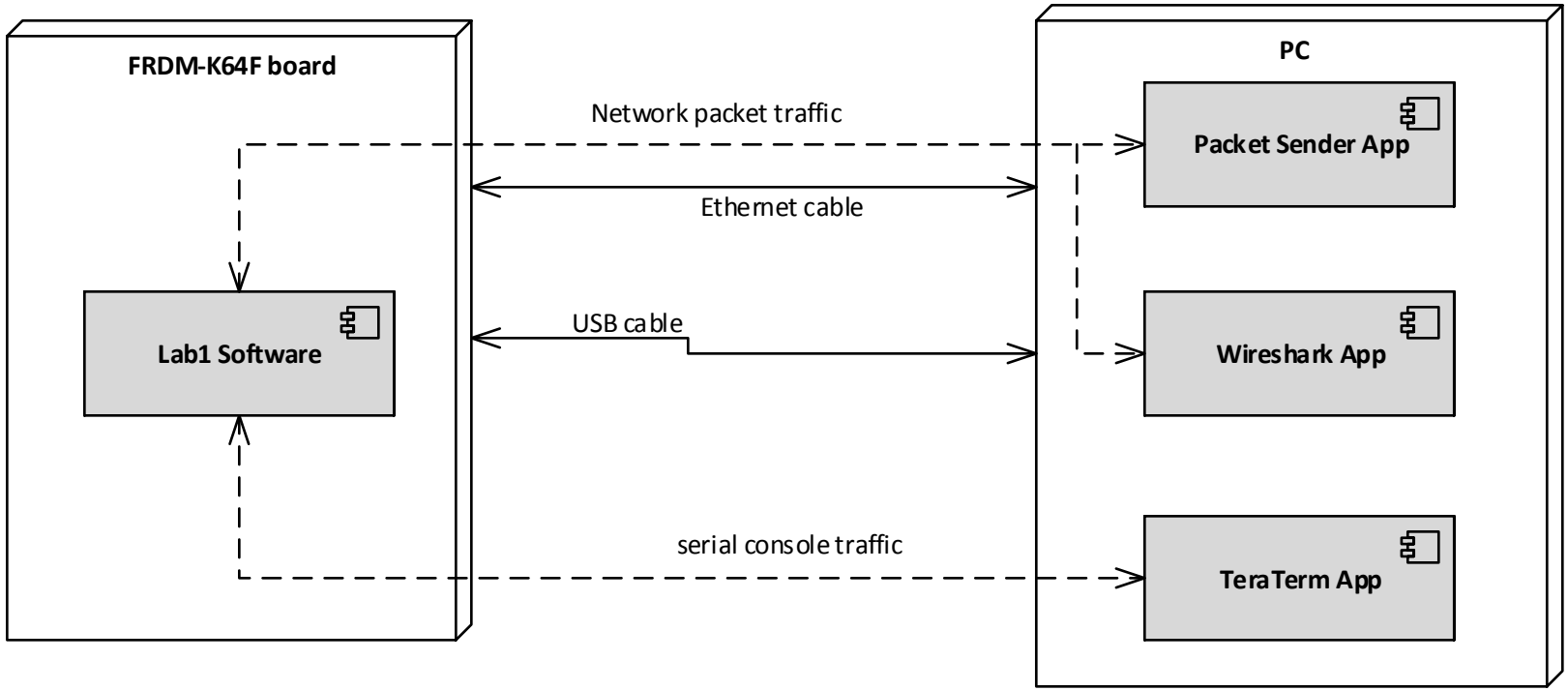
Test Plan (6)

- Test Case 5: Board Network Stats Analysis
  - Start a TeraTerm session and connect it to the board's serial console
  - While doing pings from the board to the PC and from the PC to the board, look at the Network stats dashboard on the board's serial console. Verify the consistency of the stats (e.g., # of packets received at layer 3 cannot be greater than # of packets received at layer 2)



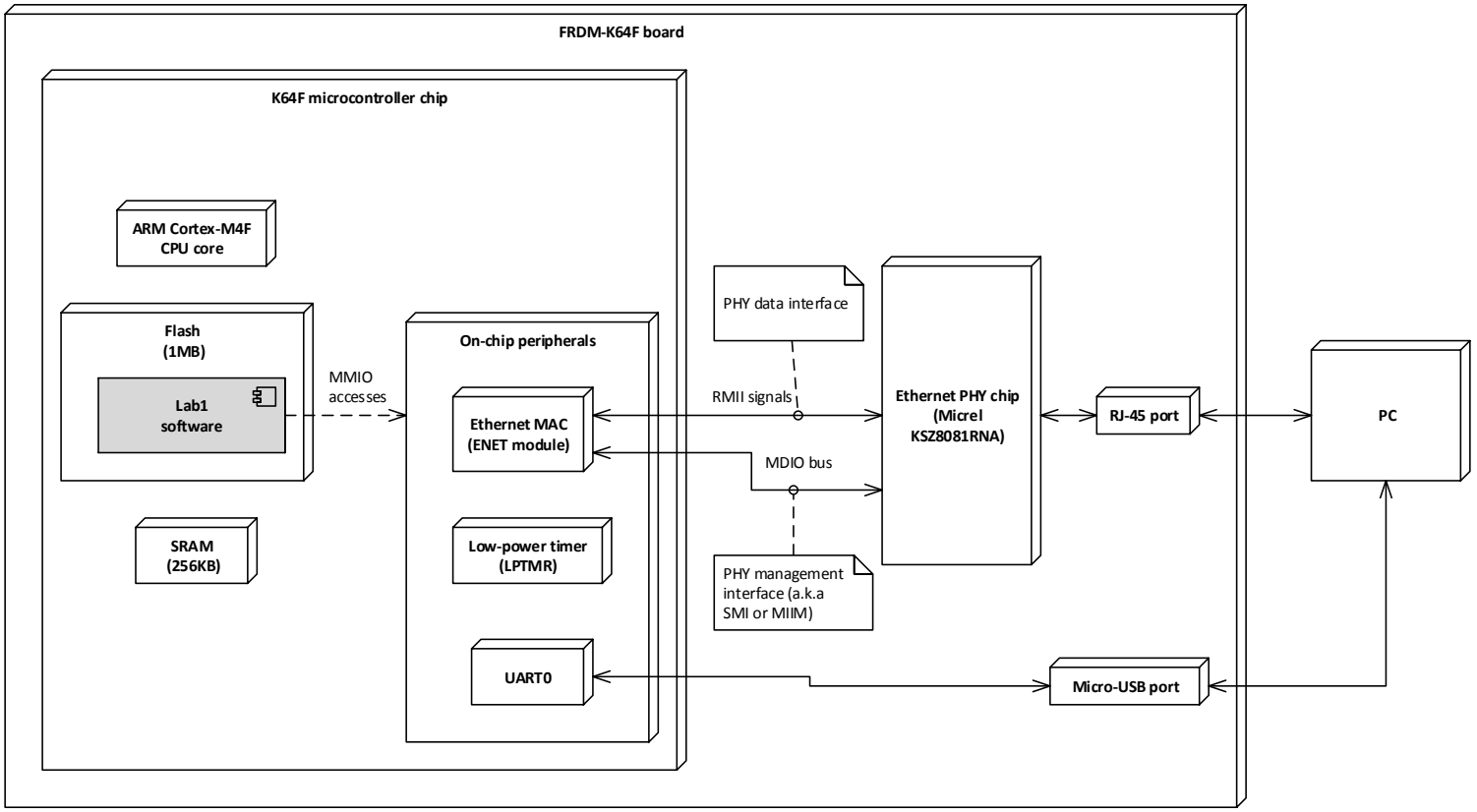
Software Architecture Diagrams

- Hardware/Software Topology



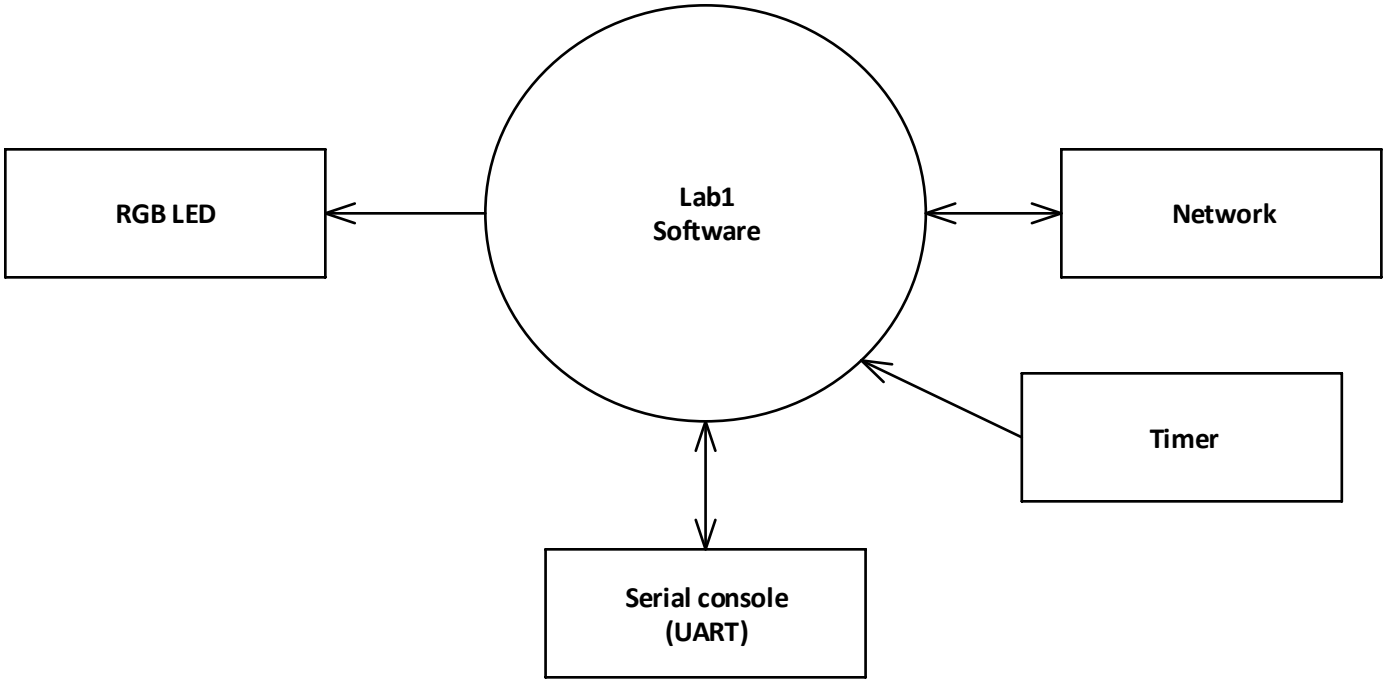
Software Architecture Diagrams (2)

- Hardware Context Diagram



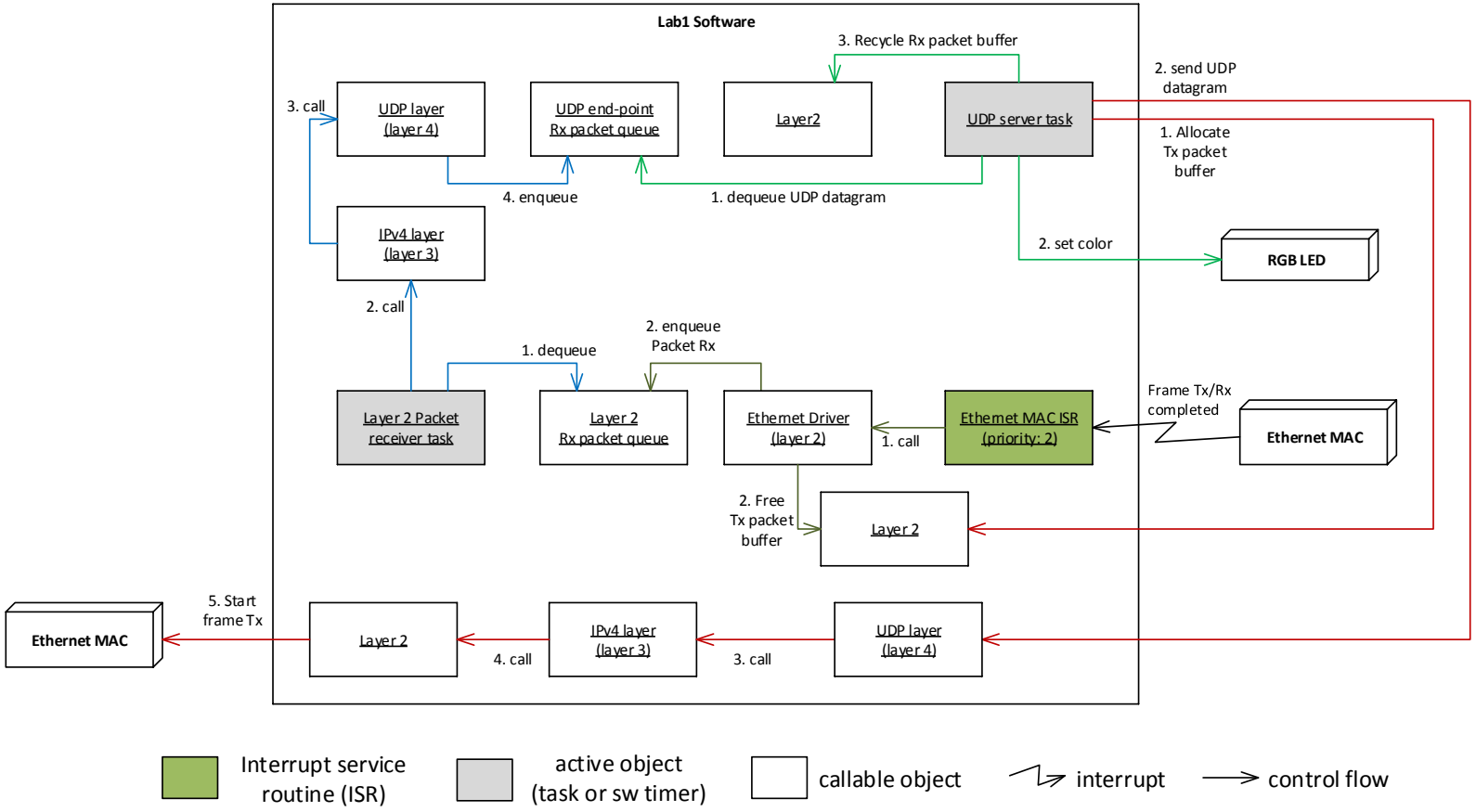
Software Architecture Diagrams (3)

- Software Context Diagram



Software Architecture Diagrams (4)

- Runtime Architecture



Software Architecture Diagrams (5)

- Runtime Architecture (cont.)