Cairo University
Faculty of Engineering
Department of Electronics and Electrical
Communications Engineering

3rd Year
Advanced Microprocessor Architecture (ELC 3030)
Project – Fall 2021

ELC_3030
Advanced Microprocessor Architecture
Term Project

Team Report:

Team Members		Section	BN
1-	محمد أيمن عبد العليم أحمد	3	36
2-	محمد حسام عثمان يسن	3	37
3-	محمود موسى جاد موسى	4	6

UART & USB SEMULATOR

UART:

The Assumptions taken:

*The number of data bits per packet:

-we read the input text data we transfer each character in the file into its asci equivalent in binary.

-Our code can handle whether in our config file we decided that the data bits in the packets are 7 or 8 bits.

In case of 7 data bits: we have each character after we transfer it into binary written into 7 bits .so, each packet we transfer 1 character.

In case of 8 data bits: we have each character after we transfer it into binary written into 7 bits .so, each packet we transfer 1 character, and the eighth bit is the LSB of the next character until the last packet we fill the remaining data bits in zero.

*Parity:

-It can be none, even, odd depends on what is written in the config file. If we have even or odd parity it will be stored in1 bit.

*Final shape:

Start bit (1 bit =0) → data bits (7 or 8 bits) →parity bit (1 bit if exist) → stop bit (1bit=1)

Results:

Draw the first 2 bytes of UART

-when we send data size (7 bits) with odd parity(1bit).

1(start=0) +7(data)+1(parity)+1(stop bit=1) =10 bits. We will send two characters.

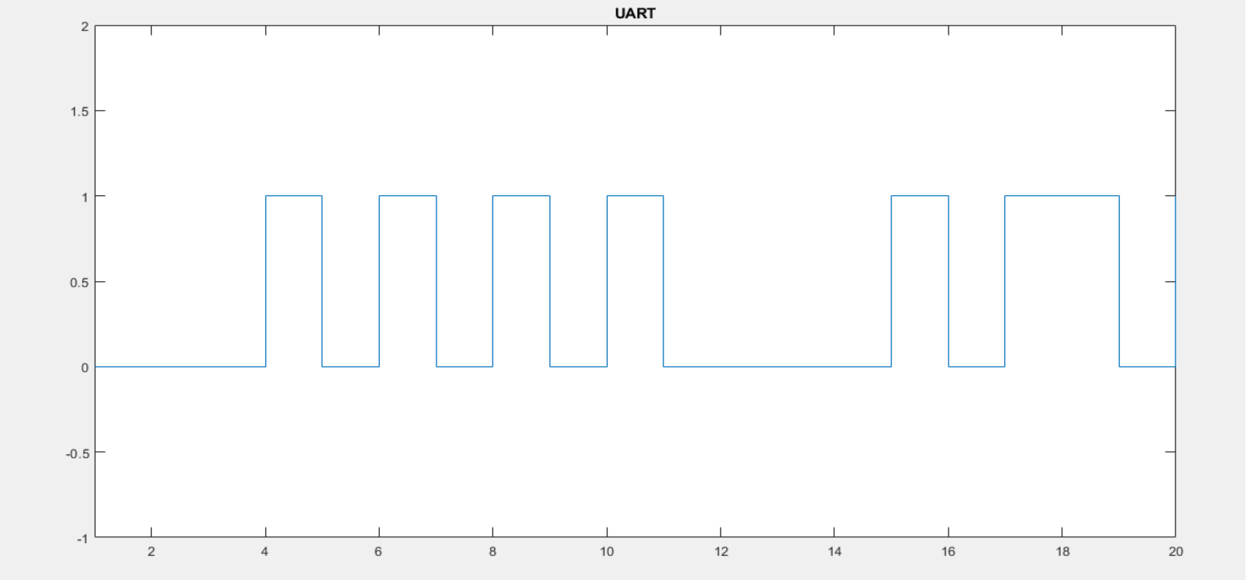

Figure 1 First 2 packets when data size(7) & parity (odd)

- when we send data size (8 bits) without parity.
1(start=0) +8(data)+1(stop bit=1) =10 bits. We will send two characters.

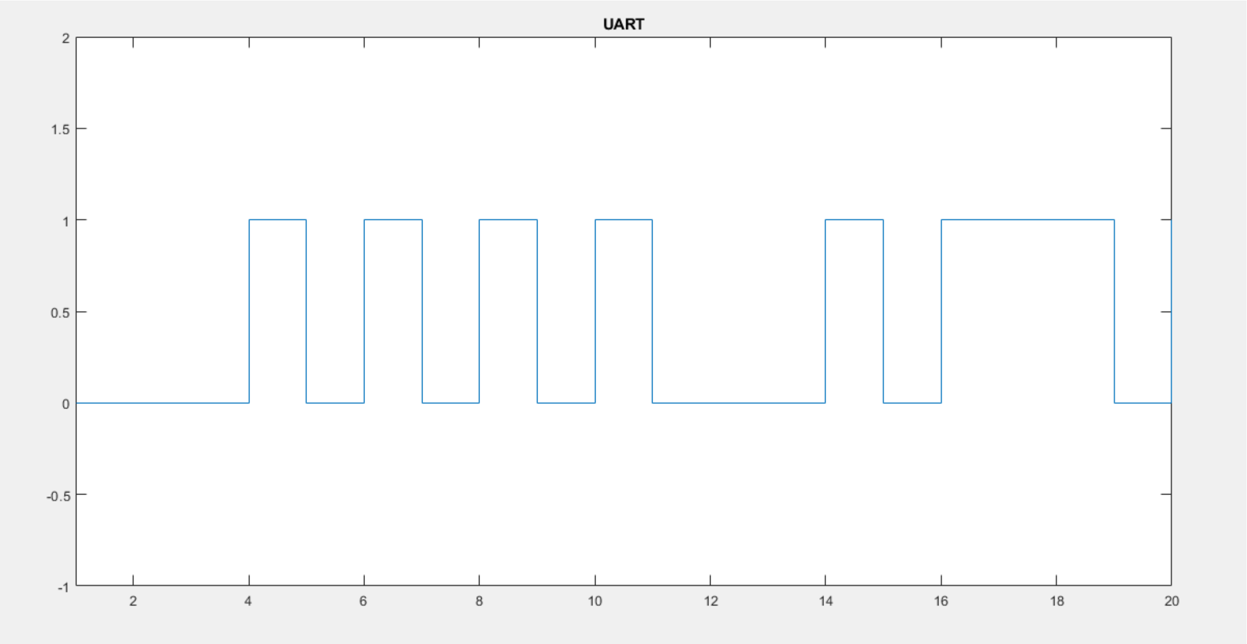

Figure 2 First 2 packets when data size(8) & parity (none)

- when we send data size (8 bits) with even parity (1 bit) parity.
1(start=0) +8(data)+1(even parity) +1(stop bit=1) =10 bits. We will send two characters.

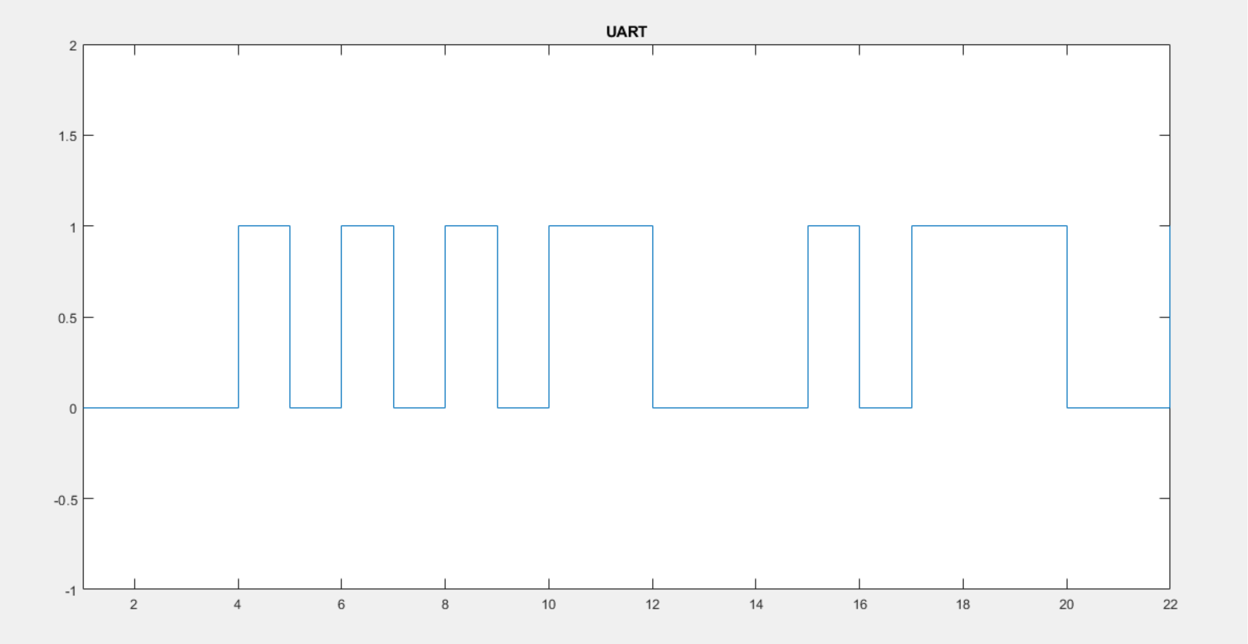

Figure 3 First 2 packets when data size(8) & parity (even)

For The original input text data:

	Efficiency	Overhead	Transmission time
7 bits with odd parity	0.7	0.3	1.2800 sec
8 bits with none	0.8	0.2	1.1200 sec
8 bits /even parity	0.7273	0.2727	1.2320 sec

-in 8 bits we have lower transmission time because in each packet we send 8 bits (7 from the first character and 1 from the next one) so, not just one character in the packet like the case with 7 bits.

For Data size (8 bits) & parity (none):

-we add Two characters every time and calculate the efficiency so after 640 times we will have 1280-character adder equal to the file size.(X-axis explanation)

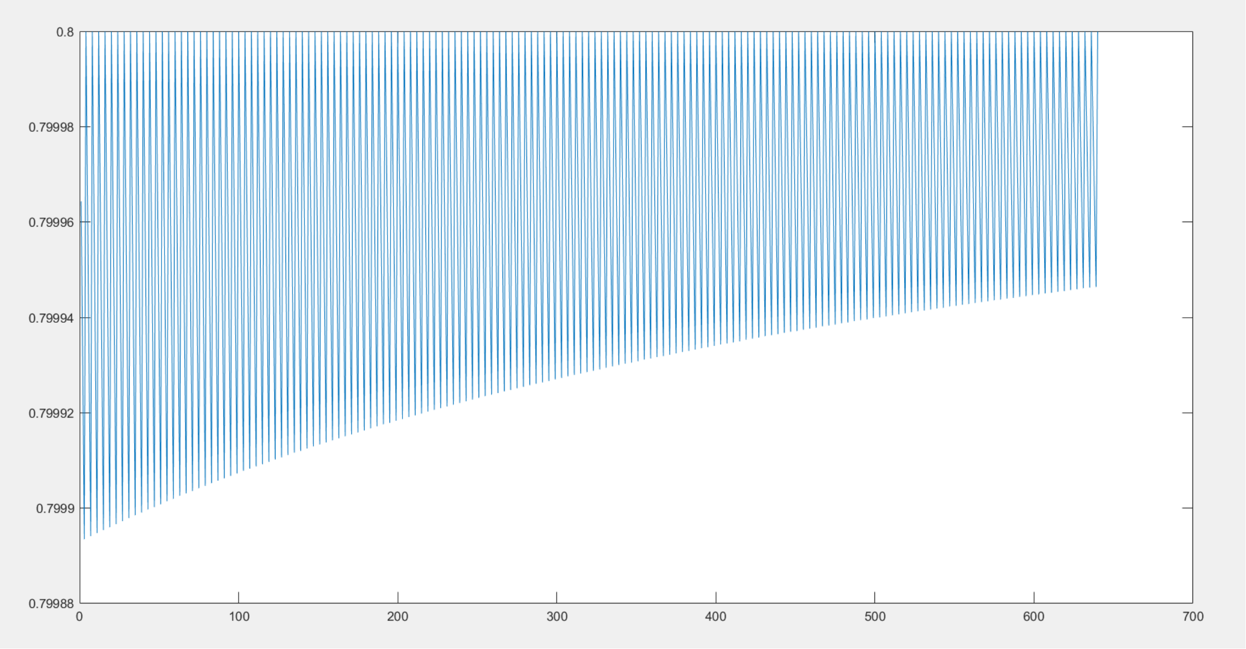

Figure 4 efficiency vs file size

when we operate in the original file size, we have 1280 char*7=8960 bit.

So, If we send each time 8 bits per packet, we will need 1120 packet to send them all. And in this case with original file, we will have the maximum efficiency because the last packet will be completely filled and we won't have to complete it with zeros.

But when we add characters to the original file the efficiency will decrease fist.

For example, if we add 2 characters then we will add 14 bits and then complete the last packet with two zeros so, the efficiency will decrease.

And so on efficiency will keep varying as some times the last packet will be completely filled with data and some times we will have to complete it with zeros as it is partially filled with data.

-

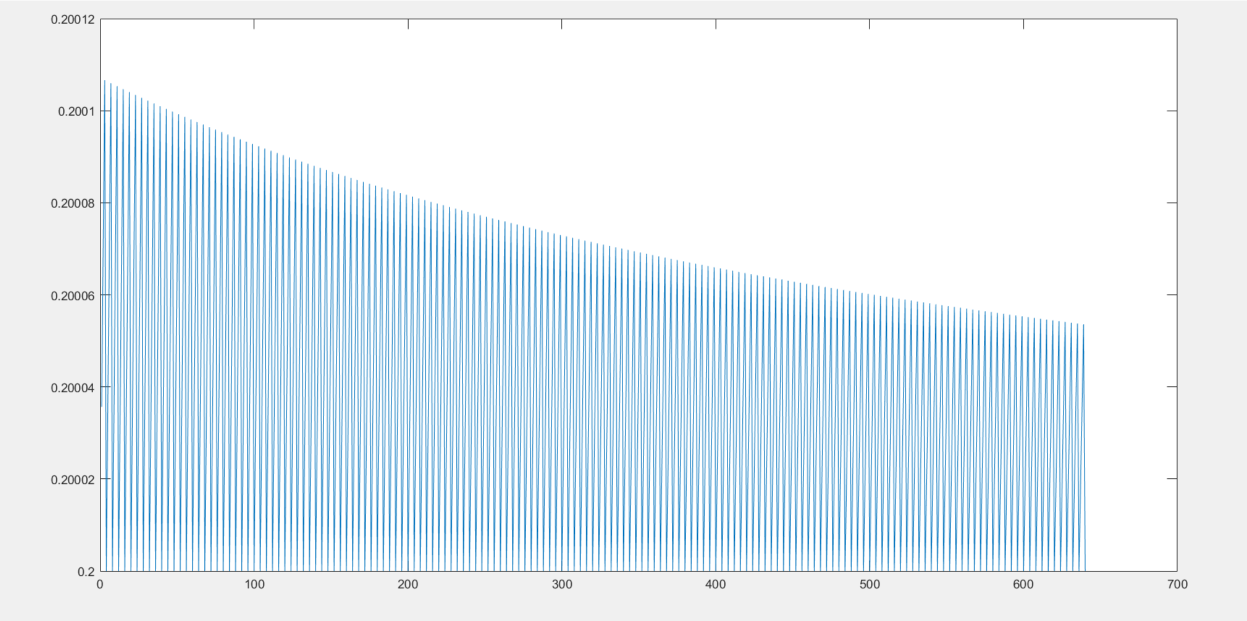

Figure 5 overhead vs file size

Overhead=1-efficieny
So, in points of maximum efficiency we will have minimum overhead and the opposite.

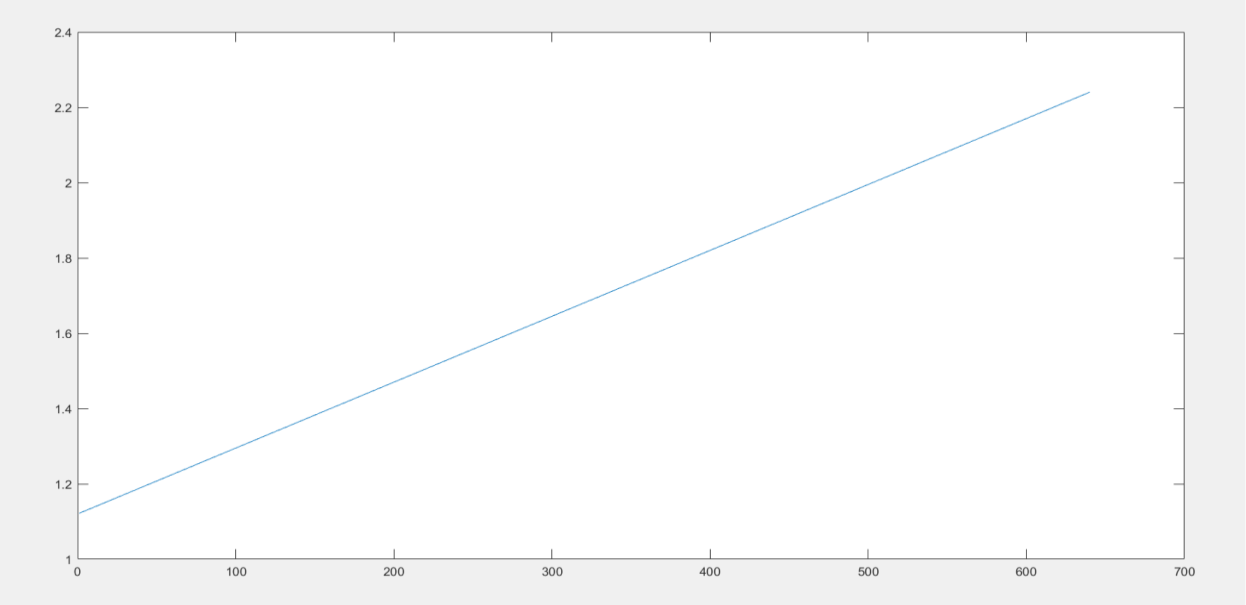

Figure 6 Transmission time vs File size

-Transmission time increase when we add new characters because number of bits transmitted increase so we will need more time.

USB:

*The number of data bits per packet:

-we read the input text data we transfer each character in the file into its asci equivalent in binary.

-we have each character after we transfer it into binary written into 7 bits, so we add 1 zero in the MSB so each character is now written in byte.so, each packet we transfer 128 characters.

-In our code we handled the case if we have for example 130 byte that the first packet will take 128 byte and the second packet will take the remaining two bytes, but we will not fill the remaining 128 bytes of the second packet with zeros.

*Final shape:

Idle (1 bits =1) → synchronization pattern (8 bits: 7 zeros followed by a one) →PID (8 byte) →Destination address (11 bit if exist) → data bits (128 byte) →EOP (2 bits=0)

Draw the first two packets of USB

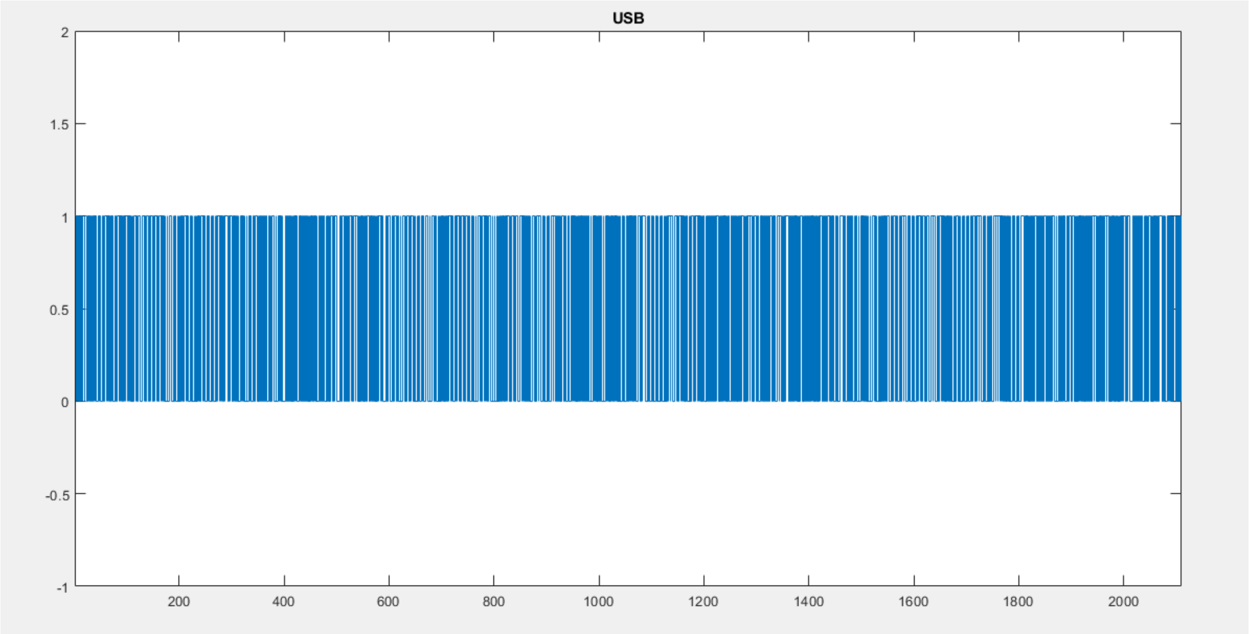

Figure 7 USB (+) Transmitted data for the first two packets

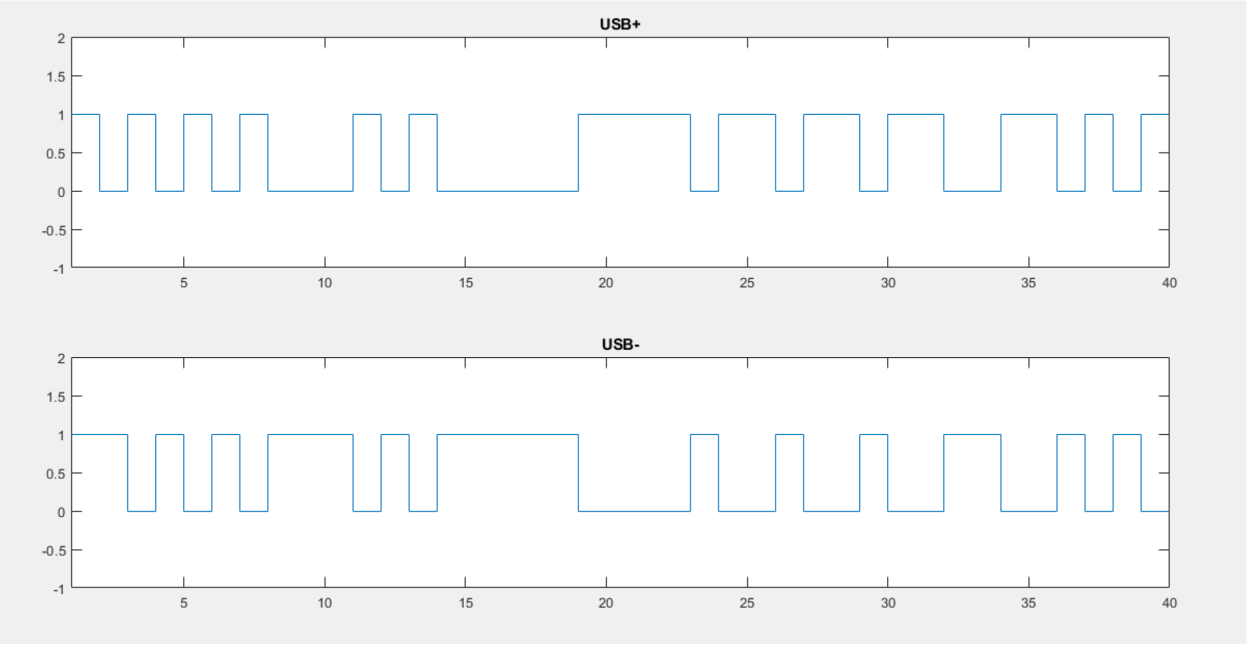

Figure 8 USB (+) & USB (-) First 40 bits

note: we start with one idle bits.

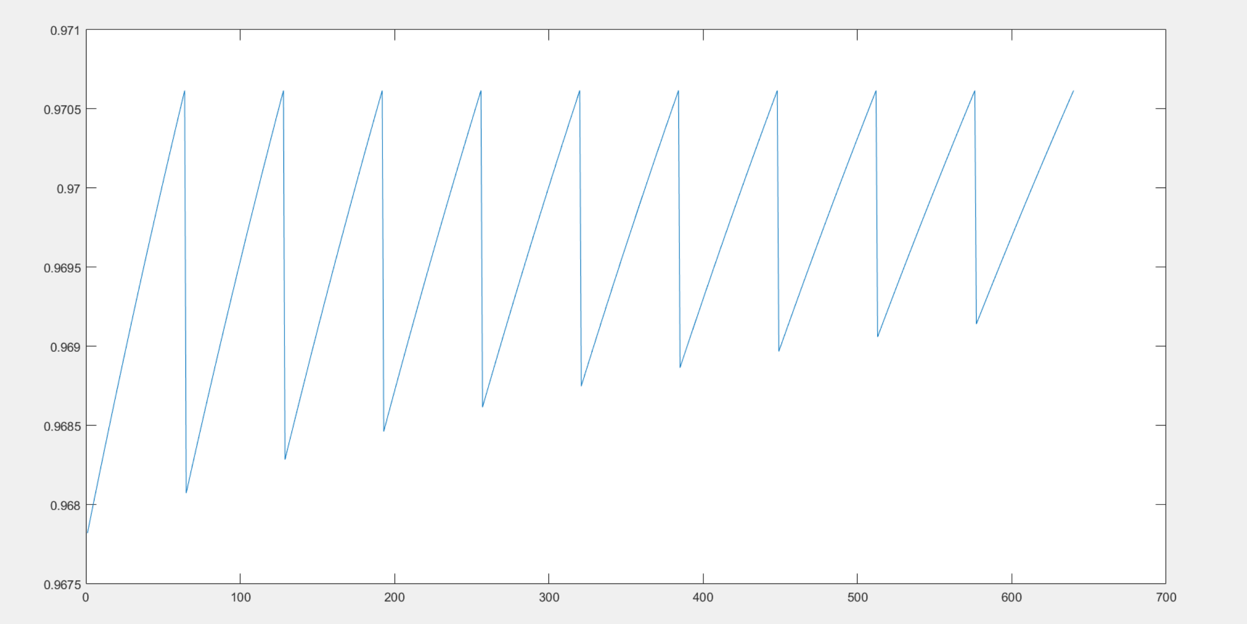

Figure 7

aa

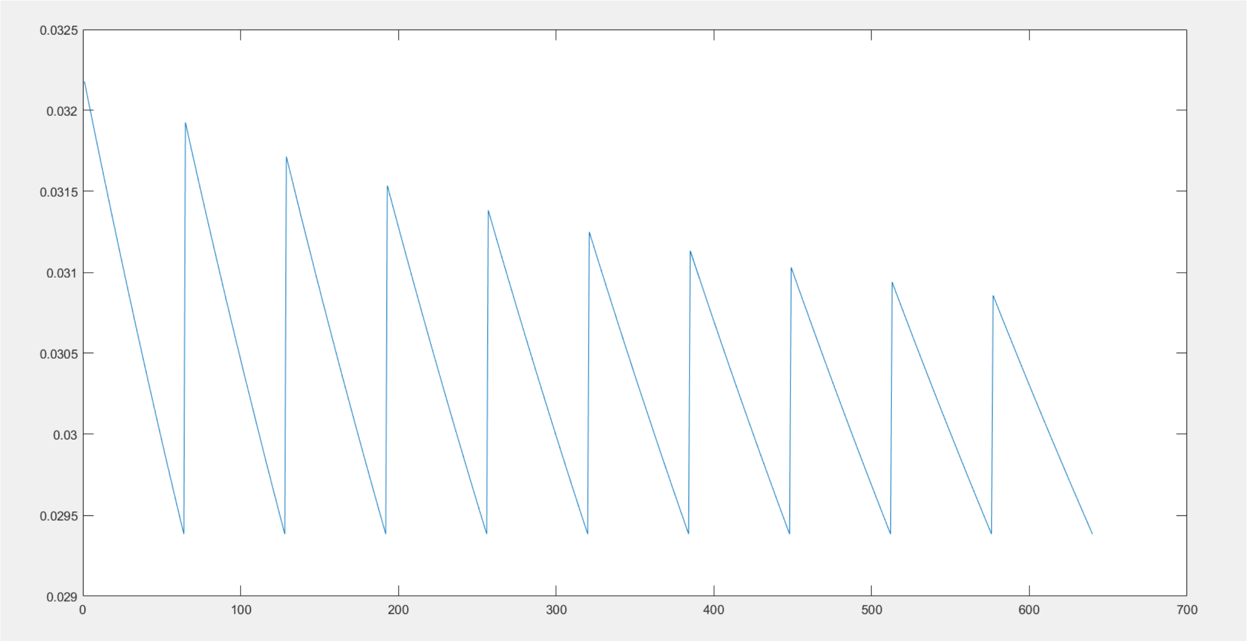

Figure 8

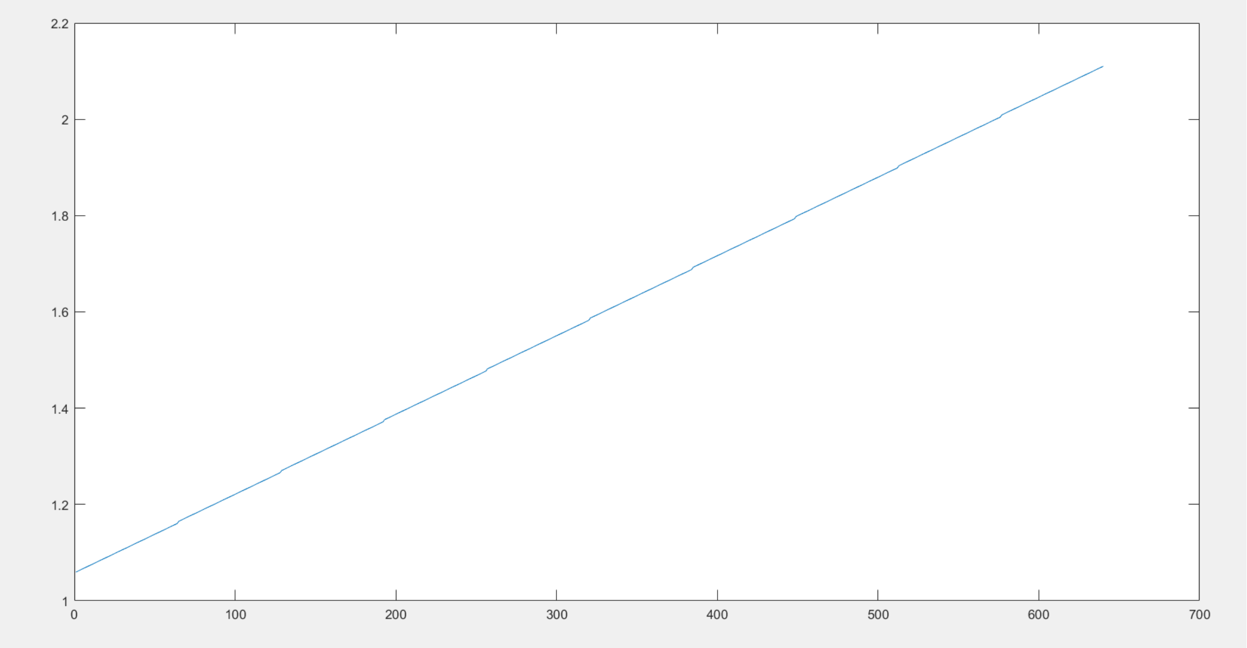

Figure 9

-Transmission time increase when we add new characters because number of bits transmitted increase so we will need more time.

<u>-Comparison between USB & UART with the same bit duration:</u>

Configuration: Both are with 8 bits data and the UART has no parity bit.

	UART	USB
Efficiency	0.8	0.9715
Overhead	0.2	0.0285
Transmission time	1.12 sec	1.0540 sec