CS12020
Arduino Serial Networking Project Assignment Documentation

Muntazir Rashid – mrm19

17th December 2021

Introduction:

This assignment aims to program an **Arduino** with the **Software Serial Shield** to analyse the transmission of the messages sent and received through software serial ports connected to a virtual network of Hosts running on the connected **Nano**.
This project relies on **Aberystwyth Software Serial Protocol (ASSP)** to listen for messages on more than 1 Port. **ASSP** implements a reliable communication mechanism by ensuring a receiving Arduino is listening before a message is sent. It also allows the names of the destination and sender to be included in the message.
The overall aim of the assignment is to determine which Hosts are present on what ports and analyse the network's performance.
*"Also need to set the **Arduino Nano** to emulate a small network which is achieved by calling 'SerialShield.setEmuMode(6)' from setup function."*

Analysis:

Assignment consists of 4 Tasks divided into more minor problems.

Task 1: Does the port connect to that Host?

An array definition with the list of 26 Host Names was provided (i.e., ***names***).
This task aims to check which Host was connected to **Port 4**.
To tackle this problem, we were advised to create a *'Boolean*' function named ***"isHostOnPort"*** which send a '*ping'* message to the particular Host on the specified Port and returns '*true'* if the host is reachable and '*false'* if otherwise. To receive the ping response '*fetchASSPMessage'* is looped to try **50** times until it returns '*true'* and if it doesn't return '*true'* within 50 times the Host is not connected to the Port.
For testing "***isHostOnPort"*** a '*void*' function called ***task1()*** is written which goes through all the Hosts in the '***names array***' and prints whether the corresponding Host is connected on the Port 4 or not (as directed in the Task).

Task 2: Timing Port 2.

This task aims to return the time it takes for each Host connected on Port 2 to respond.
To achieve this an '*unsigned long'* function named "***getTimeToHost"*** is created which capitalizes on ***"isHostOnPort"*** function and the built in Arduino timer '***millis()'***.
To test this a '*void'* function named ***task2()*** is created which goes through all the Hosts in the '***names array'*** and prints the respond time if they are connected to **Port 2** and **NC** if not.

Task 3: Mapping the Network.

This task aims to return each Host connected to **Ports 1-4** and **NC** (Not Connected) for Hosts not connected to any port (i.e., **0**).

To get the Port number of every Host an '***int'*** function named "***getPortForHost"*** is created which returns the **Port Number (1-4)** or **0** if Host is not connected to any port at all.
A '*void'* function named "***Details ()" is*** created which gets the details (Port Number, Name and Time Taken) of all the Hosts and saves it in an '***Array of Struct (HostDetail)'*** named "**hostDetails_list"**.
For testing and to get the data in desired format i.e., **1-4 Port Numbers** and **NC** for Hosts not connected at all, a *'void'* function "***task3()"*** is written which loops through all the Hosts stored in "**hostDetails_list"** and prints the result in required format i.e., to what Port each Host is connected to.
***I used struct "HostDetail" to collect all the Host details to increase the efficiency of the program.**

Task 4: Network response time analysis.

Aim of this task is to analyse the response time of hosts on different Ports.
Same ***'Details()'*** function created during Task 3 is used and the details obtained are looped to calculate the number of Hosts on each Port, fastest and slowest Host on each Port and to print the details in required format.
*I utilised the struct created in ***task3()*** for efficiency.
*I couldn't figure out How to print the Name of fastest and Slowest Hosts on each Port.

Problems Encountered:

Task 1 and Task 2 were straight forward and went smooth.
Getting Task 3 results was not that hard but **formatting** the Task 3 results as described in the assignment was complicated and required a lot of time. First, I came up with an idea to use loops to format it in the desired way, but this was extremely inefficient as it had to recollect the data every single time during the loop and took a lot of time to print the result. After that I created a struct named "***HostDetail"*** which collected all the details required to get the result and used the loops to access the details from "***HostDetail"*** instead of recalculating it again and again.
Same struct "***HostDetail***" was also used in Task 4 to collect the time it took for each host on a particular Port to respond.
*I couldn't figure out How to get the Name of fastest and Slowest Hosts in Task 4.

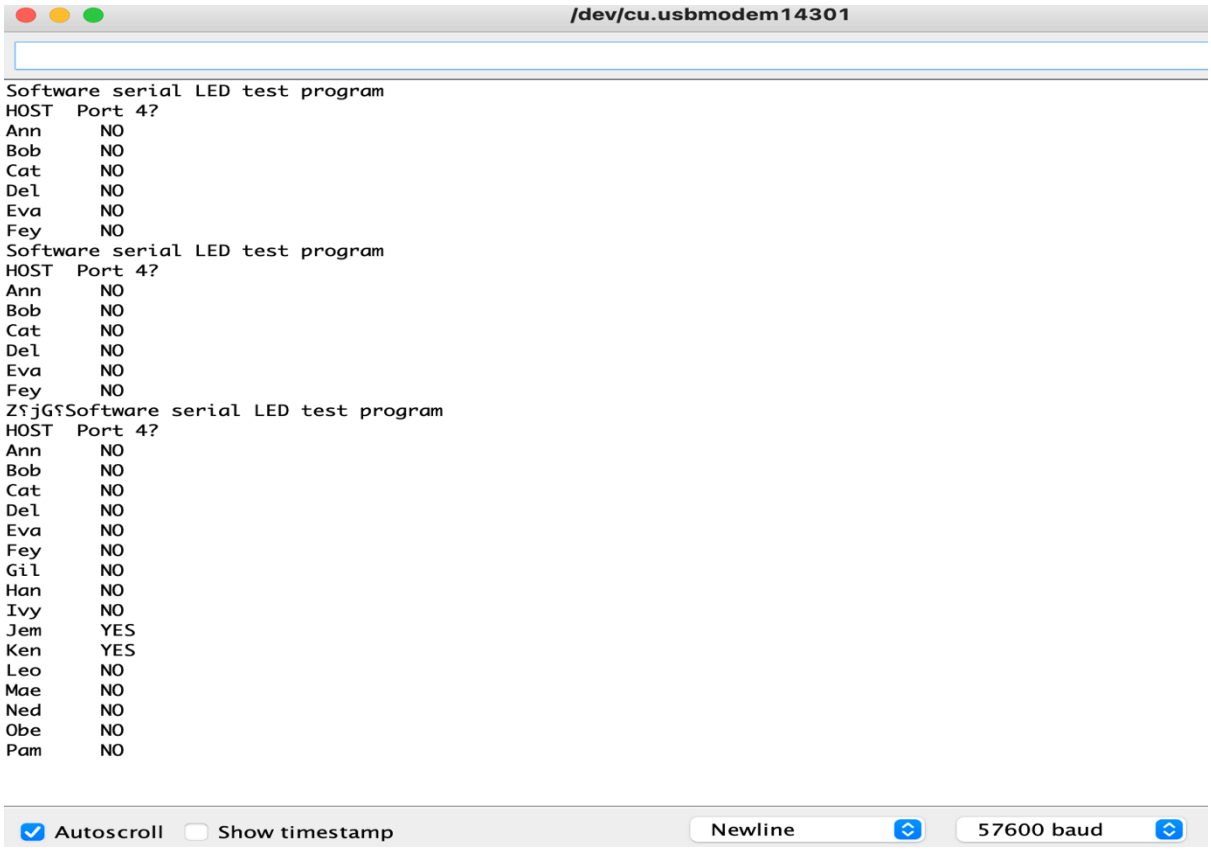
Your assessment of the mark you would award yourself for the work (based on the assessment criteria detailed):

1st: 70-79%.
I believe that the written components are professionally presented in both layout on the page and logical structure with a high grammatical standard. Each Task is implemented using appropriate technology and will at least completely fulfil the functional requirements.

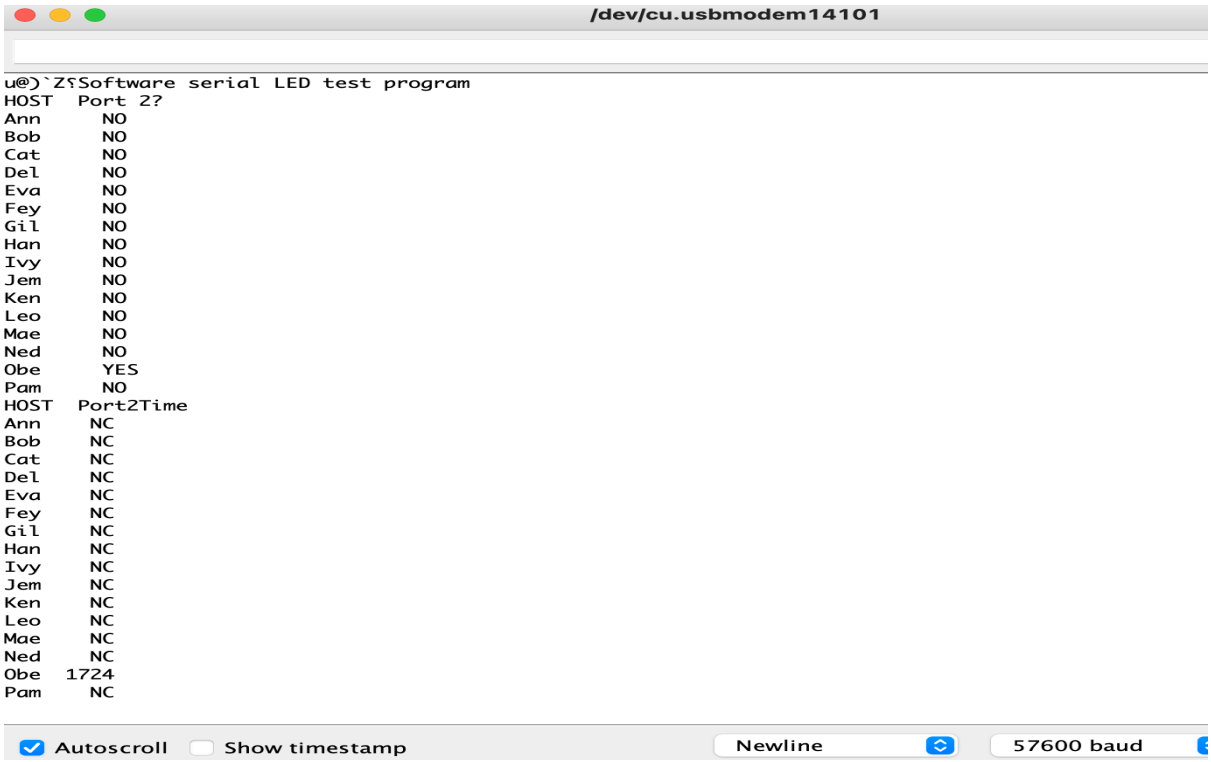
Output

Test Outputs:

Task 1:



Task 2:

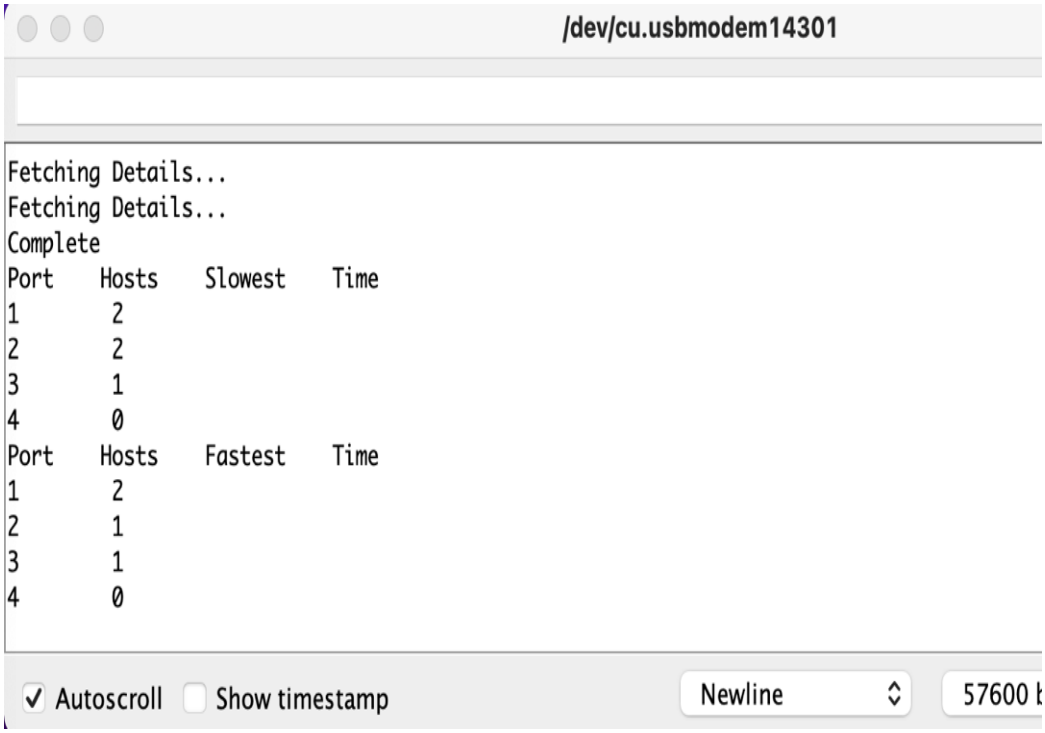


Task 3:

```
Port    Hosts
NC      Ann
4        Bob
NC      Cat
1        Del
2        Eva
3        Fey
2        Gil
NC      Han
3        Ivy
4        Jem
```

Results before Formatting

Task 4:

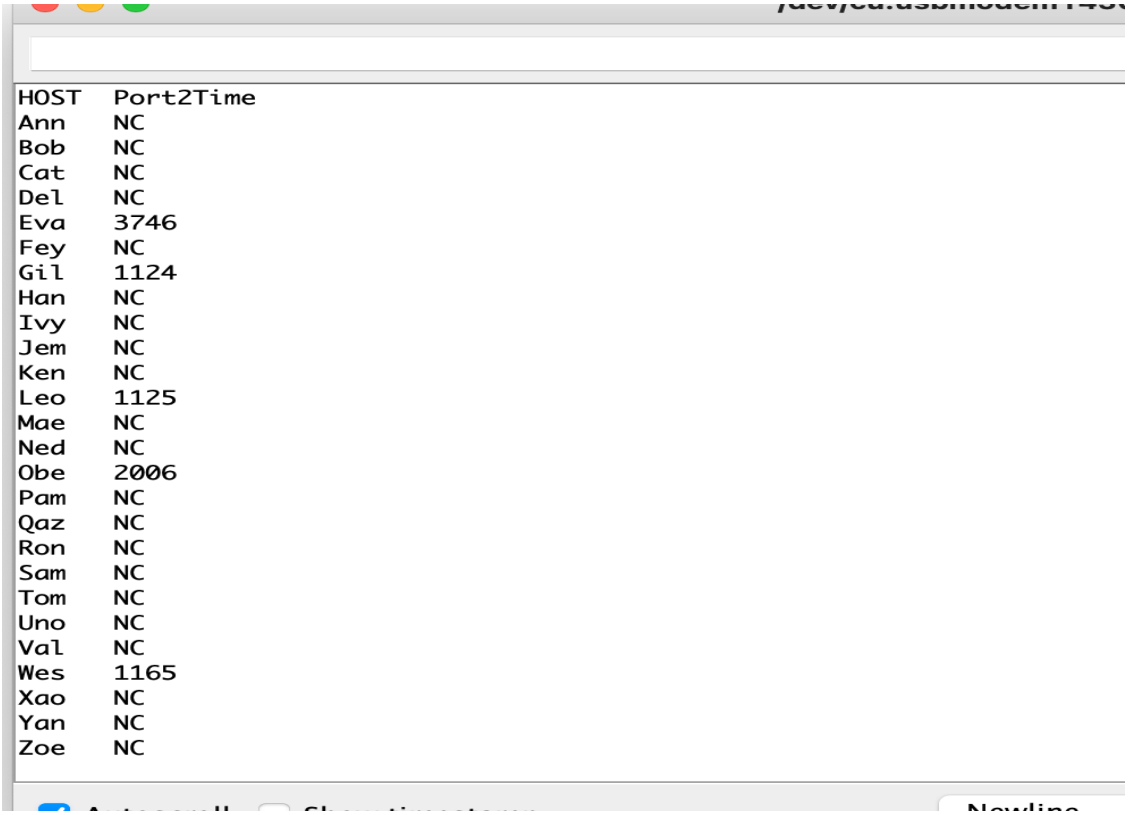


//Actual Formatted Output is Below

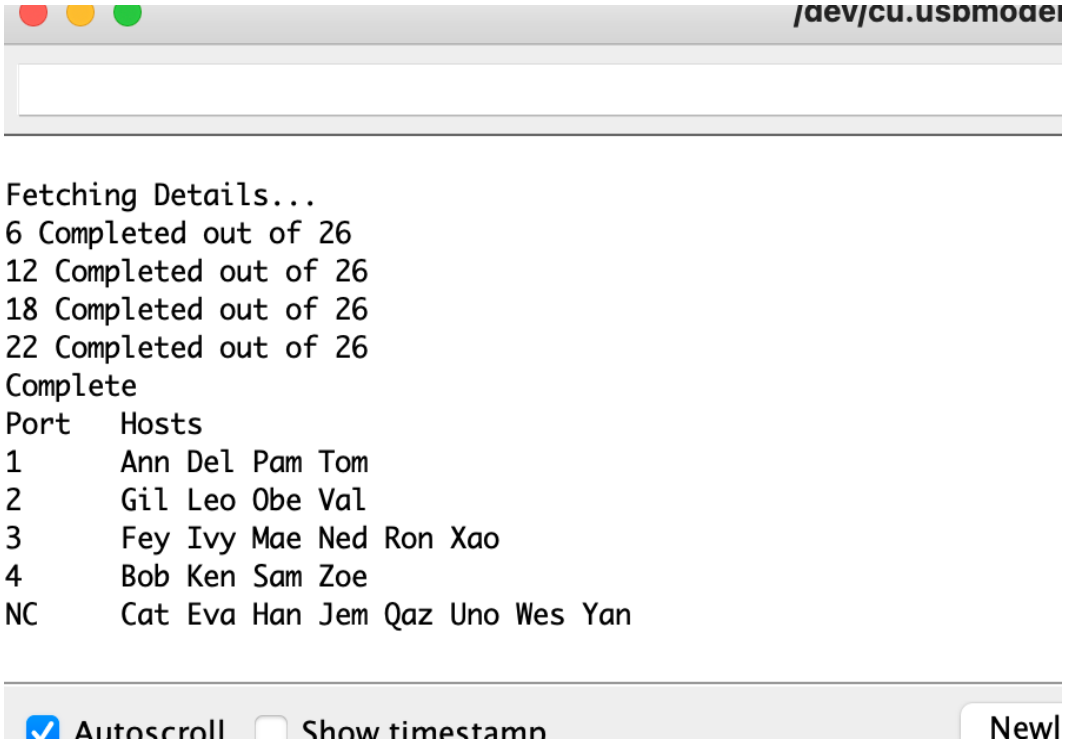
Desired Output:

Task 1:

Task 2:

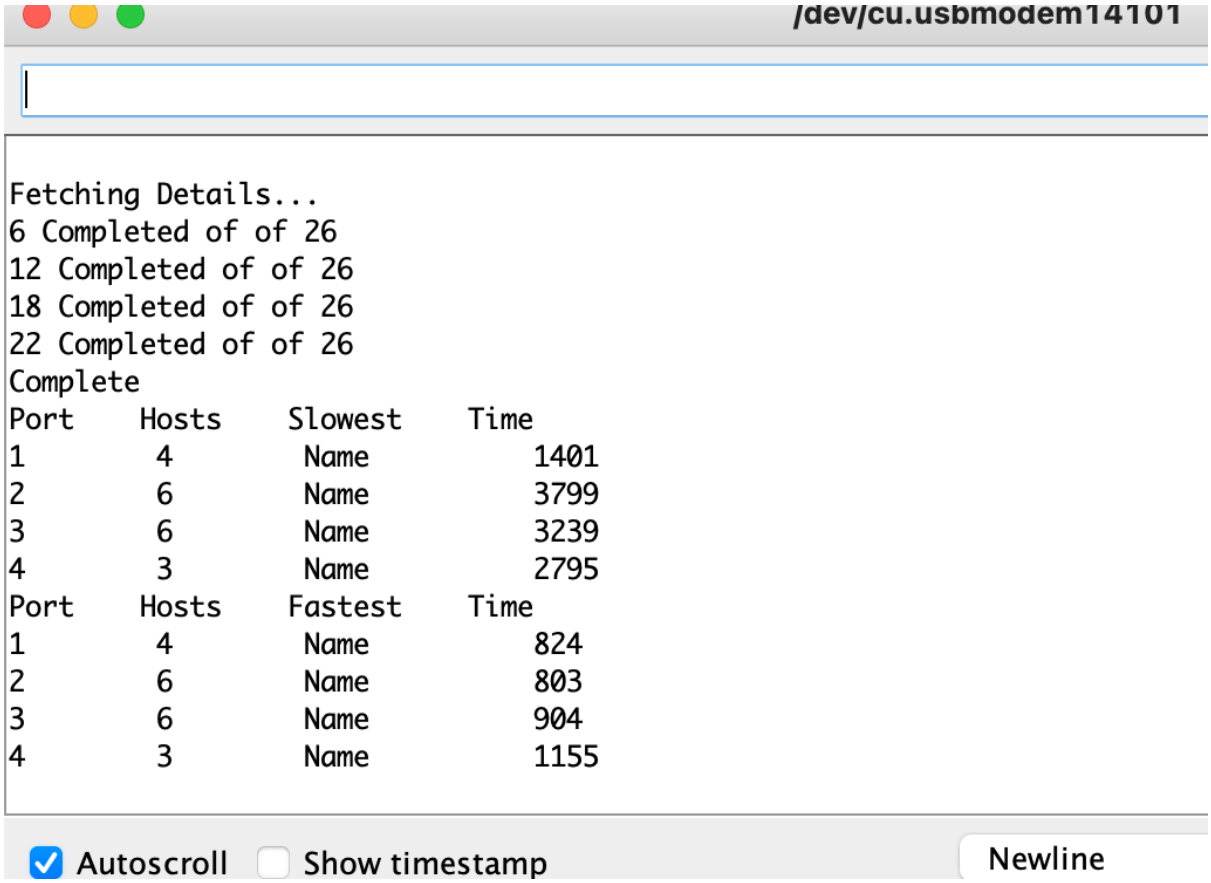


Task 3:



// printing progress message //

Task 4:



/*Wasn't Able to figure out how to get Names of Fastest and Slowest Hosts*/