

**HKUSTx: ELEC1200.3x A System View of Communications: From Signals to Packets (Part 3)**

Bookmarks

► Pre-course
Materials

► Topic 1: Course
Overview

▼ Topic 2: The Link
Layer

2.1 Link Layer

Week 1 Quiz due Jan 25,
2016 at 15:30 UTC

**2.2 Multiple Access
Protocols**

Week 1 Quiz due Jan 25,
2016 at 15:30 UTC

**2.3 Aloha Protocol**

Week 1 Quiz due Jan 25,
2016 at 15:30 UTC

**2.4 Efficiency of Slotted
Aloha**

Week 1 Quiz due Jan 25,
2016 at 15:30 UTC



Topic 2: The Link Layer > 2.5 Lab 1: Link Layer > LAB 1 - TASK 2



Bookmark

LAB 1 - TASK 2 (EXTERNAL RESOURCE) (1.0 points possible)

2.5 Lab 1: Link LayerLab due Jan 25, 2016 at
15:30 UTC

- ▶ MATLAB download and tutorials

LAB 1 - TASK 2

In this task, you will learn how to create the frame that is used at the link level to send the datagram.

INSTRUCTIONS

The MATLAB code in the below window is similar to the code described in task 1 where we simulate the frame structure. The only difference in the code is that here we do not use the function **createFrame** to generate the frame. The detail.

Here, we consider a modified (simplified) frame structure, which consists four blocks, each with four bits. The first block contains the preamble, the node ID, the datagram, and the checksum. An example of the frame is

```
[ 1 0 1 0   0 0 0 1   1 0 1 0   0 0 0 1 ]
```

The preamble, which is "1010" in this example, is a fixed sequence of bits utilized to indicate the arrival of a frame. The second block, which is "0001" in this example, is the binary representation of the user's **id**. Here, the user **id** is 1, indicating that the frame is for user 1. The conversion of the user **id** from decimal to binary is achieved by using the function **num2bin**. Given that the frame structure only simulate a system with a maximum of 16 nodes. The third block, "1010", is the datagram of the user, which is generated by the function **getNewDatagram**.

The final block contains the checksum bits, which are "0001" in this example. The checksum bits are utilized for error detection during the transmission (recall the channel coding schemes we learned in Part I). In this simulation, the checksum is calculated for the first three blocks of the frame. Specifically, we first divide the first 12 bits in the frame into 4 groups as shown below:

```
[ 1 0 1 0 ]
[ 0 0 0 1 ]
[ 1 0 1 0 ]
```

Then, the 4 checksum bits are computed by performing the "exclusive or" operation over all three bits in each column. Equivalently, we can obtain the checksum bits by binary addition. The exclusive or of the bits in the first column is 0, the exclusive or of the bits in the second column is 0, the exclusive or of the bits in the third column is 0, and the exclusive or of the bits in the rightmost column is 1. As a result, the checksum bits for the above example are [0 0 0 1].

counting the number of 1s in the words and set the bit of the checksum to 1 if this number is odd, and 0 if we have two ones, which is even, so the checksum for that bit is 0.

Your task is to create the frame for the datagram and store the result inside the variable **frame**. In order to use the user's **id**, you can use the function **num2bin(id,4)**, where the second argument indicates the length of the binary representation of **id**. For example, if **id** is 10, the binary representation is **[1 0 1 0]** for this simulation. Please, revise the code between the lines

```
% % % % Revise the following code % % % %
```

and

```
% % % % Do not change the code below % % % %
```

Do not change other parts of the code and do not use the function **createFrame**.

Your Solution



Save



Reset



MATLAB Document

```
31 id_bin=num2bin(id,4);
32 checksum = [0 0 0 0];
33 checksum = bitxor(preamble, id_bin);
34 checksum = bitxor(checksum, datagram);
35 %frame = rand(1,16)>0.5;
36 frame = [preamble id_bin datagram checksum]
37 %disp(frame);
38 % % % % Do not change the code below % % % %
39
```

Assessment Tests: Passed



Is problem setup unmodified?

✓ Are the frames correct?

Output

frame =

1 0 1 0 0 0 0 1 1 1 0

frame =

1 0 1 0 0 0 1 0 0 0 1

frame =

1 0 1 0 0 0 1 1 0 1 1

frame =

1 0 1 0 0 1 0 0 0 0 1

frame =

1 0 1 0 0 0 1 0 1 1 0

frame =

1 0 1 0 0 1 0 0 1 1 0

frame =

1 0 1 0 0 0 1 0 0 1 1

frame =

1 0 1 0 0 0 1 1 0 1 1

frame =

1 0 1 0 0 1 0 0 1 1 0

frame =

1 0 1 0 0 0 1 1 1 0 1

frame =

1 0 1 0 0 1 0 0 0 0 1

frame =

1 0 1 0 0 0 1 1 0 1 1

frame =

1 0 1 0 0 1 0 0 1 1 1

frame =

1 0 1 0 0 0 1 1 0 0 0

frame =

1 0 1 0 0 0 1 0 0 0 0

frame =

1 0 1 0 0 0 0 1 1 0 0

frame =

1 0 1 0 0 0 1 0 1 1 0

frame =

1 0 1 0 0 0 1 1 1 0 1

frame =

1 0 1 0 0 0 1 0 0 0 0

frame =

1 0 1 0 0 0 0 1 1 1 0

frame =

Trame =

1 0 1 0 0 1 0 0 1 1 0

frame =

1 0 1 0 0 0 0 1 0 0 0

frame =

1 0 1 0 0 0 0 1 1 1 1

frame =

1 0 1 0 0 0 1 0 1 1 0

frame =

1 0 1 0 0 0 1 1 1 1 1

frame =

1 0 1 0 0 1 0 0 1 1 1

frame =

1 0 1 0 0 0 1 0 1 0 0

frame =

1	0	1	0	0	0	0	1	1	1	0
---	---	---	---	---	---	---	---	---	---	---

frame =

1	0	1	0	0	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---

frame =

1	0	1	0	0	0	1	1	1	1	0
---	---	---	---	---	---	---	---	---	---	---

frame =

1	0	1	0	0	0	0	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---

frame =

1	0	1	0	0	1	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---

frame =

1	0	1	0	0	1	0	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---

frame =

1	0	1	0	0	0	0	1	1	0	0
---	---	---	---	---	---	---	---	---	---	---

frame =


```
1 0 1 0 0 0 1 1 0 1 0  
  
frame =  
1 0 1 0 0 1 0 0 1 0 1  
  
frame =  
1 0 1 0 0 1 0 0 0 0 0
```

frame

© All Rights Reserved



© edX Inc. All rights reserved except where noted. EdX, Open edX and the edX and Open EdX logos are registered trademarks or trademarks of edX Inc.

POWERED BY
OPENedX



