Sami Farhat
STUDENT ID: 40097236

# LAB 3

**Develop a Server/Client application.**

**Performed on 7th February 2023**

Embedded Systems
COEN 421
WINTER 2023

Concordia University
7th March 2023
Section U-UJ

"I certify that this submission is my original work and meets.
THE FACULTY'S EXPECTATIONS OF ORIGINALITY"

SUBMITTED TO: CRISTIAN TIRIOLO

## 2. Objectives

The main objectives in this lab were to:

- Learn about the TCP Connections
- Understand how to program and establish TCP connection.
- Understand how to send and receive data from the connection.
- Get familiar with QNX

## 3. Theory

TCP or Transmission Control Protocol is a protocol in computer networks that allow for a reliable communication between two networking devices. The connection is established with a handshake, making sure there is a connect and a acknowledgement of the connection from both side. This is different from UDP, which is the second most used protocol, which is less reliable but faster.

Data is sent between the devices in segments once the connection has been made. Each segment receives a sequence number from TCP, enabling the receiving device to reassemble the data in the proper sequence. The rate of data transmission between devices is also regulated by TCP using flow control methods.

Given the foregoing , it is then possible to say that TCP is a robust and reliable protocol. Therefore, it makes sense that it is used when we are coding the TCP connection between our computer and the Robot.

In the scope of this laboratory, we are using the QNX environment to code. QNX is a real time operating system that provides a high level of reliability and performance for embedded systems. QNX also allows for an efficient use of multi-threading with POSIX which will be of great use in this lab.

The principle of coding a CLIENT and a SERVER means that everything should run on its own thread. To do that, we will be using the concepts of multithreading.

## 4. Task/Results/Discussion

**CPP FILE:**

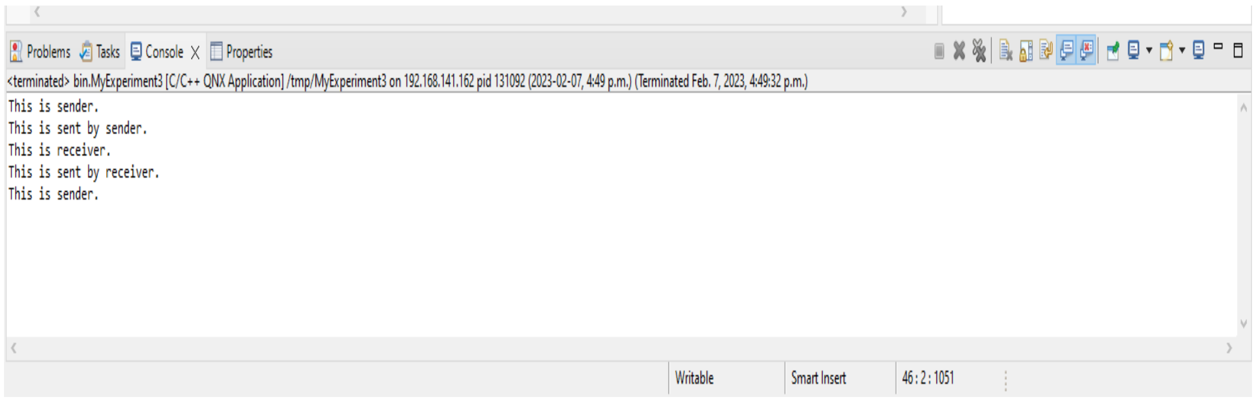
This experiment had us experiment with multiple built in functions in the socket library. I will explain them below. The CPP file had us design and define the methods needed in order to create and connect the TCP connections.

| | |
|---|---|
| socket () | Creates a TCP socket object. This returns a socket descriptor as an int , with -1 if the socket is closed |
| Close () | Takes a TCP socket object and closes it |
| Bind () | Takes a TCP socket object and associate it to an IP and to a port |
| Connect () | Establish the connection |
| Listen () | Takes a socket and it starts listening for incoming TCP connections. |
| Send () | Send the data required through the connection |
| Recv () | Receive the data required through the connection |

**MAIN FILE:**

In the main file, we made call to the proper functions needed. There is two function, one for the sender and one for the receiver. They both call on the Socket to create a socket object. The Sender gets mapped to an Ip address of the robot. We send and receive data. The receiver binds, listen, accept, and receive messages. Then when messages are received, it sends back an answer.

These two functions are void functions which then get mapped to threads which are created and then joined. This ensure that every function is running on its own thread.



```
<terminated> bin.MyExperiment3 [C/C++ QNX Application] /tmp/MyExperiment3 on 192.168.141.162 pid 131092 (2023-02-07, 4:49 p.m.) (Terminated Feb. 7, 2023, 4:49:32 p.m.)
This is sender.
This is sent by sender.
This is receiver.
This is sent by receiver.
This is sender.
```

Once the main file has been finish, we can execute the program. Given this output, we can see that the sender is created. Then the sender sends data. The receiver then gets called and the receiver receives and answers back. This small output allow us to prove that both client and server were properly functional and that the TCP connection is valid.

## 5. Questions:

**What is TCP socket programming?**

TCP socket programming use the transmission control protocol to deliver data reliably from one network to another network. The developer creates an application for communication. To be able to do it, we first create a socket. Then, we need to bind that socket to a specific IP and to a specific port in order to allow for communication . Then once the socket has been bound, we then make it listen to any incoming TCP connection to be ready to establish a connection a transfer data between the client and the server. When a connection is found, it is accepted. Then, send and receive functions get called to send and receive data from both ends. Once the whole data has been transferred and there is no need for the connection anymore, the connection and the socket get closed.

**What is the difference between socket and port?**

A port is always associated with an IP. It is mainly used to distinguish processes or protocols that are running on this IP. The port maps to a specific application. Common examples are port 80 which is always used for the HTTP and port 443 which is always used for HTT¨PS protocol. On the other hand, a socket uses both the port number and the IP addresses to establish a communication channel. That socket is then used to send and receive data.

## 6. Conclusion

To sum up, the experiment was successful since all the objectives were achieved. The practical exercise provided a comprehensive understanding of socket programming and the socket library. QNX was also properly used and the threads were a fresh reminder.