Communication Networks

Assignment

Title : Implement ethernet switch forwarding engine using Python and write verification system using MATLAB.

Group Members

1) Arabelly Abhinav (2018B4AA0868H)
2) Garima Patel (2018B4AA0931H)
3) Paavani Khanna (2018B5AA1071H)

Date of Submission : 10/04/22

Contact Email ID : f20180868@hyderabad.bits-pilani.ac.in
f20180931@hyderabad.bits-pilani.ac.in
f20181071@hyderabad.bits-pilani.ac.in

BIRLA INSTITUTE OF TECHNOLOGY &
SCIENCE, PILANI- Hyderabad Campus

THEORY

Ethernet operates in the data link layer and physical layer. Ethernet switches are layer 2 devices that link other Ethernet devices together by sending Ethernet frames between the devices connected to them. Ethernet switches move the Ethernet frames between the switch ports using the learning and forwarding process.a
The switch keeps the MAC Address table in the memory. The MAC Address table is also known as the CAM table.

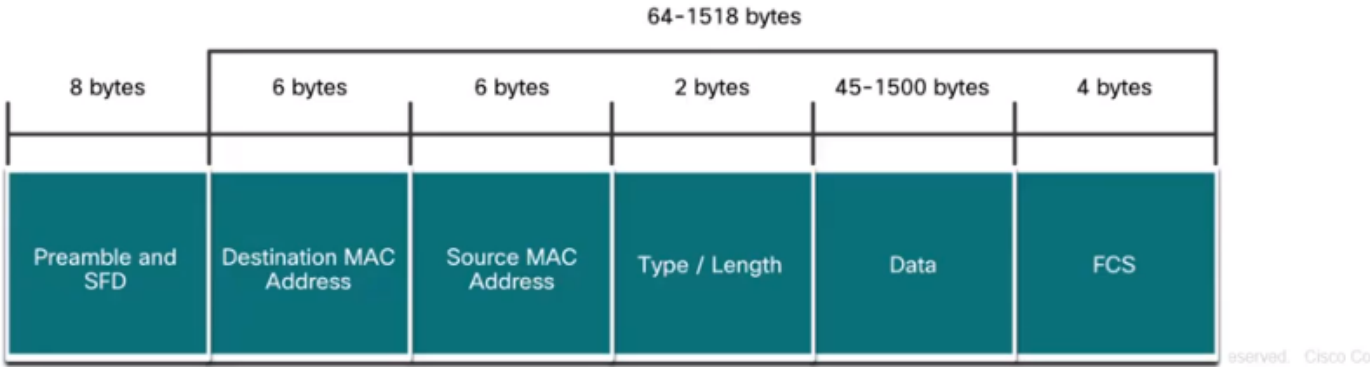


Figure 1. Ethernet Frame

Ethernet Switch: Learning and Forwarding

1. **Learn** - Switches learn by examining the source MAC Address and incoming port number. It checks if the MAC address and incoming port number are there in the MAC Address table. If they are not in the MAC Address table, it will add them to the table.
2. **Forward -** Forwarding of the frame is done by examining the destination MAC Address. It checks to see if the destination MAC address is present in the MAC Address Table. If it isn't present in the MAC Address Table, then it will flood it out at all the ports except the incoming port. In this case, it acts as a hub. If it is present in the MAC Address table, then it will just send it out to just that port.

FLOW CHART

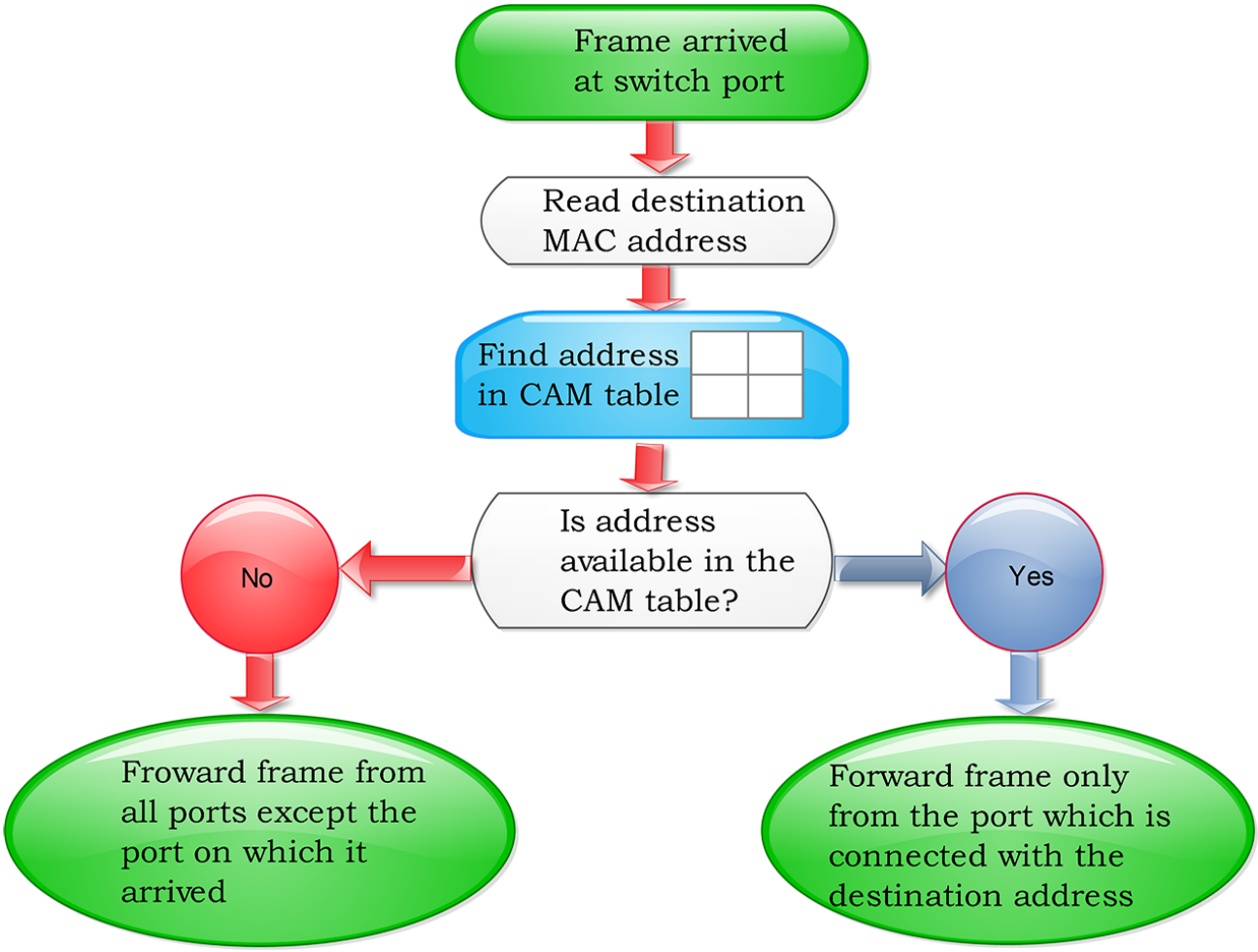
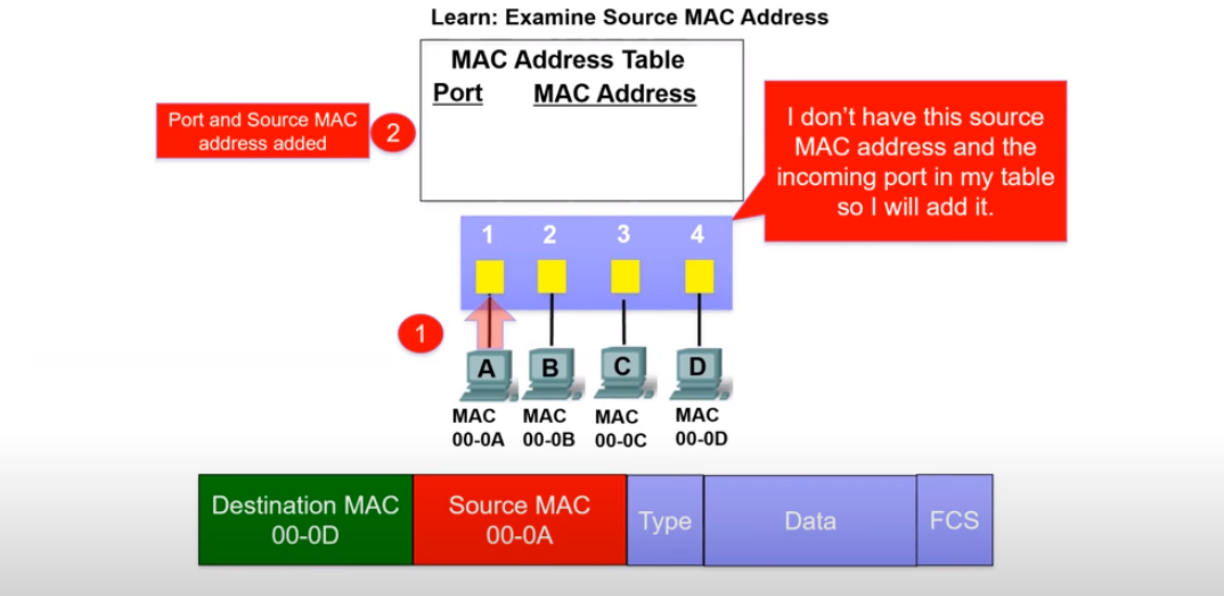
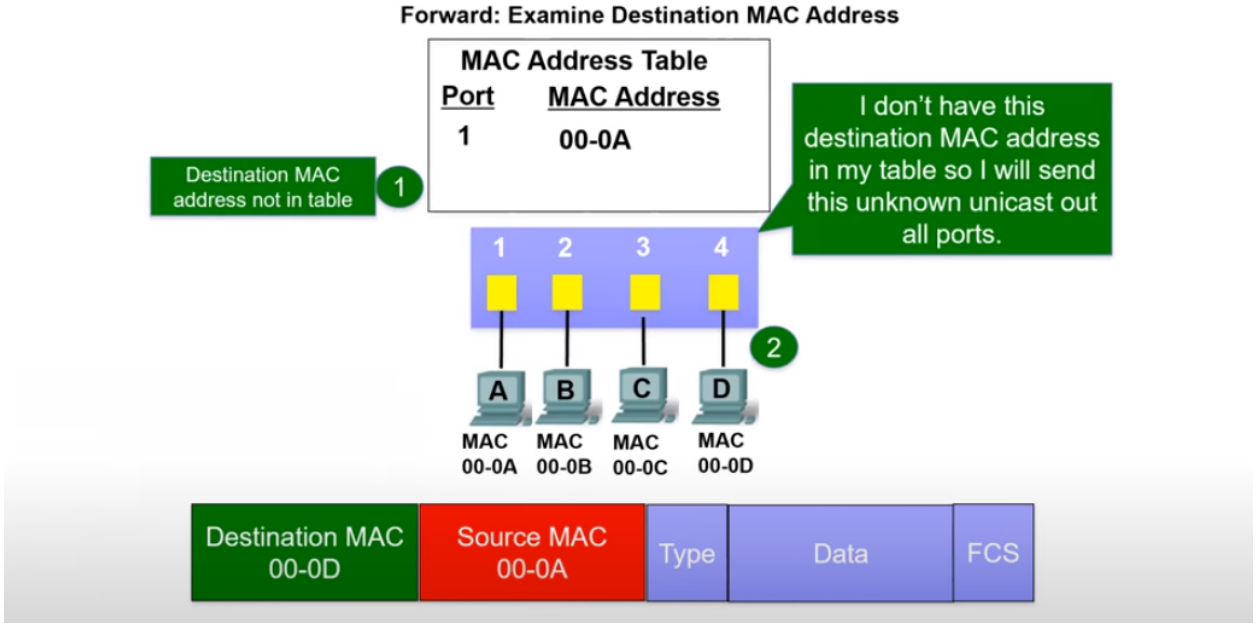


Figure 2. Learning and Forwarding Process

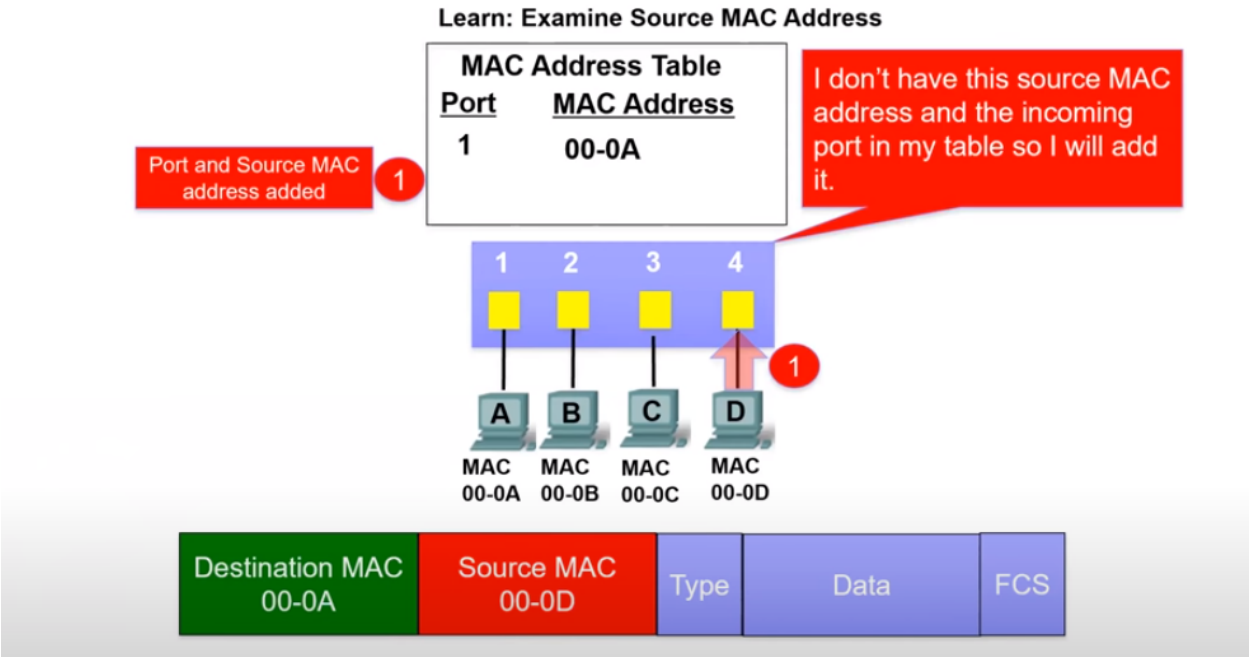
At the start of the learning and forwarding process, the MAC Address table is empty, so it stores the Source MAC address and the port number.

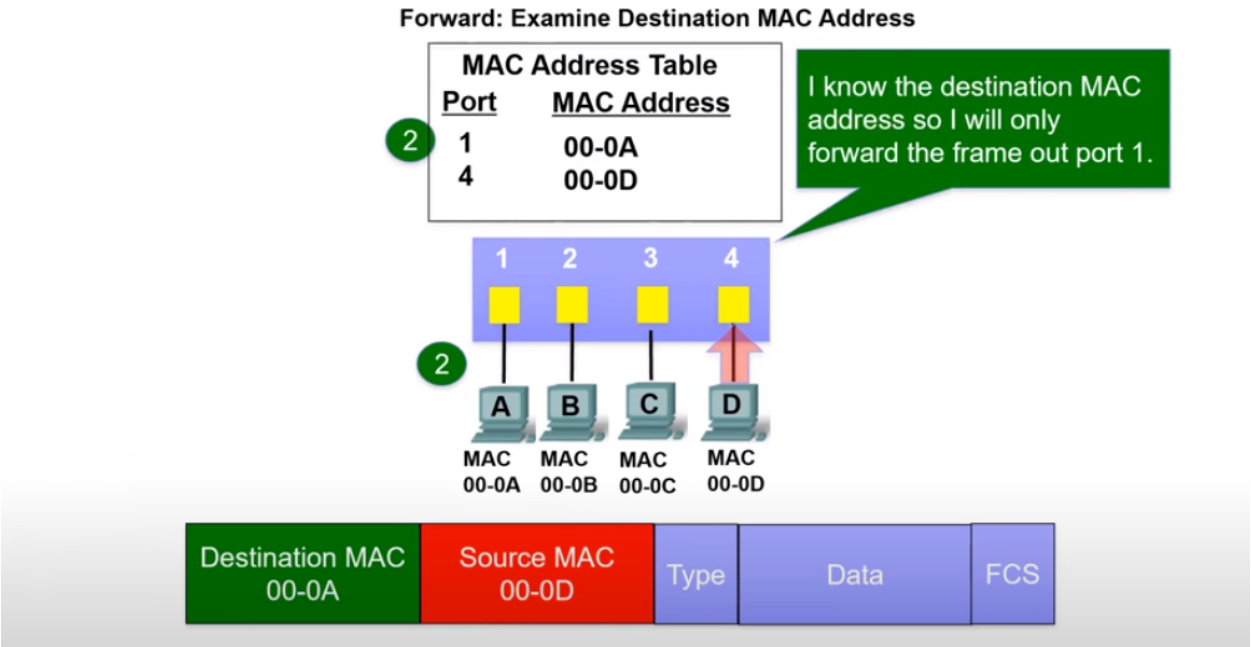


Next, it has to forward the frame based on the destination address. It checks in the MAC address table if the destination address is present or not. Since the destination address isn't present in the MAC address table, it will forward the frame to all the ports except port A (input port).



Next, if it wants to send a frame from port D to port A, then, it again looks up the Mac address from port D in the MAC address table and we find it's not there so it stores the address and the incoming port number in the MAC address table.





Next, it checks if the Destination MAC Address at port A is present in the MAC address table or not. Since it is present in the MAC address table, this time it will not flood out the frame to all the ports but will send only to Port A.

INPUTS AND ASSUMPTIONS TAKEN

Inputs taken from the user :
Input port number, source MAC address, destination MAC address

```
in_port = input("Enter input port :")
SRC_MAC = input("Enter the Source MAC Address :").strip()
DST_MAC = input("Enter the Destination MAC Address :").strip()
```

We have assumed that the mac addresses are connected to the fixed ports which cannot be changed for the entire duration of the run once the user enters it in the input.

MATLAB SIMULATION AND RESULTS

Matlab Code

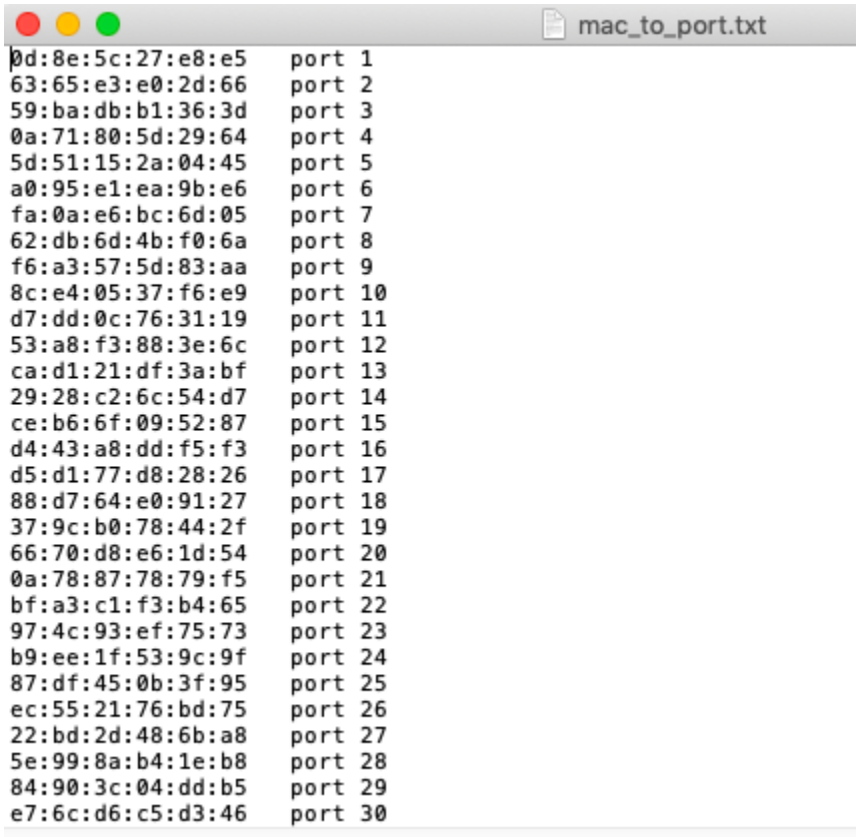
```
M= containers.Map();

while true

    input_port_number= input('Enter the input port no : ', 's');
    src_mac_address= input('Enter source mac address : ', 's');
    dest_mac_address = input('Enter destination mac address : ','s');


    %learn
    if isKey(M , src_mac_address) ~=1
        M(src_mac_address)= input_port_number;
    end

    %forwarding
    if isKey(M , dest_mac_address) ~=1
        statement= 'broadcast to all except input port';
        disp(statement)

    else
        statement= 'data sent to particular port';
        disp(statement)
        port= M(dest_mac_address);
        disp(port)
    end

end
```

Output:

```
>> ethernet_switch
Enter the input port no : 1
Enter source mac address : 123
Enter destination mac address : 234
broadcast to all except input port
Enter the input port no : 2
Enter source mac address : 24
Enter destination mac address : 123
data sent to particular port
1
Enter the input port no : 3
Enter source mac address : 19
Enter destination mac address : 24
data sent to particular port
2
Enter the input port no :
```

PYTHON SIMULATION AND RESULTS

The following text file shows the MAC address and port number generated



The following figure shows the simulation results of the python file
We enter the input port, source address and destination address. The code checks if the destination address is already in the Lookup Table if yes it sends the data to that particular port as " Data sent from port 19 to port 1" in the below figure.

The source address each time is learned implies we add it to the Lookup Table.

If the destination address is not present in the MAC address table, we send the data to all the ports except the port which is sending the data itself as shown in the figure below " Broadcast sent from port 1 to all ports except port 1"

```
Code_New — python input.py — 80×24
(base) abhinav:Code_New arjun$ python input.py
Enter input port :1
Enter the Source MAC Address :0d:8e:5c:27:e8:e5
Enter the Destination MAC Address :88:d7:64:e0:91:27
Broadcast sent from port 1 to all ports except port 1
Enter input port :22
Enter the Source MAC Address :bf:a3:c1:f3:b4:65
Enter the Destination MAC Address :22:bd:2d:48:6b:a8
Broadcast sent from port 22 to all ports except port 22
Enter input port :19
Enter the Source MAC Address :37:9c:b0:78:44:2f
Enter the Destination MAC Address :0d:8e:5c:27:e8:e5
Data sent from port 19 to port 1
Enter input port :26
Enter the Source MAC Address :ec:55:21:76:bd:75
Enter the Destination MAC Address :e7:6c:d6:c5:d3:46
Broadcast sent from port 26 to all ports except port 26
```

The following table shows the source address, Destination address and the action performed in the text file.

SRC_MAC	DST_MAC	Action
0d:8e:5c:27:e8:e5	88:d7:64:e0:91:27	Broadcast sent from port 1 to all ports except port 1
bf:a3:c1:f3:b4:65	22:bd:2d:48:6b:a8	Broadcast sent from port 22 to all ports except port 22
37:9c:b0:78:44:2f	0d:8e:5c:27:e8:e5	Data sent from port 19 to port 1

CONCLUSION

Through the assignment, we learnt the Ethernet Switch Forwarding Process. We wrote a script to learn the MAC addresses along with the input port number as the user enters the input values. When the ethernet switch encounters a destination MAC address that it hasn't learnt yet, then it will forward the information to all the ports except the input port. When it receives a destination MAC address that it has already learnt then it will forward the information to that specific port.

REFERENCES

1) https://www.youtube.com/watch?v=C0tGywT16dY
2) https://www.oreilly.com/library/view/ethernet-switches/9781449367299/ch01.html
3) https://www.youtube.com/watch?v=q4ZGh7INQgw

APPENDIX

1) PYTHON CODE

a) Generating MAC Address

We use the following code to obtain a text file containing the randomly generated mac addresses along with the port numbers (from 1 to 100). This is mainly for our own reference when running the code and does not affect the rest of the code.

Example : f6:a3:57:5d:83:aa port 1

```
import random

def random_MAC():
   return [ random.randint(0x00, 0xff),
            random.randint(0x00, 0xff),
            random.randint(0x00, 0xff),
            random.randint(0x00, 0xff),
            random.randint(0x00, 0xff),
            random.randint(0x00, 0xff) ]
```

```
def MACprettyprint(mac):
    return ':'.join(map(lambda x: "%02x" % x, mac))


if __name__ == '__main__':

    file= open('mac_to_port.txt', 'w')
    for i in range(1, 101):
        file = open("mac_to_port.txt", "a")
        file.write(MACprettyprint(random_MAC()))
        file.write("   port ")
        file.write(str(i))
        file.write("\n")
        file.close()
```

b) Ethernet Forwarding Process

```
def validate_mac_format(mac):  # This function is used to check the validity of the
MAC address entered by the user
    valid = True               # In case user enters address in the wrong format we
return error and exit the programme
    if not (len(mac) == 17):    # Checks if the length of the address is 17 or not
        valid = False
    byte_list = mac.split(":")
    for each_byte in byte_list:   # Checks if for each byte the length is less than 255
        try:
            mac_int = int(each_byte, 16)
            if not mac_int <= 255:
                valid = False
        except ValueError:
            valid = False
    return valid                # returns True or False as the value of the function
depending on the computations

text_file= open("write it.text","w")  # Define a text file to store the Source
address, Destination Address and Action Performed
text_file.write("\n")
text_file.write("SRC_MAC\t\t\t\t")
text_file.write("DST_MAC\t\t\t\t")
text_file.write("Action\t\n")       # Here Action Performed is whether the data is
sent to all the ports or one individual port
text_file.close()
```

```
dict1= {}   # Dictionary to store the Address and Port Values (MAC Address Table)

while True:

   in_port = input("Enter input port :") # Ask for the input port number by user
   if(int(in_port)>=100):                  # check for validity
       print("Enter Valid Port number less than 100")
       exit()

   SRC_MAC = input("Enter the Source MAC Address :").strip() # Ask for the Source MAC
address by user
   if(validate_mac_format(SRC_MAC)== False):                  # check for validity
       print("Invalid entry : " + SRC_MAC + " not in proper format, Please try again")
       exit()

   DST_MAC = input("Enter the Destination MAC Address :").strip() # Ask for the
Destination MAC address by user
   if(validate_mac_format(DST_MAC)== False):                  # check for validity
       print("Invalid entry : " + DST_MAC + " not in proper format, Please try again")
       exit()

   if DST_MAC == SRC_MAC:                                          # Return error if
source and destination address is same and exit the programme
       print("Invalid entry : Destination and source cannot be same. Try Again !! ")
       exit()

   packet=[in_port,SRC_MAC,DST_MAC]            # Store the input port, source address
and destination address in packet array

   # Learning                                  # Here we learn , as in if the source
address is not already present in the dictionary we add
   if packet[1] not in dict1:                  # the address along with the port
number to the dictionary
       dict1[packet[1]]= packet[0]

   # Forward                                    # The Forwarding process takes place
here depending on whether or not the destination address is already
                                               # present in the Lookup Table
   if packet[2] not in dict1:                  # If destination address is not
present in the lookup table we send the data to all the ports (broadcast)
```

```
        Action = "Broadcast sent from port "+packet[0]+ " to all ports except port "+
packet[0]
        print(Action)
    else:                                       # If the destination address is
present in the Lookup table, we only send the data to that particular port
        Action = "Data sent from port "+packet[0]+ " to port "+ dict1[packet[2]]
        print(Action)

    text_file = open("write it.text", "a")        # In this text file we store the
source address, destination address and the action
    text_file.write("\n")                         # that has been performed
    text_file.write(packet[1])
    text_file.write("\t\t")
    text_file.write(packet[2])
    text_file.write("\t")
    text_file.write(Action)
```

2) MATLAB CODE

```
M= containers.Map();

while true

    input_port_number= input('Enter the input port no : ', 's');
    src_mac_address= input('Enter source mac address : ', 's');
    dest_mac_address = input('Enter destination mac address : ','s');


    %learn
    if isKey(M , src_mac_address) ~=1
        M(src_mac_address)= input_port_number;
    end

    %forwarding
    if isKey(M , dest_mac_address) ~=1
        statement= 'broadcast to all except input port';
        disp(statement)

    else
        statement= 'data sent to particular port';
        disp(statement)
        port= M(dest_mac_address);
        disp(port)
    end

end
```