Homework 2

By: Ryan Scott

**Protocol 1 Unrestricted Simplex Protocol:**

**Assumptions:**

1. Simplex – Communication is one direction only
2. Reliable Channel – The communication channel never introduces errors (i.e. no duplicate, no missing, or damaged frames)
3. Receiver never gets behind, is can process incoming data infinitely fast (no need for flow control)

Important points of note are:

1. Independent processes – Both sender and receiver are independent processes whether running on different or the same machine
2. Infinite Execution – Both processes are assumed to start at the same time, and forever due to the while(true) execution loop in each
3. Frame – The data link layer passes frames between the physical layer. A frame contains a network layer packet.
4. Packet – The data link layer passes packets between the network layer. A packet is network layer data.



This output uses these default settings for the constants:

Text

Description automatically generated

Output for these settings:

Graphical user interface, text

Description automatically generated

All the data in the .txt file is transmitted in less than a second.

Text

Description automatically generated

This test will be changing the timeout:

* For 10 ms, the output was the same as the default: all the data was transferred in less than a second.
* For 2500 ms, same thing at the 10 ms.
* For 5000 ms, same thing as the first two.

With these results, it’s fair to say timeout doesn’t impact protocol 1 too much as long as it has the appropriate lostframes and buffersize settings.

Text

Description automatically generated

This test will be changing the buffersize:

* For buffersize 1, the output is the same as the default
* For buffersize 1000, the output is the same as the default
* For buffersize 2000, the output is the same as the default

With these results, it’s fair to say buffersize doesn’t impact protocol 1 too much as long as it has the appropriate timeout and lostframes settings.

Text

Description automatically generated

This test will be changing the amount of lostframes:

* For 5 lostframes: the output is the same as the default.
* For 10000 lostframes: the output is the same as the default.
* For 20000 lostframes: the output is the same as the default.

With these results, it’s fair to say lostframes doesn’t impact protocol 1 too much as long as it has the appropriate timeout and buffersize settings.

In conclusion, Protocol 1 sends data so quickly and relentlessly that even major changes in one setting doesn’t affect reliability.

**Protocol 2 A Simplex Stop-and-Wait Protocol:**

**Key Assumptions:**

1. Simplex – Communication is one direction only
2. Reliable channel – The communication channel never introduces errors (i.e. no duplicate, no missing, or damaged frames)

**Improvements:**

1. Flow Control – by requiring sender to wait for a confirmation from receiver.

**Problems:**

1. Deadlock – when either transmit or confirmation frame lost. No way to stop waiting on either sender2 or receiver2.
2. Delay – while sender2 waits for confirmation, under-utilizing the channel

Default settings for Protocol 2:

Text

Description automatically generated

Text

Description automatically generated

The results for the default settings are the data being transmitted in a couple seconds, slightly slower than Protocol 1.

However, now the constants need to be adjusted to test for different circumstances:

Text

Description automatically generated

This test will be changing the timeout:

* For 10 ms, the data is transferred slightly faster than this default
* For 2500 ms, same result as 10 ms
* For 5000 ms, the data is sent slightly faster than 10 and 2500 ms

With these results, it’s fair to say the larger the timeout is, the faster the data will be transferred in Protocol 2.

Text

Description automatically generated

This test will be changing the buffersize:

* For buffersize 1, the data is sent slightly faster than the default.
* For buffersize 1000, the result is the same as the buffersize 1.
* For buffersize 2000, the result is the same as the buffersize 1.

With these results it’s fair to say, unless the buffersize is about 13 (the default), buffersize doesn’t slow the data transfer at all.

Text

Description automatically generated

This test will be changing the lostframes:

* For 5 lostframes, the result is slightly faster than the default.
* For 10000 lostframes, the result is the same as 5 lostframes.
* For 20000 lostframes, the result is the same as 10000 and 5 lostframes.

This these results, it’s fair to say when lostframes is increased from 0, the data is transferred faster.

In conclusion, making major changes to a single setting at a time all seem to cause the data to flow faster.

**Protocol 3 A Simplex Protocol for a Noisy Channel:**

**The Assumption is:**

1. Simplex – Data communication is one direction only.

**The Assumption is:**

1. Reliable Channel – The communication channel never introduces errors (i.e. no duplicate, no missing, or damaged frames)

Default settings for Protocol 3:

Text

Description automatically generated

Output for the default:

Text

Description automatically generated

The result for the default is no data is transferred.

Text

Description automatically generated

This test will be changing the timeout:

* For 10 ms, the result is the same as the default.
* For 2500 ms, the result is the same as the default.
* For 5000 ms, the result is the same as the default.

With these results, it’s fair to say, if data is failing to send; changing the timeout isn’t enough to fix the issue.

Text

Description automatically generated

This test will be changing the buffersize:

* For buffersize 1, the result is the same as the default.
* For buffersize 1000, the result is the same as the default.
* For buffersize 2000, the result is the same as the default.

With these results, its fair to say changing the buffersize doesn’t fix the data not sending issue.

Text

Description automatically generated

This test will be changing the lostframes:

* For 5 lostframes, the result is the same as the default.
* For 10000 lostframes, the result is the same as the default.
* For 20000 lostframes, the result is the same as the default.

With these results, it’s fair to say that changing the lostframes alone doesn’t having the sending issue.

**So what will fix the sending issue:**

At the moment, I’m not sure.

**Protocol 4:**