

**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
- ▶ Topic 3: The Network Layer
- ▶ Topic 4: Routing
- ▶ Topic 5: The Transport Layer
- ▼ **Topic 6: Reliable Transfer Protocols**


Topic 6: Reliable Transfer Protocols > 6.4 Lab 3: Transport Layer > Lab 3 - Task 3




Bookmark

LAB 3 - TASK 3 (EXTERNAL RESOURCE) (1.0 points possible)


6.1 Stop-and-Wait Protocol

Week 3 Quiz due Feb 15, 2016 at 15:30 UTC 


6.2 Throughput of Stop-and-Wait

Week 3 Quiz due Feb 15, 2016 at 15:30 UTC 

6.3 Sliding Window Protocol

Week 3 Quiz due Feb 15, 2016 at 15:30 UTC 

6.4 Lab 3: Transport Layer

Lab due Feb 15, 2016 at 15:30 UTC 

► MATLAB download and tutorials

Lab 3 - Task 3

In this task, you will learn how the receiver in the stop and wait protocol passes data to the application layer. Remember that in the stop and wait protocol the receiver may receive duplicated packets, due to acknowledgements being lost and time outs. However, reliable transport protocols must ensure that segments containing the message are passed to the application layer in order and without duplication.

INSTRUCTIONS

The MATLAB code in the window below is similar to that in Task 1. However, since we are interested in the operation of the receiver, we show the implementation of the function **receiver_stopwait()** below.

```
% receiver
```

In each iteration of the simulation, the receiver first checks if a packet has arrived during the current time step using the function **receiver_get_packet()**. This function returns the packet (a 24 bit binary vector) if a packet has arrived or an empty matrix if no packet has arrived. If a packet has arrived, it extracts the first 4 bits of the packet, which contain the sequence number of the packet, and sends them back to the sender using the function **receiver_send_ack()**. Then, it adds the received packet to the list of received packets, called **receiver_packet_list**. Note that, when doing this, the initial code does not check if the new packet is duplicated or not, i.e., whether the new packet has been received before. As a result, the message, **rx_msg_1**, reconstructed from **receiver_packet_list** by the function **reconstruct_message()** may contain duplicated characters.

Your task is to filter out the duplicated packets from **receiver_packet_list** and to create a new list called **revised_packet_list**, that will contain the packets in the correct order and without redundant items. The initial code contains an incorrect implementation of the filtering described in the lecture on the stop and wait protocol.

To help you check your result, the code will reconstruct the correct message from the **revised_packet_list** and save it to the variable **rx_msg_2**. Revise the code between the lines

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

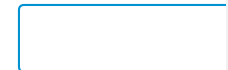
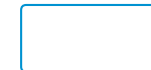
and

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

Please, do not change other parts of the code.

Your Solution Reset MATLAB DOCUMENTATION (<https://www.mathworks.com>)

```
57     %disp(c);
58     %disp(packet);
59     %disp(seq_num);
60     %disp(last_deliv);
61     % compare to last_deliv
62     if seq_num == last_deliv+1,
63         revised_packet_list = [revised_packet_list; packet];
64         last_deliv = seq_num;
65     %disp(revised_packet_list);
```

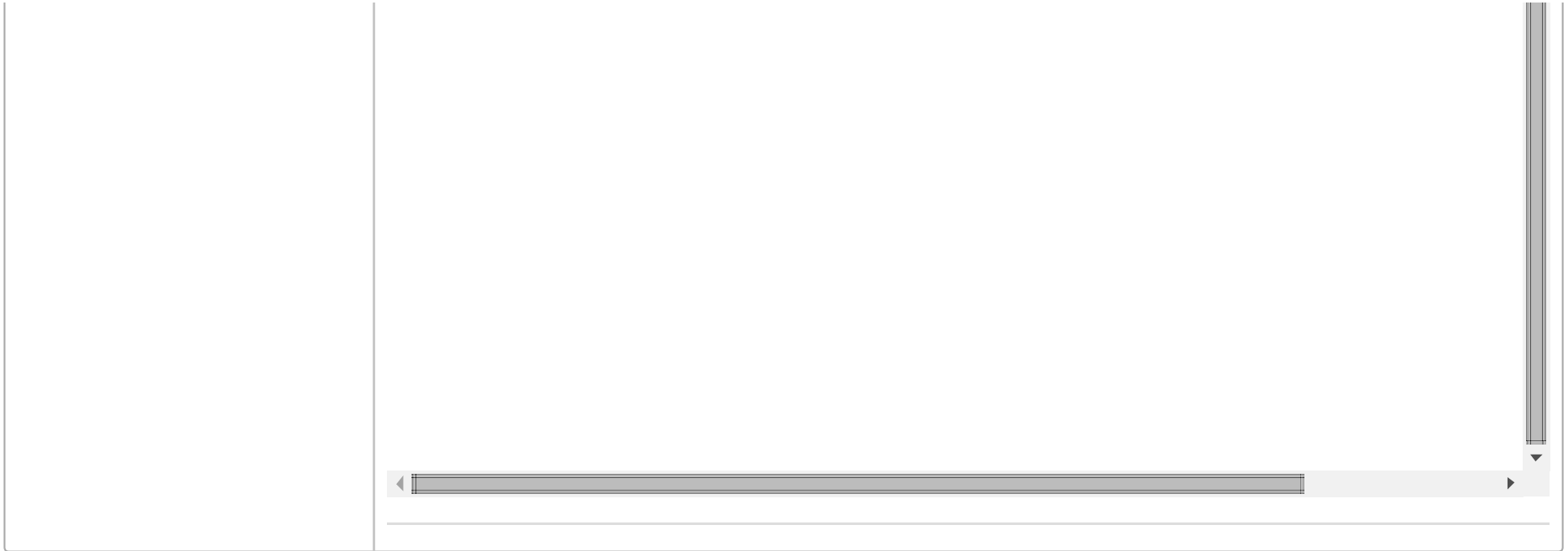
**Assessment Tests: Passed**

✓ Is the problem unchanged?

✓ Is the receiver correct?

Output

```
Simulation Time: 369
Number of packets sent correctly: 12
The message before filtering is: Heelllo  WWWWoorrld!
The message after filtering is: Hello World!
```

© 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



