# EE450 Lab4

June 17th, 2019

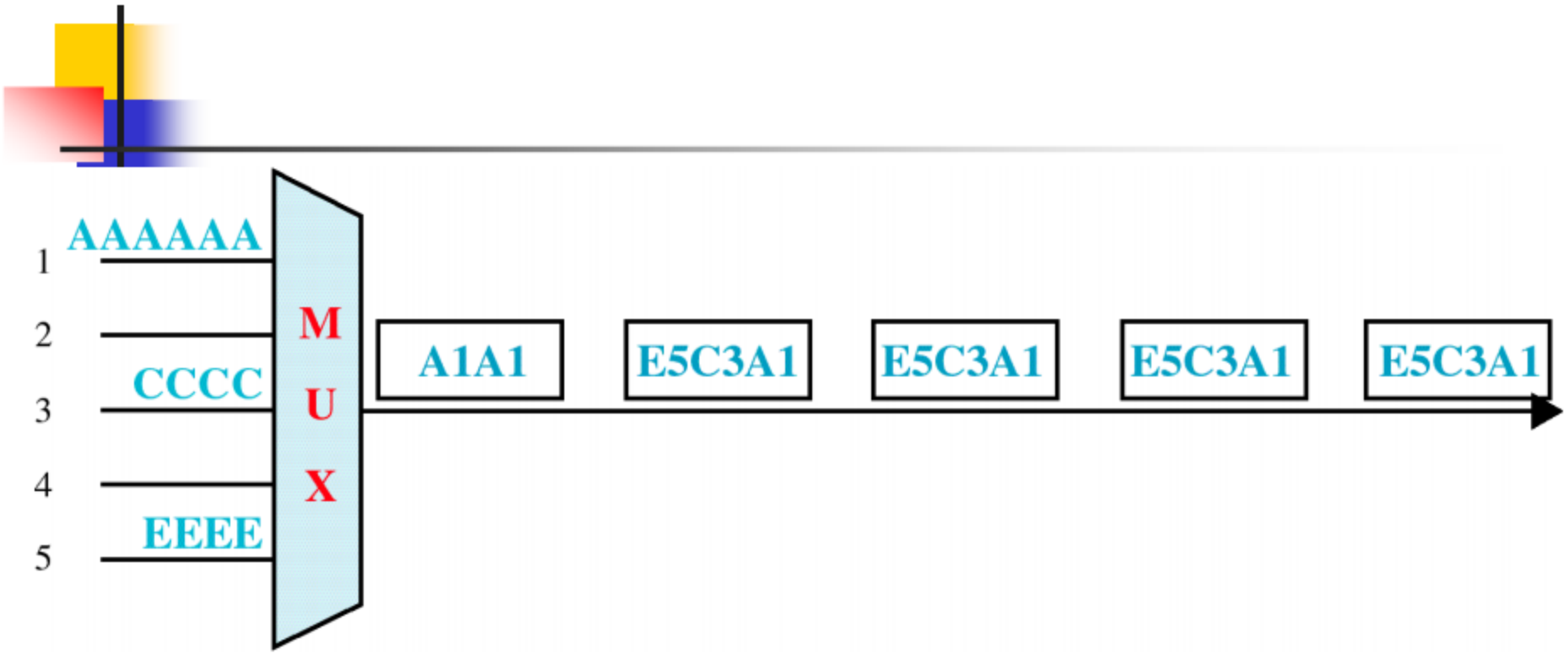
# STDM

- Statistical Time Division Multiplexing
- Similar to regular TDM but different in this:
  - Traffic is sent on demand – Only if there is data on line 1, will slot 1 be occupied by line 1
  - Resources are not guaranteed
- If we are no longer guaranteed a time slot why use it?
  - We (the carrier) can take advantage of one of the input lines not being busy
- Important distinction – STDM is used mainly for Digital Lines

# STDM (Asynchronous TDM)

- How does it Work?
  - Checks to see if there is data to transmit on input line
    - If there is transmit data
    - If not move on to next input line

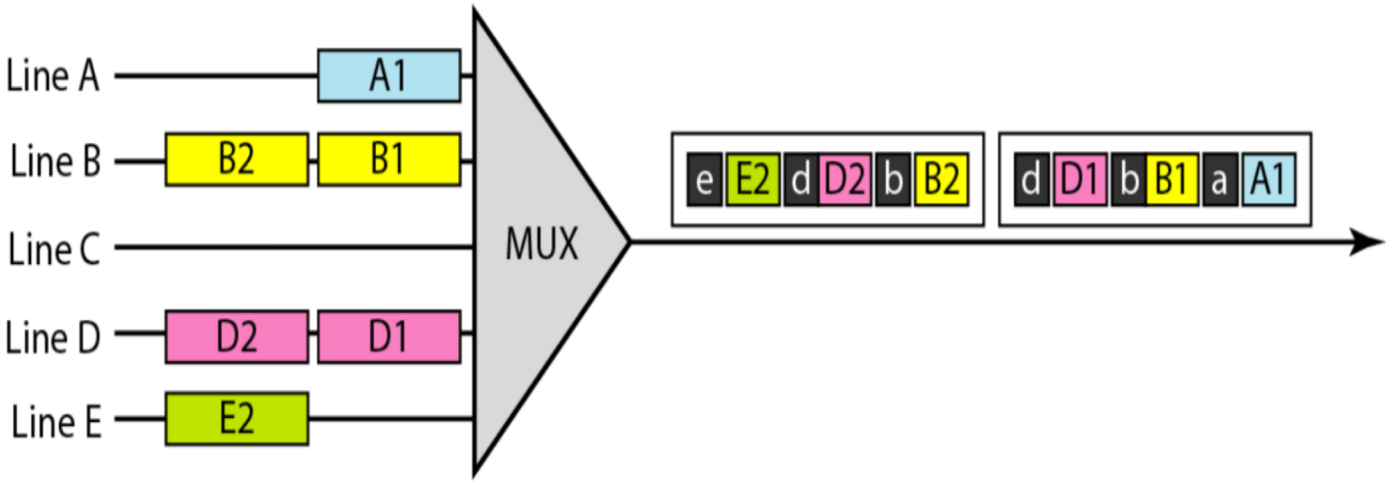
# Frames and Addresses




a. Only three lines sending data

# Lab 4

- Statistical TDM works by calculating the average transmission rates of the streams to be combined, and then uses a high-speed multiplexing link with a transmission rate that is equal to (or slightly greater than) the statistical average of the combined streams. we dynamically assign the appropriate number of slots to accommodate the current transmission rates from each stream. Because the combined rate of all the streams will also fluctuate in time between two extreme values, we need to buffer the output of the low-bit-rate streams when the combined rate exceeds the transmission rate of the high-speed link



# Lab 4

- Prior to transmission, we divide each stream of bits coming from a source into fixed-size blocks. We then add a small group of bits called a header to each block, with the header containing the addresses of the source and intended user for that block. The block and the header are then transmitted together across the channel. Combined, the block and header are called a sub-frame. Sub-frames from multiple sources form a frame.

# Lab 4

**Your tasks for this lab:**

- Implement Statistical TDM in either C or C++.
- Your compiled C/C++ program should be able to read the sources described in an input file. The file name should be an input parameter of your complied program. Suppose your compiled C/C++ program is STDM. A command of running your program can be: “./STDM input.txt” where “input.txt” is the name of the input file.

# Lab 4

An example of input file is as follows:

**SourceA: 0 1 A1, 1 2 A2, 2 3 A3, 5 6 A4**

**SourceB: 0 1 B1, 1 2 B2**

**SourceC: 1 2 C1, 2 3 C2, 3 4 C3, 5 6 C4, 6 7 C5**

**SourceD: 4 5 D1, 5 6 D2, 8 9 D3**

- Assume all sources have the same data rate but have different fraction of time transmitting. The data blocks are separated by comma in each line. For example, in the first line, SourceA sends data block A1 with start time of time 0, and end time of time 1, and sends data block A2 with start time of time 1, and end time of time 2, sends data block A3 with start time of time 2, and end time of time 3, sends data block A4 with start time of time 5, and end time of time 6. For simplicity, all time stamps are integer. The first source has the highest priority, and the next line follows. The last line has the lowest priority

# Lab 4

- Calculate the average transmission rates of the streams to be combined in the input file, and then uses a high-speed multiplexing link with a transmission rate that is equal to (or slightly higher than) the statistical average of the combined streams. The address bits are determined by the number of input sources. The frame time slot is determined by yourself and can be determined by the transmission rate of the output channel.


Sub-frame with multiple sources per frame

# Lab 4

- Define the input buffer and output buffer of the statistical multiplexer by yourself. Input buffers smooth fluctuations in input data rates. Headers containing addresses are added to the data blocks to form sub-frames. Sub-frames are then fed into the output buffer.
- Sub-frames from the various sources form frames. In this part, you need to output the frame. You should include the flags of the frame, the address and the time duration of the frame in your output. You can decide the output format by yourself.


Sub-frame with multiple sources per frame

# Lab 4

- For simplicity, the number of input sources is assumed to be 4. Thus, two bits are required for address. We can ignore the overhead of address bits and flag when calculate the output transmission rate and schedule the output frame.

- Extra credit (10%): The number of input sources is not deterministic in the input file.


Sub-frame with multiple sources per frame

# Lab 4

- Example output:

The average transmission rates of the streams to be combined in the input file is 14/9. We choose the 2 as the output link transmission rate.

SF
0, 1 1.5 A1
1, 1.5 2 B1
0, 2 2.5 A2
1, 2.5 3 B2
2, 3 3.5 C1
0, 3.5 4 A3
2, 4 4.5 C2
2, 4.5 5 C3
3, 5 5.5 D1
0, 6 6.5 A4
2, 6.5 7 C4
3, 7 7.5 D2
2, 7.5 8 C5
3, 9 9.5 D3
EF