# University of Dhaka

# Department of Computer Science and Engineering

# 3rd Year,1st Semester

# Session - (2014 - 15)

# Course - (CSE - 3112)

# Report on

# **“Emulation of Sync-TDM and Stat-TDM”**

# Submitted by

# Redwan Ahmed Rizvee(Roll - 09)

# Mehreen Rahman (Roll - 61)

**Table of contents**

[**Problem Definition**](#_7hjoiiel71gz) **2**

[Defining the problem as what we expected to accomplish](#_1i6gizg511o3) 3

[The inputs and outputs](#_5z2lfisvvx63) 3

[Equipment required](#_9upt4lqpawp0) 3

[**Theoretical background**](#_w7vo6uhsjitq) **3**

[What is Multiplexing](#_3l7jomtcatpq) 4

[Why do we require multiplexing](#_ycs0ozez12hu) 4

[Different Kinds of Multiplexing](#_pg0p2rtxmgln) 4

[Details of TDM](#_85fi0mlkc96j) 6

[**Working principle**](#_wa04aset5gqw) **9**

[**Discussion**](#_w3y3p44wssyt) **12**

[Application](#_t9b605angby7) 13

[Comparative Analysis](#_x847andd69g0) 14

# Problem Definition

## Defining the problem as what we expected to accomplish

Main task of our lab experiment was to understand the usability,efficiency and usefulness of Synchronous time division multiplexing(sync - TDM) and Statistical time division multiplexing(Stat - TDM).

## The inputs and outputs

To understand the topics we had to perform the following task -

From one server computer we had our five input files.So by maintaining a certain bit rate(B) and time(T) we had to send the files to the client computer and also generate the duplicate type in the client computer.This sending we had to implement with both time division multiplexing system. This was the primary task. And after that, the task was extended. Then we had to connect multiple computers altogether. Then at the same time multiple computers could send the files at a main server computer. This will merge the files from different computers and will send it to another computer. This computer will break down the input frame and again send the input to the computers from which they came from.

This was the final task of our lab on the emulation of Sync - TDM and stat - TDM.

## Equipment required

To solve our problem and implement the communication we needed -

1. Server and Client Computers
2. Java Development Kit(JDK) for language support.
3. Gedit or any type of text editor to create the files, to see and erase.
4. Secured Lan (local area network) support.

# Theoretical background

## What is Multiplexing

Multiplexing(Muxing) is a way of sending multiple signals or streams of information over a communications link at the same time in the form of a single,complex signal. The receiver receives the signal, recovers the separate signals, a process called demultiplexing(demuxing).

## Why do we require multiplexing

There are basically two reasons for multiplexing. They are following -

1. To make it possible for any network device to talk with any other networking device without having to dedicate a connection for each pair. This requires shared media.
2. To make a scarce or expensive resource stretch further. e.g - to send many signals down each cable or fiber strand running between major metropolitan areas, or across one satellite uplink.

Multiplexing combines multiple analog or digital signals bound for transmission through a single communication line or computer channel. This technique has been introduced to increase channel utilization in multicomputer communication systems and time sharing systems and also to reduce the communication cost.

## Different Kinds of Multiplexing

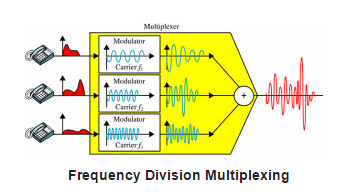
There are basically three types of multiplexing. They are given in following -

1. Frequency division multiplexing(FDM)
2. Wavelength division multiplexing(WDM)
3. Time Division Multiplexing(TDM)

They are discussed briefly in following.

1. **Frequency Division Multiplexing(FDM)**

In the 20th century, many telephone companies used frequency-division multiplexing for long distance connections to multiplex thousands of voice signals through a coaxial cable system. For shorter distances, cheaper balanced cables were used for various systems like bell systems K-and N-carrier, but they didn’t allow large bandwidths. The FDM is an analog multiplexing that combines analog signals. Frequency division multiplexing is applied when the bandwidth of the link is greater than the combined bandwidth of the signals to be transmitted.

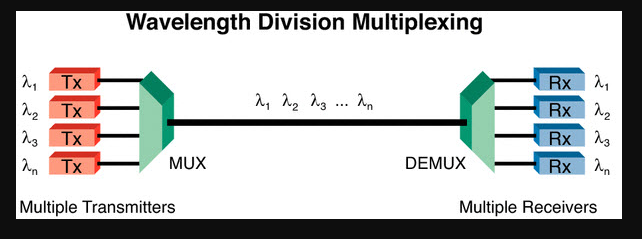


In this type of multiplexing, signals are generated by sending different device-modulated carrier frequencies, and these modulated signals are then combined into a single signal that can be transported by the link. To accommodate the modulated signal, the carrier frequencies are separated with enough bandwidth, and these bandwidth ranges are the channels through which different signals travel. These channels can be separated by unused bandwidth. Some of the examples for the time division multiplexing include radio and television signal transmission.

**b) Wavelength Division Multiplexing(WDM)**

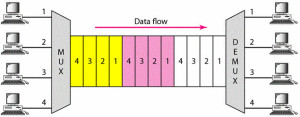
Wavelength division multiplexing (WDM) is a technology in fiber optic communications; and, for the high capacity communication systems, wavelength division multiplexing is the most promising concept. This system uses multiplexer at transmitter to join signals and demultiplexer to split the signals apart, at the receiver end. The purpose of WDM is to combine multiple light sources into a single light source at the multiplexer; and, at the demultiplexer the single light is converted into multiple light sources.

WDM is designed to use the high data rate capability of the fiber optic cable. The data rate of this cable is higher than the metallic transmission cable’s data rate. Conceptually, the wavelength division multiplexing is same as the frequency division multiplexing, except for the transmission through the fiber optic channels wherein the multiplexing and demultiplexing involves optical signals.

****

**c) Time Division Multiplexing(TDM)**

Time division multiplexing is a technique used to transmit a signal over a single communication channel by dividing the time frame into slots – one slot for each message signal. Time-division multiplexing is primarily applied to digital signals as well as analog signals, wherein several low speed channels are multiplexed into high-speed channels for transmission. Based on the time, each low-speed channel is allocated to a specific position, where it works in synchronized mode. At both the ends, i.e., the multiplexer and demultiplexer are timely synchronized and simultaneously switched to the next channel.



## Details of TDM

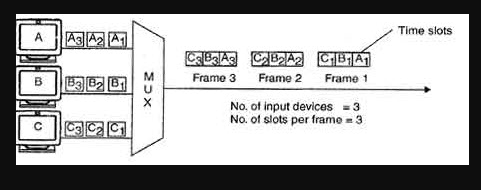
**Types of TDM:**

Can be classified into 4 TDMs

* Synchronous time-division multiplexing
* Asynchronous time-division multiplexing
* Interleaving time-division multiplexing
* Statistical time-division multiplexing

1. **Synchronous Time Division Multiplexing(sync-TDM)**

Synchronous time division multiplexing can be used for both analog and digital signals. In synchronous TDM, the connection of input is connected to a frame. If there are ‘n’ connections, then a frame is divided into ‘n’ time slots – and, for each unit, one slot is allocated – one for each input line. In this synchronous TDM sampling, the rate is same for all the signals, and this sampling requires a common clock signal at both the sender and receiver end. In synchronous TDM, the multiplexer allocates the same slot to each device at all times.

****

**Figure - Sync TDM**

1. **Asynchronous Time Division Multiplexing**

In asynchronous time-division multiplexing, the sampling rate is different for different signals, and it doesn’t require a common clock. If the devices have nothing to transmit, then their time slot is allocated to another device. Designing of a commutator or de-commutator is difficult and the bandwidth is less for time-division multiplexing. This type of time-division multiplexing is used in asynchronous transfer mode networks.

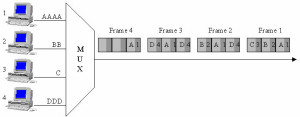
****

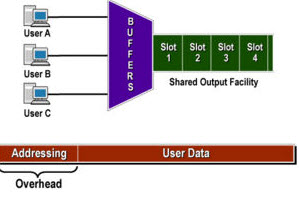
Figure - Asynchronous Time-Division Multiplexing

1. **Interleaving Time Division Multiplexing**

This type of TDM is explained in the latter part of the report.

1. **Statistical Time Division Multiplexing(stat-TDM)**

Statistical time-division multiplexing is used to transmit several types of data concurrently across a single transmission cable. This is often used for managing data being transmitted via LAN or WAN. The data is simultaneously transmitted from the input devices that are connected to the network including printers, fax machines, and computers. This type of multiplexing is also used in telephone switch board settings to manage the calls. Statistical TDM is similar to dynamic bandwidth allocation, and in this type of time-division multiplexing, a communication channel is divided into an arbitrary number of data streams.

****

**Figure - Statistical Time-Division Multiplexing**

## 

**Advantages of TDM over other multiplexing systems**

Basically FDM and WDM is almost theoretically same as they both divide the whole bandwidth or wavelength into multiple parts so that multiple sources can use at the same time except FDM uses normal metallic cable and WDM uses fiber optic cable to travel the data faster.

Now, So their appliance method is almost same. And so we will bound our discussion between FDM and TDM. FDM divides the channel into multiple, but smaller frequency ranges to accommodate more users, while TDM divides a channel by allocating a time period for each channel. Due to this fact, we can say that for TDM, each signal uses all of the bandwidth some of the time, while for FDM, each signal uses a small portion of the bandwidth all of the time. TDM provides greater flexibilityand efficiency**,** by dynamically allocating more time periods to the signals that need more of the bandwidth, while reducing the time periods to those signals that do not need it. FDM lacks this type of flexibility, as it cannot dynamically change the width of the allocated frequency.

# Working principle

1. **Synchronous Time Division Multiplexer (Sync TDM)**

The main idea to implement this multiplexer is given following -

We were given five sources ( five files) from each of which generating data rate or Bit rate was 5KBPs and Total time given was 10 ms. From this calculation we can come to this ideology -

5\*10^3 \* 10 \* 10^(-3) = 50 Bits . 50 bits and the closest character number is 6 (6 \* 8 = 48 bits).

So, actually we made our frame on that hypothesis depending on 5 slots from 5 sources and each slot sending 6 characters. If sufficient character is not found then we denoted as (#) for that empty character as synchronous tdm always have same sized frame and each time slot having the same size of characters. By this way we made our server frame and sent it to the client. We closed our connection when all the data was sent.

At the client side we demultiplexed it almost in the mirror way. Like when we get the each frame, we knew that first time slot came from the first source, second time slot of the frame came from the second source, third came from the third one and by following the same manner the other sources. So

by that ideology we decoded our data and saved it to the respective destinations.

**Pseudo Code -**

**Server Side -**

Open the Server Socket

Establish the socket by server socket’s client ACCEPTING.

open the DataOutputStream.

Open the File readers to read from the files.

Read from the files(5 files) using file readers.

5.1) Read 6 characters from each files.

5.2) If 6 character is not found in the file then the remaining characters of

that time slot is filled with(#) sign.

5.3) After the reading from each source the strings were merged to a total each string

which corresponds to our frame.

6) Sending the string/frame through socket.

7) Repeat from 5 until all the files are not read.

8) Close the socket, readers. Terminate the program.

**Client Side -**

1. Open the client socket to establish connection with server socket.
2. Open the DataInputStreamReader to read from the sockets.
3. Open the FileWriters to write in the file.
4. Get data from the sockets.

4.1) Distribute the frame according to the slot.

4.2) i’th slot of the frame is for i’th source. So write in the defined file destination with FileWriter.

4.3) Write character by character. If the character is # don’t write it in that destination

as it represents NULL character according to our definition.

4.4) Continue till writing the whole frame. Each slot defines one destination’s data. Write in that

destination.

5) Repeat from 4 until all the frames are not found.

6) Close all the sockets, writers and terminate the program.

1. **Statistical Time Division Multiplexer**

Statistical TDM, basically don’t have fixed frame size. Generally in sync TDM we had to maintain the same sized frame each time. But in statistical TDM we don’t have to maintain that field. In fact the main restriction from our problem is at each frame we can have highest five time slots and each time slot can carry highest six characters. If enough character is not found frame size can be reduced and even time slot. So in each time slot we have one character to identify the destination. Then we have a space( ) and the we start our data starting as (\*). This means after this character this slot will have data which will be written in our destination and end of the data will also be defined as \*. so

our main ideology can be defined as like this.

| Destination Identifier | space | \* | D | A | T | A | \* |
| --- | --- | --- | --- | --- | --- | --- | --- |

**Pseudo Code**

**Server Side -**

1. Open the Server Socket
2. Establish the socket by server socket’s client ACCEPTING.

3) Open the DataOutputStream.

4) Open the File readers to read from the files.

5) Read from the files(5 files) using file readers.

5.1) Read 6 characters from each files.Before writing to the sending string first write the destination or source identifier. Then \* to denote the starting of the writing of the data.

5.2) If 6 character is not found then close the reading from this source with \*. If 6 characters are already read then denote the ending with \* also.

5.3) After the reading from each source the strings were merged to a total each string

which corresponds to our frame.

5.4) If any corresponding source don’t possess data then skip the frame won’t have time slot for this source.

6) Sending the string/frame through socket.

7) Repeat from 5 until all the files are not read.

8) Close the socket, readers. Terminate the program.

**Client Side -**

1. Open the client socket to establish connection with server socket.
2. Open the DataInputStreamReader to read from the sockets.
3. Open the FileWriters to write in the file.
4. Get data from the sockets.

4.1) Distribute the frame according to the slot.Each slot has the destination identifier and also data starting and data ending characters(\*). These characters will define the characters for this precise destination. So from each time slot the desired destination and data size can be defined.

4.2) Continue till writing the whole frame. Each slot defines one destination’s data. Write in that destination.

5) Repeat from 4 until all the frames are not found.

6) Close all the sockets, writers and terminate the program.

1. **The Modification of Lab - o5 Project**

Here, basically it is an advanced application of the previously discussed synchronous and statistical TDMS.

The task was, we will be given three source computers each of them will have one file and data will arrive

from each source at a different speed to our main computer(named server computer as ISP02) . This computer

will merge the files in one frame and send it to another client computer(named client computer as ISP03).

And then the client computer will decode the frame and send it to it's defined destination.

So our idea was to implement the task by following methods and procedure.

public class PCSourceSent

1) PC Source class will collect data from the source file.

2) It will connect the data with it's socket address through which it

got connected with the main server. Example - by following format

[SOCKET\_ADDRESS][SPACE][\*][D A T A][\*]

the first \* means starting of the data and second \* means the ending of the reading of data.

3) The messages will be sent through socket. The highest character could be got is six(which will be sent at one pass)

4) If there is less than six character in the file. It will be sent. So frame size can varry like STATtdm.

5) After all the data sent the program will terminate.

public class ModificationServer

1) In the modification class there will be two threads.

1.1) Thread Connection can always establish connection.

1.2) Thread Receive will always get the messages from the sources and will be saved in an array of string.

1.3) This two threads will be under class controller.

1.4) After all the reading, the array of string will be sorted on the basis of connection establishment.

2) At the time of sending data to the ISP02 the, the controller will built the frame. Each frame will be built

by giving time slot to each source. If the source contains data then.

3) After the frame setup it will be sent.

4) After all the frames been set this program will terminate.

public class ClientServer

1) This server will be connected to main Server. It will receive the frame and decode it and and from each address

or from each time slot it will send, it will select the destination address and there the data will be sent.

2) After sending all the data this will terminate.

public class PCSourceReceive

1) This class will just receive data from the client server and then just write into file.

After the completion of writing this program will terminate.

Main Challenges and Solutions-

Sync TDM vs Stat TDM : Why in general Stat TDM is better that sync TDM

In synchronous TDM time slot and whole frame size is fixed. And it's actually a very big problem. Because it can

happen some files are much larger than other files and so, if this happens extra time slots will be wasted and

also extra additional bits will be unuseful. To solve this issue, statistical TDM helps. Because in this,

we can say that if there are no data left in some sources, then this slot will be given to another source and

time slot will be properly utilized and also as if there is not enough data to fill up the total slot then

automatically time slot will reduce and this will make the best use of time optimization.

**Sync TDM vs Stat TDM : Reverse Case**

In general cases and in fact almost all cases STAT TDM works better than sync TDM, But in stat TDM it can be found that the as the sources which generates more data gets larger time slot, this can cause some problems. The most useful example where sync TDM is better than Stat TDM is while voice calling. During voice calling, constant stream of data and right ordering is very important and for this case Synchronous TDM gives better facilities.

# Discussion

## Application:

The purpose of Multiplexing is optimization. Many signals are pushed on the carrier as possible to get reasonable return on investment. Communication speed will not increase. But multiplexing the already multiplexed signals gives rise to high bit rate. In TDM, the time slot is shared by different inputs( calls) for example, 24 or 32 channels in one 125 microsecond time frame. That means 24/32 simultaneous calls can be routed together in that frame without giving rise to any cross talk.

The communication although digital will be modulated as PCM (Pulse Code Modulation) as there is no way a bit can be directly sent. The PCM waves will be sent via cable after modulation – done after multiplexing – and at the receiver it is demodulated back and demultiplexed into the various channels, and at each channel the digital data is converted back to analog (like voice, for example) for the receiver.

For digital transmission of several telephone calls over the same four-wire copper cable (T-carrier or E-carrier) or fiber cable in the circuit switched digital telephone network

* The synchronous digital hierarchy (SDH)/synchronous optical networking (SONET) network transmission standards that have replaced PDH.
* The Basic Rate Interface and Primary Rate Interface for the Integrated Services Digital Network (ISDN).
* The RIFF (WAV) audio standard interleaves left and right stereo signals on a per-sample basis

TDM can be further extended into the time division multiple access (TDMA) scheme, where several stations connected to the same physical medium, for example sharing the same frequency channel, can communicate. Application examples include:

* The GSM telephone system
* The Tactical Data Links Link 16 and Link 22

In circuit-switched networks, such as the public switched telephone network (PSTN), it is desirable to transmit multiple subscriber calls over the same transmission medium to effectively utilize the bandwidth of the medium. TDM allows transmitting and receiving telephone switches to create channels (*tributaries*) within a transmission stream. A standard DS0 voice signal has a data bit rate of 64 kbit/s.A TDM circuit runs at a much higher signal bandwidth, permitting the bandwidth to be divided into time frames (time slots) for each voice signal which is multiplexed onto the line by the transmitter. If the TDM frame consists of *n* voice frames, the line bandwidth is *n*\*64 kbit/s.

Numerous chips, processors, and hardware devices around the world support Time Division Multiplexing, not least of which includes the 4 Channel TDM PCM Transmitter and Receiver Scientech2804. It is an educational device, quite like a trainer board. According to Its product description it is said that Scientech TechBooks are compact and user-friendly learning platforms to provide a modern, portable, comprehensive and practical way to learn Technology. Each TechBook is provided with detailed Multimedia learning material which covers basic theory, step by step procedure to conduct the experiment and other useful information. It can be ordered to be brought up to Bangladesh through contacting the owner. Other than this, too, several such training tools for understanding TDM exist.

There are a number of ways to implement TDM in terms of hardware. The most direct way is to use a multiplexer and a demultiplexer and use a timing circuit to give the selector pins the inputs. However an easy approach is to use the arduino as a timing circuit as well as a demux. This decreases the probability of error in the output, makes it easier for us to observe the output on a serial monitor and also removes an entire IC from the circuit, making the soldering and connecting much easier.

Components required are: LEDs, resistors, IC 74150 mux, 8-bit DIP switch, Arduino microcontroller board, connecting wires and jumper cables.

All these components are easily available in Bangladesh.

## Comparative Analysis

1. **Different implementations and versions:**

Time division multiplexing is classified into four types:

* Synchronous time-division multiplexing
* Asynchronous time-division multiplexing
* Interleaving time-division multiplexing
* Statistical time-division multiplexing

**Interleaving**

Time-division multiplexing can be visualized as two fast rotating switches on the multiplexing and demultiplexing side. At the same speed these switches rotate and synchronize, but in opposite directions. When the switch opens at the multiplexer side in front of a connection, it has the opportunity to send a unit into the path. In the same way, when the switch opens on the demultiplexer side in front of a connection that has the opportunity to receive a unit from the path. This process is called interleaving.



1. **Why stat TDM performs better than sync TDM?**

-In Synchronous TDM data flow of each input connection is divided into units and each input occupies one output time slot. In Statistical TDM slots are allocated dynamically. i.e. input line is given slots in output frame if and only if it has data to send. So slots are not wasted.

-In Synchronous TDM no. of slots in each frame are equal to no. of input lines. In Statistical TDM, No. of slots in each frame are less than the no. of input lines. Thus less space is required.

-In Synchronous TDM buffering is not done, frame is sent after a particular interval of time whether someone has data to send or not. In Statistical TDM Buffering is done and only those inputs are given slots in output frame whose buffer contains data to send. So frames are not wasted.

-Slots in Synchronous TDM carry data only and there is no need of addressing. Slots in Statistical TDM contain both data and address of the destination.

-In sync-TDM, Max. Bandwidth is utilized if all inputs have data to send. The capacity of link is normally is less than the sum of the capacity of each channel in stat-TDM.

-In Synchronous TDM demultiplexer at receiving end decomposes each frame, discards framing bits and extracts data unit in turn. This extracted data unit from frame is then passed to destination device.

In Statistical TDM demultiplexer at receiving end decomposes each frame by checking local address of each data unit. This extracted data unit from frame is then passed to destination device.

-Synchronization bits are used at the beginning of each frame of the sync-TDM. No synchronization bits are used in stat-TDM.

Statistical TDM cannot provide fixed waiting time guarantee. But a Synchronous TDM can. In this case Synchronous TDM will perform better.

1. **Performance analysis:**

**i) Throughput**

Throughput is a measure of how many units of information a system can process in a given amount of time. It is applied broadly to systems ranging from various aspects of computer and network systems to organizations. Related measures of system productivity include , the speed with which some specific workload can be completed, and response time, the amount of time between a single interactive user request and receipt of the response.

Historically, throughput has been a measure of the comparative effectiveness of large commercial computers that run many programs concurrently. An early throughput measure was the number of batch jobs completed in a day. More recent measures assume either a more complicated mixture of work or focus on some particular aspect of computer operation. Units like "trillion floating-point operations per second (TeraFLOPs or TFLOPS)" provide a metric for comparing the cost of raw computing over time or by manufacturer. A benchmark can be used to measure throughput. In data transmission, network throughput is the amount of data moved successfully from one place to another in a given time period, and typically measured in bits per second (bps), as in megabits per second (Mbps) or gigabits per second (Gbps).

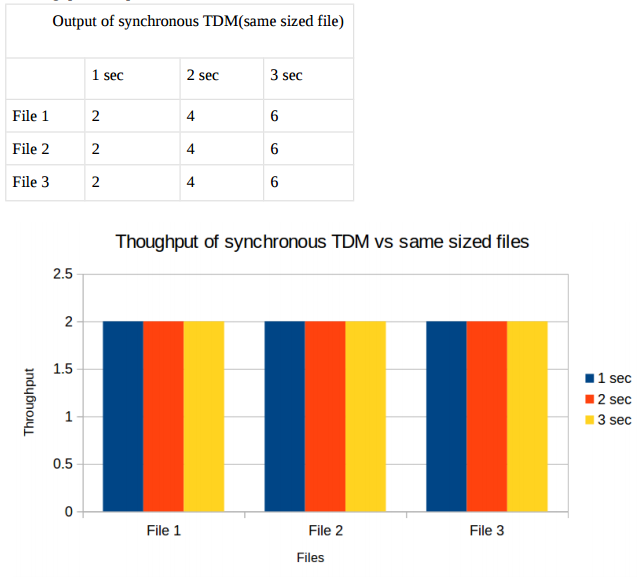
Likewise, in storage systems, throughput refers to either the amount of data that can be received and written to the storage medium or read from media and returned to the requesting system, typically measured in bytes per second (Bps). It can also refer to the number of discrete input or output (I/O) operations responded to in a second (IOPS).

**ii) How to measure throughput of sync TDM and stat TDM**

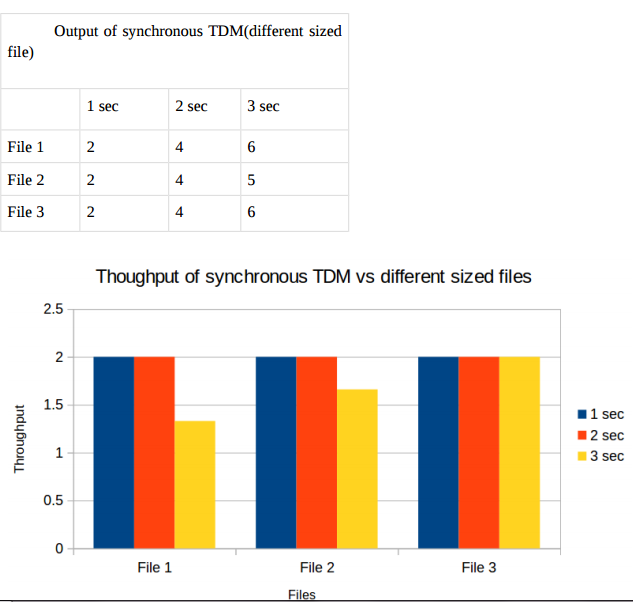
according to the definition of throughput we can say, to measure the throughput of this two multiplexing system we can use time and data pass rate and procedure. In simple words, we always pass data from the source to destination. And so, the system which can pass data with minimal time will be regarded as the efficient data transportation system.

**iii) Graphs:**

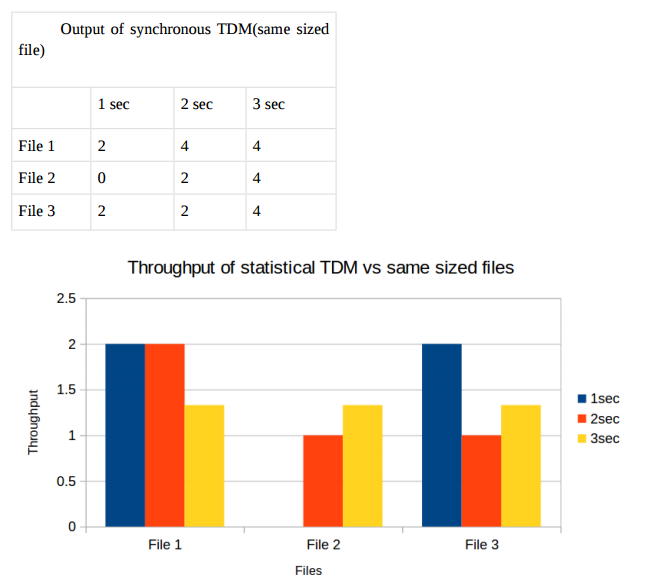
1. **Throughput of Synchronous TDM vs same sized files**

****

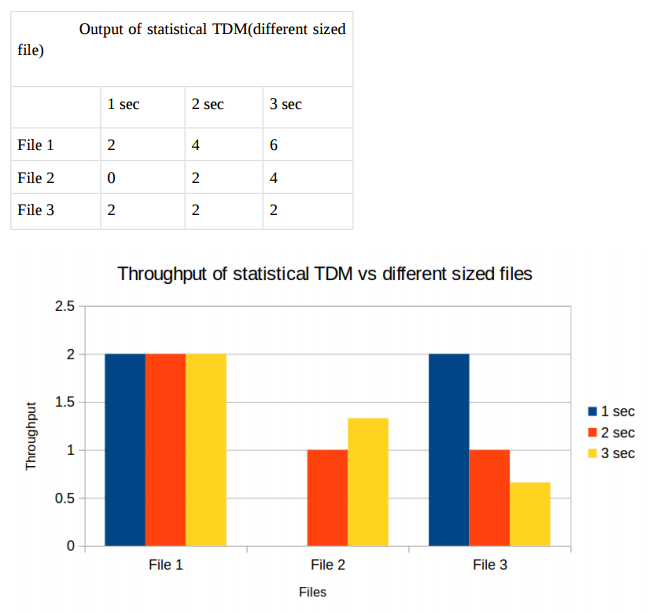
1. **Throughput of Synchronous TDM vs different sized files**

****

1. **Throughput of Statistical TDM vs same sized files**

****

1. **Throughput of Statistical TDM vs different sized files**

****