

Assignment I

Shalini Sharma, Meenakshi Rana, Aneesa Khan

May 11, 2021

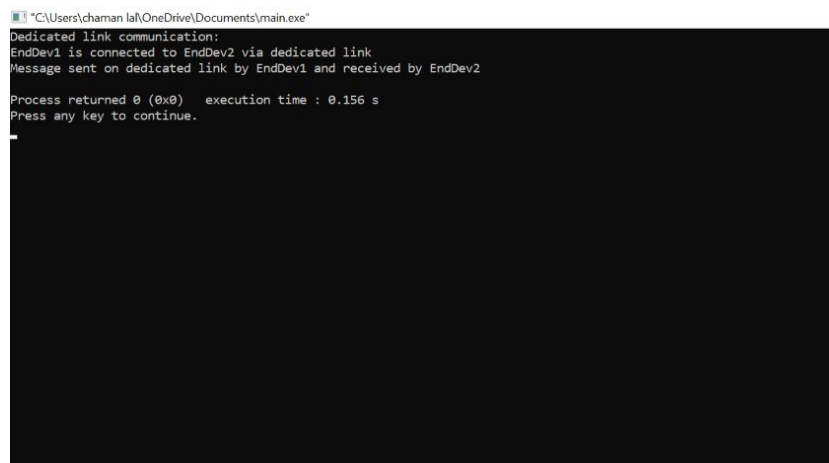
Contents

1	Implement Physical Layer Functionalities:	2
1.1	Two end devices with a dedicated connection	2
1.2	Star topology with five end devices	2
2	Implement Data link Layer Functionalities:	3
2.1	Create a switch with five end devices connected to it and enable data transmission between them. Demonstrate access and each of the flow control protocols. Also, report the total number of broadcast and collision domains in the given network.	3
2.2	Create two star topologies with five end devices connected to a hub in each case and then connect two hubs using a switch. Enable communication between all 10 end devices and report the total number of broadcast and collision domains in the given network.	8

1 Implement Physical Layer Functionalities:

1.1 Two end devices with a dedicated connection

In order to create an end device, we make a class **endDevice** and define objects for it, as and when required. The data members of this class are public, hence can be called from anywhere in the program. These include **ip_addr**, **mac**, **port**, **add**. For the case of dedicated link communication, **dedicatedLink** function is called, which simply uses two objects to connect the two end devices.



```
"C:\Users\chaman la\OneDrive\Documents\main.exe"
Dedicated link communication:
EndDev1 is connected to EndDev2 via dedicated link
Message sent on dedicated link by EndDev1 and received by EndDev2
Process returned 0 (0x0) execution time : 0.156 s
Press any key to continue.
```

Figure 1: Dedicated Link

1.2 Star topology with five end devices

In order to create Star Topology containing one hub and five end devices, we make a class **Hub** with data members, **num**, **ports** and define an array of five objects of class **endDevices**.

We have used function **getdata** to enter the data manually, in which the argument **n** represents the number of end devices, which in our case is five, the maximum limit. We can change the number of end devices accordingly. We manually enter the information including **mac**, **ip_address** and **port** of the hub for the end devices.

```

C:\Users\dhaman\la\OneDrive\Documents\main.exe
Single Hub Communication:
Enter info for device1
192.184.1.1 11-22-33-44-55-66 1
Enter info for device2
192.184.1.2 12-23-34-45-56-67 2
Enter info for device3
192.184.1.3 11-23-33-45-55-67 3
Enter info for device4
192.184.1.4 22-11-44-33-66-55 4
Enter info for device5
192.184.1.5 23-16-18-11-22-33 5
12-23-34-45-56-67 wants to send {1001} to 11-22-33-44-55-66
{1001} is broadcasted to every station through Hub 1
11-22-33-44-55-66 received the packet.Thats the destination. {1001} recieved by:11-22-33-44-55-66
Ack sent to:12-23-34-45-56-67 by 11-22-33-44-55-66
11-23-33-45-55-67 received the packet.Not the destination,packet dropped
22-11-44-33-66-55 received the packet.Not the destination,packet dropped
23-16-18-11-22-33 received the packet.Not the destination,packet dropped
Ack transmission:
{ACK} is broadcasted to every station through Hub 1
12-23-34-45-56-67 received the packet.Thats the destination. {ACK} recieved by:12-23-34-45-56-67
11-23-33-45-55-67 received the packet.Not the destination,packet dropped
22-11-44-33-66-55 received the packet.Not the destination,packet dropped
23-16-18-11-22-33 received the packet.Not the destination,packet dropped
Process returned 0 (0x0)   execution time : 166.312 s
Press any key to continue.

```

Figure 2: Star Topology with 5 End Devices

2 Implement Data link Layer Functionalities:

- 2.1 Create a switch with five end devices connected to it and enable data transmission between them. Demonstrate access and each of the flow control protocols. Also, report the total number of broadcast and collision domains in the given network.

```

C:\Users\chaman la\OneDrive\Documents\main.exe
Single BRIDGE communication:
Enter info for device1
192.184.1.1 00-00-56-F2-B5-12 1
Enter info for device2
192.184.1.2 00-26-DD-14-C4-EE 2
00-26-DD-14-C4-EE wants to send {1001} to 00-00-56-F2-B5-12
{1001} is broadcasted to every station through Hub 1
00-00-56-F2-B5-12 received the packet.Thats the destination. {1001} recieved by:00-00-56-F2-B5-12
Ack sent to:00-26-DD-14-C4-EE by 00-00-56-F2-B5-12
Ack transmission:
{ACK}is broadcasted to every station through Hub 1
00-26-DD-14-C4-EE received the packet.Thats the destination. {ACK} recieved by:00-26-DD-14-C4-EE
Process returned 0 (0x0)   execution time : 51.198 s
Press any key to continue.

```

Figure 3: Bridge Communication

```

C:\Users\chaman la\OneDrive\Documents\main.exe
Switch with 5 end devices
Enter info for 1 end devices
192.184.1.1 11-22-33-44-55-66 1
Enter info for 2 end devices
192.184.1.2 12-23-45-34-65-56 2
Enter info for 3 end devices
192.184.1.3 A2-71-23-81-51-44 3
Enter info for 4 end devices
192.184.1.4 2C-51-11-77-12-22 4
Enter info for 5 end devices
192.184.1.5 23-58-22-6A-14-AE 5
2C-51-11-77-12-22 wants to send {100101}to 11-22-33-44-55-66
Table is empty!Message broadcasted to all end devices
11-22-33-44-55-66 received the packet.Thats the destination. {100101} recieved by:11-22-33-44-55-66
Ack sent to:2C-51-11-77-12-22 by 11-22-33-44-55-66
12-23-45-34-65-56 received the packet.Not the destination,packet dropped
A2-71-23-81-51-44 received the packet.Not the destination,packet dropped
23-58-22-6A-14-AE received the packet.Not the destination,packet dropped
1 | 11-22-33-44-55-66
2 | 12-23-45-34-65-56
3 | A2-71-23-81-51-44
4 | 2C-51-11-77-12-22
5 | 23-58-22-6A-14-AE
Table created!
So,the destination port is 4
{ACK} recieved by 2C-51-11-77-12-22
Ack sent back to:11-22-33-44-55-66
Collision domain:5
Broadcast Domain:1
Process returned 0 (0x0)   execution time : 387.583 s
Press any key to continue.

```

Figure 4: Single switch with 5 end devices

StopNWait

In Stop N Wait ARQ protocol, the window size of both sender and receiver is one i.e. only one frame is sent, received and acknowledged at a time. Here, sender sends the frame which the receiver is expecting and when the latter gets the frame, it checks whether it is the expected frame or not. If yes, then the receiver will extract the data and deliver it to the network layer. It then sends an ACK for the next expected frame or discards the frame.

```
"C:\Users\chaman la\OneDrive\Documents\main.exe"
WHAT YOU WANT US TO DEMONSTRATE 1.STOP N WAIT 2.GO BACK N:1
Enter the number of frames : 5
Sent Frame 1
Frame 1 Not Received
Retransmitting Window

Sent Frame 1
Acknowledgment for Frame 1

Sent Frame 2
Frame 2 Not Received
Retransmitting Window

Sent Frame 2
Acknowledgment for Frame 2

Sent Frame 3
Acknowledgment for Frame 3

Sent Frame 4
Frame 4 Not Received
Retransmitting Window

Sent Frame 4
Frame 4 Not Received
Retransmitting Window

Sent Frame 4
Frame 4 Not Received
Retransmitting Window

Sent Frame 4
Acknowledgment for Frame 4

Sent Frame 5
Acknowledgment for Frame 5

Total number of transmissions : 10

Process returned 0 (0x0)   execution time : 5.420 s
Press any key to continue.
```

Figure 5: StopNWait

In order to create a switch connected to five end devices, we make a class **Switch** with data members **ports**, and an array of five objects of class **end-Device**. Also a parameterized constructor is defined to initialize ports to **p**.

We have used function **getData** to enter the data i.e. mac address , ip address and port number manually for the five end devices. We have defined the following functions : **GoBackN**, **StopNWait** to demonstrate the flow control protocols.

Inside the **StopNWait** function, **nf** represents the number of frames. This value is entered manually. **N** is the window size on the sender side which in this case is fixed i.e., 1. Since we start with frame 1, value of variable **i** also begins from 1. Here, **f** represents the frame that is currently being transmitted.

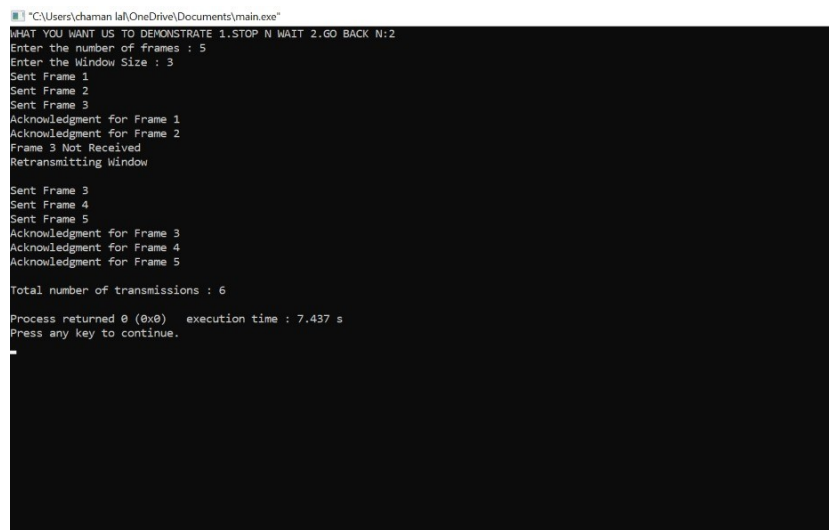
At the receiver end, **flag** value is randomly generated using **rand** function. If **flag**=0, it indicates that the frame is received successfully, and if it is 1, then frame needs to be resent, i.e. no ACK received. Here, **x** represents the *acknowledgement no.* which can take only two values, i.e. 0 or 1. Finally **i** is incremented by **x**. Also, **no_tr** represents the total number of successful

transmissions.

GoBackN

In Go Back N protocol the sender window size is 2^m-1 (where m is the reserved bits for sequence number of a frame) and Receiver window size is one. Here, the sender can send 2^m-1 frames without being acknowledged individually. Once the window is sent, the sender waits for the *cumulative acknowledgement*, which is an ACK which signifies that, all the previous frames are received successfully.

eg. ACK=3 means, frame 0,1,2 are successfully received. Once ACK is received, the window slides to the next set of unsent frames.



```
"C:\Users\chaman lah\OneDrive\Documents\main.exe"
WHAT YOU WANT US TO DEMONSTRATE 1.STOP N WAIT 2.GO BACK N:2
Enter the number of frames : 5
Enter the Window Size : 3
Sent Frame 1
Sent Frame 2
Sent Frame 3
Acknowledgment for Frame 1
Acknowledgment for Frame 2
Frame 3 Not Received
Retransmitting Window
Sent Frame 3
Sent Frame 4
Sent Frame 5
Acknowledgment for Frame 3
Acknowledgment for Frame 4
Acknowledgment for Frame 5
Total number of transmissions : 6
Process returned 0 (0x0)   execution time : 7.437 s
Press any key to continue.
```

Figure 6: GoBackN

We have used the following variables to implement Go Back N protocol. The variable **nf** is used to keep the count of the frame no. that the sender wants to send. **N** is the sender window size which can be entered manually. **i** represents the frame number from where the sliding window starts i.e. *first outstanding packet*.

At the receiver side, **flag** value is generated randomly using the **rand** function. When flag=0 it indicates that the frame is received successfully. If it is 1, then the frame needs to be resent. Here, if the frame is received

successfully, then **x** is used to keep track of the number of frames whose acknowledgements are received continuously. **i** is incremented by **x**. Variable **no_tr** is the total *number of transmissions*.

Slotted Aloha

In Slotted Aloha, we divide time into slots of **T_fr** seconds and force the station to send at the beginning of time slot. If a station misses out beginning, it has to wait for the next slot.

```

=====Slotted ALOHA=====
Station 1 wants to send data?y:n  y
Enter data:1011
Enter destination station id [1,5]:2
At what time interval beginning?Interval is of 5 seconds:5
Station 2 wants to send data?y:n  y
Enter data:1101
Enter destination station id [1,5]:3
At what time interval beginning?Interval is of 5 seconds:10
Station 3 wants to send data?y:n  n
Station 4 wants to send data?y:n  n
Station 5 wants to send data?y:n  n
1011received successfully by 2 at 5th interval
1101received successfully by 3 at 10th interval
Process returned 0 (0x0)   execution time : 28.331 s
Press any key to continue.

```

Figure 7: Slotted Aloha

In order to implement Slotted Aloha protocol, we've created a structure station, with three data members, i.e. **data**, **dest** and **st_no** representing the message sent, destination station and sender station, respectively. Next, a map **mp** is created which can contain two things, i.e. an integer which stores the time interval and a vector corresponding to that integer which stores the stations which is sending data in that time interval.

If the vector size is greater than 1, it indicates that multiple devices are sending data in that particular interval, i.e. a collision occurs. Inside the **slotted_aloha** function, the variable **ch** is used to determine whether a particular station is sending or not. Here, 'y' indicates a yes and 'n' indicates no.

The **time_** variable is used to enter the interval in which station wants to send. It also acts as a key value to the map and the variable **i** is entered

into the vector. The **visited** variable is used to indicate that the station has already been visited, in case it's value is set to 1.

In case the size of vector, i.e. **mp[i].size** is greater than 1, it indicates a collision and hence the packet needs to be dropped. Also if no. of attempts exceeds K_{max} which here is set to 5, no attempt is made to resend the packet. If not, then the array **t** is incremented, which keeps track of no. of resend attempts. A random no. greater than 5 is generated, to assign a time interval and it is made sure that it is a multiple of 5.

In case the size of the vector is equal to 1, it indicates that the packet is successfully sent. In case the no. of stations visited is equal to 5, i.e. all the stations are checked, the loop is broken.

2.2 Create two star topologies with five end devices connected to a hub in each case and then connect two hubs using a switch. Enable communication between all 10 end devices and report the total number of broadcast and collision domains in the given network.

```

C:\Users\shaman\OneDrive\Documents\main.exe
Switch with two hubs each connected to 3 end devices:
Enter info for device1
192.168.1.1 23-11-24-12-15-68 1
Enter info for device2
192.168.1.2 24-12-12-15-68 2
Enter info for device3
192.168.1.3 A1-18-7D-12-13-6C 3
Enter info for device4
192.168.1.1 C2-31-41-55-67-89 1
Enter info for device5
192.168.1.2 A3-41-61-32-77-41 2
Enter info for device6
192.168.1.3 A5-31-23-65-33-11 3
A1-18-7D-12-13-6C wants to send message {hello,there} to:23-11-24-12-15-68
{hello,there} is broadcasted to every station through Hub 1
23-11-24-12-15-68 received the packet.That's the destination. {hello,there} received by:23-11-24-12-15-68

Ack sent to:A1-18-7D-12-13-6C by 23-11-24-12-15-68
24-12-12-15-68 received the packet.Not the destination,packet dropped
{hello,there} is forwarded to switch
Switch will check table to which port it should send{hello,there}
table is empty
Switch will broadcast {hello,there}to port 1 and port 2
{hello,there} is broadcasted to every station through Hub 2
23-11-24-12-15-68 received the packet.That's the destination. {hello,there} received by:23-11-24-12-15-68

Ack sent to:A1-18-7D-12-13-6C by 23-11-24-12-15-68
24-12-12-15-68 received the packet.Not the destination,packet dropped
{hello,there} is broadcasted to every station through Hub 2
C2-31-41-55-67-89 received the packet.Not the destination,packet dropped
A3-41-61-32-77-41 received the packet.Not the destination,packet dropped
A5-31-23-65-33-11 received the packet.Not the destination,packet dropped
A1-18-7D-12-13-6C
1 23-11-24-12-15-68
2 24-12-12-15-68
3 A1-18-7D-12-13-6C
4 C2-31-41-55-67-89
5 A3-41-61-32-77-41
6 A5-31-23-65-33-11
Table created
Ack transmission:
{ACK} is broadcasted to every station through Hub 1

```

Figure 8: (a) Switch with 2 hubs and 3 end devices


```

C:\Users\chaman\OneDrive\Documents\main.exe
Switch with two hubs each connected to 3 end devices:
Enter info for device1
192.184.1.1 23-11-24-12-15-68 1
Enter info for device2
192.184.1.2 24-22-41-E2-A5-44 2
Enter info for device3
192.184.1.3 A1-18-7D-12-13-6C 3
Enter info for device1
191.184.1.1 C2-31-41-55-67-89 1
Enter info for device2
191.184.1.2 A3-41-61-32-77-41 2
Enter info for device3
191.184.1.3 A5-31-23-65-33-11 3
A1-18-7D-12-13-6C wants to send message {Hello,there!} to 23-11-24-12-15-68
{Hello,there!} is broadcasted to every station through Hub 1
23-11-24-12-15-68 received the packet. That's the destination. {Hello,there!} received by: 23-11-24-12-15-68
Ack sent to A1-18-7D-12-13-6C by 23-11-24-12-15-68
24-22-41-E2-A5-44 received the packet. Not the destination, packet dropped
{Hello,there!} forwarded to switch
Switch will check table to which port it should send {Hello,there!}
Table is empty!
Switch will broadcast {Hello,there!} to port 1 and port 2
{Hello,there!} is broadcasted to every station through Hub 1
23-11-24-12-15-68 received the packet. That's the destination. {Hello,there!} received by: 23-11-24-12-15-68
Ack sent to A1-18-7D-12-13-6C by 23-11-24-12-15-68
24-22-41-E2-A5-44 received the packet. Not the destination, packet dropped
{Hello,there!} is broadcasted to every station through Hub 2
C2-31-41-55-67-89 received the packet. Not the destination, packet dropped
A3-41-61-32-77-41 received the packet. Not the destination, packet dropped
A5-31-23-65-33-11 received the packet. Not the destination, packet dropped
1 | 23-11-24-12-15-68
1 | 24-22-41-E2-A5-44
1 | A1-18-7D-12-13-6C
2 | C2-31-41-55-67-89
2 | A3-41-61-32-77-41
2 | A5-31-23-65-33-11
Table created!
Ack transmission:
{ACK} is broadcasted to every station through Hub 1

```

Figure 9: (b) Switch with 2 hubs and 3 end devices

In order to establish communication in a network of two hubs, each having five end devices and connected to a switch as well, we call the **lastFunc** function. The argument to **s** determines the number of devices connected to the switch, which here is two. **h1** and **h2** are the two objects of class **Hub** which represent the two hubs connected to the switch.

```

C:\Users\chaman\OneDrive\Documents\main.exe
Switch with two hubs each connected to 5 end devices:
Enter info for device1
192.184.1.1 34-11-22-33-44-55 1
Enter info for device2
192.184.1.2 A2-C3-11-22-33-44 2
Enter info for device3
192.184.1.3 12-23-34-45-56-67 3
Enter info for device4
192.184.1.4 A3-21-11-22-33-44 4
Enter info for device5
192.184.1.5 C2-31-33-33-33-33 5
Enter info for device1
191.184.1.1 11-21-31-41-51-61 1
Enter info for device2
191.184.1.2 21-22-32-42-52-62 2
Enter info for device3
191.184.1.3 31-32-33-34-35-36 3
Enter info for device4
191.184.1.4 41-42-43-44-45-46 4
Enter info for device5
191.184.1.5 51-52-53-54-55-56 5
12-23-34-45-56-67 wants to send message {Hello,there!} to 34-11-22-33-44-55
{Hello,there!} is broadcasted to every station through Hub 1
34-11-22-33-44-55 received the packet. That's the destination. {Hello,there!} received by: 34-11-22-33-44-55
Ack sent to 12-23-34-45-56-67 by 34-11-22-33-44-55
A2-C3-11-22-33-44 received the packet. Not the destination, packet dropped
A3-21-11-22-33-44 received the packet. Not the destination, packet dropped
C2-31-33-33-33-33 received the packet. Not the destination, packet dropped
{Hello,there!} forwarded to switch
Switch will check table to which port it should send {Hello,there!}
Table is empty!
Switch will broadcast {Hello,there!} to port 1 and port 2
{Hello,there!} is broadcasted to every station through Hub 1
34-11-22-33-44-55 received the packet. That's the destination. {Hello,there!} received by: 34-11-22-33-44-55
Ack sent to 12-23-34-45-56-67 by 34-11-22-33-44-55
A2-C3-11-22-33-44 received the packet. Not the destination, packet dropped
{Hello,there!} is broadcasted to every station through Hub 2
11-21-31-41-51-61 received the packet. Not the destination, packet dropped
21-22-32-42-52-62 received the packet. Not the destination, packet dropped

```

Figure 10: (a) Switch With two hubs and five end devices each

The **getData** function is used to take input from the user regarding the details of the end devices i.e., IP and MAC addresses as well as the port to which they are connected. The number of end devices is determined by the argument to this function, which in this case is five.

```

"C:\Users\chaman lah\OneDrive\Documents\main.exe"
11-21-31-41-51-61 received the packet.Not the destination,packet dropped
21-22-32-42-52-62 received the packet.Not the destination,packet dropped
31-32-33-34-35-36 received the packet.Not the destination,packet dropped
1 | 34-11-22-33-44-55
1 | A2-C3-11-21-32-44
1 | 12-23-34-45-56-67
1 | A3-21-13-11-42-55
1 | C2-31-33-21-55-44
2 | 11-21-31-41-51-61
2 | 21-22-32-42-52-62
2 | 31-32-33-34-35-36
2 | 41-42-43-44-45-46
2 | 61-62-34-32-56-67
table created!
Ack transmission:
{ACK}is broadcasted to every station through Hub 1
A2-C3-11-21-32-44 received the packet.Not the destination,packet dropped
12-23-34-45-56-67 received the packet.That's the destination. {ACK} recieved by:12-23-34-45-56-67
A3-21-13-11-42-55 received the packet.Not the destination,packet dropped
C2-31-33-21-55-44 received the packet.Not the destination,packet dropped
{ACK} forwarded to switch
Switch will check table to which port it should send{ACK}
so port:1 have desired end device
{ACK}is broadcasted to every station through Hub 1
A2-C3-11-21-32-44 received the packet.Not the destination,packet dropped
12-23-34-45-56-67 received the packet.That's the destination. {ACK} recieved by:12-23-34-45-56-67
100101010 is broadcasted to send message (1001010) to:31-32-33-34-35-36
100101010 is broadcasted to every station through Hub 1
34-11-22-33-44-55 received the packet.Not the destination,packet dropped
A2-C3-11-21-32-44 received the packet.Not the destination,packet dropped
A3-21-13-11-42-55 received the packet.Not the destination,packet dropped
C2-31-33-21-55-44 received the packet.Not the destination,packet dropped
{1001010} forwarded to switch
Switch will check table to which port it should send{1001010}
so port:2 have desired end device
{1001010}is broadcasted to every station through Hub 2
11-21-31-41-51-61 received the packet.Not the destination,packet dropped
21-22-32-42-52-62 received the packet.Not the destination,packet dropped
31-32-33-34-35-36 received the packet.That's the destination. {1001010} recieved by:31-32-33-34-35-36
Ack sent to:12-23-34-45-56-67 by 31-32-33-34-35-36

```

Figure 11: (b) Switch With two hubs and five end devices each

```

"C:\Users\chaman lah\OneDrive\Documents\main.exe"
Ack sent to:12-23-34-45-56-67 by 31-32-33-34-35-36
Ack transmission:
{ACK}is broadcasted to every station through Hub 2
11-21-31-41-51-61 received the packet.Not the destination,packet dropped
21-22-32-42-52-62 received the packet.Not the destination,packet dropped
41-42-43-44-45-46 received the packet.Not the destination,packet dropped
61-62-34-32-56-67 received the packet.Not the destination,packet dropped
{ACK} forwarded to switch
Switch will check table to which port it should send{ACK}
so port:1 have desired end device
{ACK}is broadcasted to every station through Hub 1
34-11-22-33-44-55 received the packet.Not the destination,packet dropped
A2-C3-11-21-32-44 received the packet.Not the destination,packet dropped
12-23-34-45-56-67 received the packet.That's the destination. {ACK} recieved by:12-23-34-45-56-67
Collision domain:4
Broadcast Domain:3
Process returned 0 (0x0) execution time : 935.075 s
Press any key to continue.

```

Figure 12: (c) Switch With two hubs and five end devices each

The **send** function is called to initiate communication between the third and the first device of the first hub. The message is broadcast-ed by the hub using the **Receive** function, to the five end devices as well as the switch.

The **ReceiveF** function is called next so that the switch also starts sending data. However, the switch broadcasts the message, the first time around since it realizes that the table is empty with no entries.

Now, for the ACK transmission, the message to be sent is **ACK**. The source and destination mac addresses are swapped and the **Receive** function is called. The hub broadcasts the message and the intended receiver accepts it.

Again we initiate data transmission, but this time around between the third end devices of the two hubs. The **Receive** function is called and the first hub broadcasts the message from the third end device. Next, the **ReceiveF** function is called so that we can check if the table is created or not.

The table, this time around is already filled so the switch doesn't broadcast the message. As already explained in the first case, ACK is sent back as confirmation. Finally, the number of collision and broadcast domains are displayed.