International Journal of Multimedia and Ubiquitous Engineering
Vol.12, No.10 (2017), pp.27-36
http://dx.doi.org/10.14257/ijmue.2017.12.10.03

Data Communication of OBS-emplaced Navigation System Based on Multi-Serial Networking

Guizhou Zheng[1], Pengfei Zheng[2] and Xiaofeng Zhang [3*]

[1]*Faculty of Information Engineering, China University of Geoscience*
[2]*Computer Engineering, University of Waterloo*
[3]*Mechanical Engineering and Electronic Information, China University of Geosciences*
[1]*zhenggz@cug.edu.cn*, [2]*p6zheng@uwaterloo.ca*, [3]*3087347745@qq.com*

Abstract

This study built an ocean bottom seismometer (OBS)-emplaced navigation system and designed a serial port networking device that contains several RS232 serial ports and an Ethernet interface. The serial ports networking device has a built-in embedded multiple serial ports gateway based on the TCP/IP protocol. The gateway comprises TCP/IP protocol conversion and multiple serial port data processing modules. The differential GPS receiver, OBS, depth sounder, compass, and other equipment are all connected to the network via RS232 serial ports. Subsequently, all serial data streams are transformed into an Ethernet data stream through protocol conversion. A TCP/IP protocol includes several message types, such as network port data receipt, serial port baud rate setting, serial port baud rate response, serial port switch set, serial port switch response, serial port status query, and serial port status query response, among others. Accordingly, we employ data analysis to obtain specific information on navigation, as well as the latitude, longitude, direction, and depth from the original data stream. Sea trials indicate that the proposed system can provide real-time, accurate, and reliable data, and can address the needs of the OBS-emplaced navigation.

Keywords: *computer communication; wireless communication; TCP/IP; network protocol; navigation*

1. Introduction

An ocean bottom seismometer (OBS) is a marine survey equipment placed on the seabed and is used for receiving artificial or natural seismic waves [1]. We can derive seabed geological structures from seismic waves through tomography. OBS can be used for the manual profiling of oceans and observing natural earthquakes. This equipment is extensively used in conducting surveys on submarine-deep structures, offshore oil, and gas hydrates [2]. An OBS position coordinate is one of the key parameters in 3D seismic structural studies [3]. We can obtain a variety of navigation and positioning information to accurately locate an OBS using an OBS-emplaced navigation system, such as differential GPS (DGPS), OBS, compass, echo sounding, and other information. In an OBS navigation system, data communication links the system and external navigation devices. Thus, a data communication scheme should be designed to enable effective organization and management of different devices, accurate data transfer and analysis, and achieve a real-time display of various types of navigation information. Current exploration ships are equipped with such devices as DGPS, OBS, compass, and echo sounding that use the RS232 serial port. Ship navigation systems use multiport serial

Received (October 26, 2016), Review Result (July 5, 2017), Accepted (September 29, 2017)
* Corresponding Author

ISSN: 1975-0080 IJMUE

cards to communicate across all devices by increasing the number of computer ports. However, the RS232 serial port presents numerous problems. First, the RS232 communication protocol is simple. Second, the RS232 transmission distance is considerably short. Third, the serial port is susceptible to electromagnetic interference. These problems present a substantial challenge in the use of the system, particularly in terms of maintenance and troubleshooting [4]. Accordingly, network-based data communication can solve these problems. The development and popularity of Internet technology have resulted in the increasing and extensive popularity of "device networking." The process of sending data from multiple serial ports to the network and the performance of remote transmission and control equipment has already been applied in many industries. This study proposes a high-speed management and communication technical scheme based on multi-serial networking to maximize the existing resources and advanced technology. The proposed scheme accomplishes serial data network communication through the networking of the serial ports of devices.

2. OBS-emplaced Navigation Communication System Designs

2.1. Framework of the OBS-emplaced Navigation System

An OBS-emplaced navigation system based on an electronic chart is an integrated navigation and positioning information system that is designed to assist OBS for precise emplacement [5]. This navigation system includes device management, survey network and station management, OBS state management, and OBS positioning results management, among others. The device management module is the core of the entire system and is mainly responsible for data communications with DGPS, OBS, compass, echo sounding, and other equipment; management and control of devices; and analysis of real-time data from multiple pieces of equipment. The navigation system framework is shown in Figure 1. Through DGPS, this system can obtain a ship's location information and basic navigation operations, such as time, longitude, latitude, actual heading, speed, and other similar information. This navigation system can also obtain the longitude and latitude to identify the OBS position from OBS. Moreover, this system can obtain the depth values through echo sounding and determine a ship's heading values through the compass.

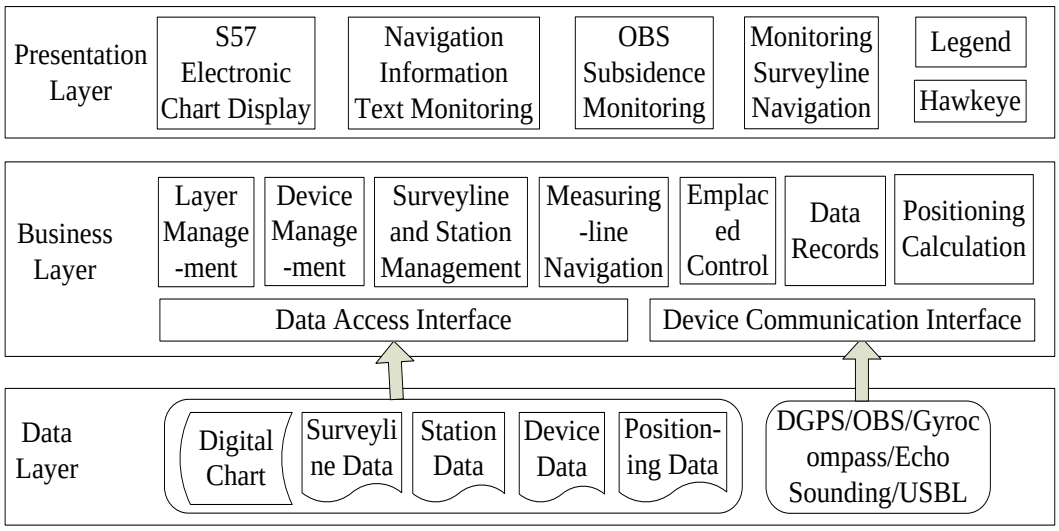

Figure 1. Framework of the OBS-emplaced Navigation System

2.2. Design of the Port Networking Device and Gateway

To enable the compatibility of the protocols of multiple serial ports and the Ethernet, we designed a serial port networking device that comprises several RS232 serial ports and an Ethernet interface. First, the DGPS receiver, OBS, depth sounder, compass, and other equipment are all connected to the networking device via the RS232 serial ports. Subsequently, all the serial data flows are transformed into the Ethernet data flow through protocol conversion. Thereafter, all host computers in the same LAN could establish a connection with the network equipment through the TCP/IP protocol, thereby enabling the data to be transferred between them. Thus, we could set the parameters, query the statuses of the remote serial ports device, and send and receive data. Users simply need to determine how to use the serial ports and do not need to perceive how the data are transformed into the Ethernet protocol [6]. The design of the system is shown in Figure 2.

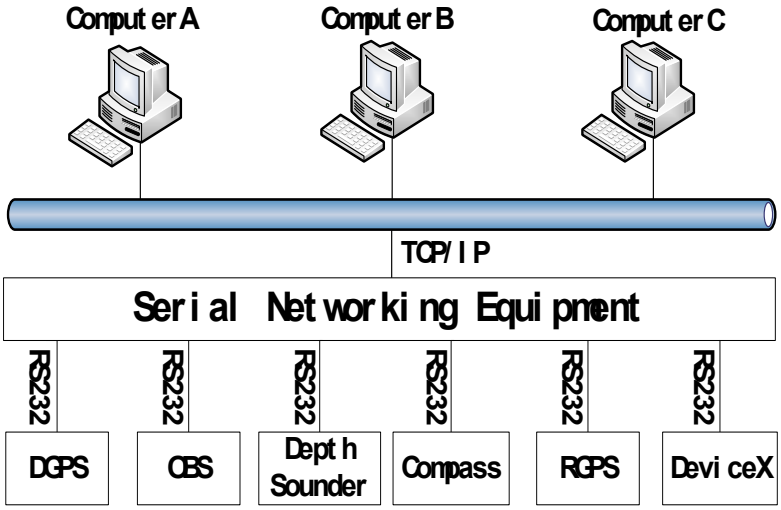

Figure 2. Design of the Serial Device Networking System

The TCP/IP protocol comprises the application, transport (UDP layer), network (IP layer), and data link layers. To achieve transparent transmission, an increase in the application of the process protocol layer (serial port layer) is implemented. The serial port layer comprises a serial port link and serial port network layers. The gateway is constructed on the serial port layer and is responsible for analyzing the RS232 data packet. Upon analysis, the result is transmitted to the TCP/IP network application layer.

The serial ports gateway comprises a TCP/IP protocol conversion and multiple serial port data processing modules [7]. The system structure is shown in Figure 3. The TCP/IP protocol conversion module is a miniature Ethernet access module that comprises the microcontroller (AT91 ARM9200), network interface chip (LXT971ALE), network isolation transformer (HR601680), SDRAM (HY57V561620T), and NOR–FLASH (AM29LV320DB), among others. The primary function of the TCP/IP protocol conversion module is to encapsulate the serial ports frame in the Ethernet transmit buffer in the UDP packet and send it to the IP layers. Simultaneously, the module receives the Ethernet data frames, unpacks the data for each layer, and separates the data from the application layer. The multiple serial port transmission modules complete the data analysis and achieve the transparent conversion between the RS232 serial port flow and Ethernet port flow. The client data of the OBS-emplaced navigation and positioning system are sent to the microcontroller through the Ethernet interface and network interface chip. ARP, ICMP, IP, and other protocols can be decoded and packaged using the microcontroller, where the TCP/IP protocol stack is embedded. The serial data processing module, which primarily comprises a microcontroller, serial port expansion

chip (GM8125), and MAX232, is a transmitting/receiving control module of a multiple serial (RS232) port data flow. The main function of this module is to receive the equipment data of multiple serial ports (e.g., GPS, OBS, and compass) and send all the data packages to the Ethernet buffer for packaged transmission.

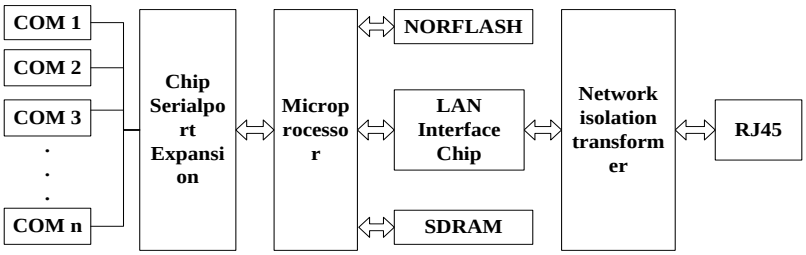

Figure 3. Multi-serial Port Gateway Structure

2.3. OBS Communication System Framework

The entire data communication architecture uses the C/S network communication model [8]. The embedded multi-serial port gateway on the server, which relies on the embedded system platform with an ARM processor as the core and Linux as the operating system, provides an independent open, close, and monitor thread for the serial device, thereby enabling each device to complete its independent operations [9]. A multi-serial port gateway that runs on a server is constantly in listening mode and achieves conversion between the multi-serial port and Ethernet data through protocol conversion and data analysis.

The OBS-emplaced navigation and positioning software system on the client includes a device management module. This module, which is designed based on the TCP communication principle and multi-threading technology, as well as realized based on the socket network communication program, establishes a connection between the client and server through the specified IP and port to realize data communication. Multiple data buffers must be created in the device management module of the client. The buffers mainly include a primary buffer, which is used to store the Ethernet data flow, and device buffers that correspond to each port. Two processes are performed after analyzing the received Ethernet data. (1) The source data of each serial port are displayed in the corresponding text box to verify and decide whether the normal data reception should occur through the test window. (2) The source data of each serial port are placed into the corresponding device buffer to be further analyzed in another thread. The final analyzed result is encapsulated in a structure and passed thereafter to each of the function modules of the OBS-emplaced navigation and positioning system. The workflow is shown in Figure 4.

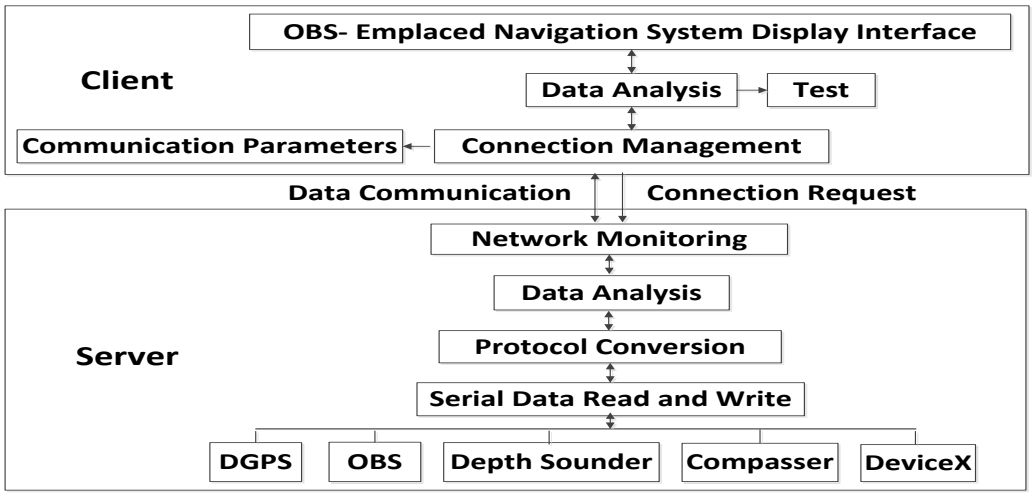

Figure 4. Communication System Framework

3. Communication Protocol Design and Data Analysis

3.1 Communication Protocol Design

The host computer communicates with the serial port network device based on the TCP/IP protocol. Although the host computer can obtain the data flow through the socket network programming because the data are not specifically semantic data, the issue cannot be properly resolved. To analyze the data flow received from the Ethernet port, a communication protocol that includes the structure and data types of the receiving and sending of information should be designed. The data of each serial port device could be isolated through the protocol. The host computer could send control information to a serial port device, such as controlling a serial port switch and changing the baud rate, and generate the appropriate response message as well. The protocol message is indicated by "string" and "int," which are two basic data types presented in ASCII. The message structure includes a message header and message body. The message header provides identification information of the message content, whereas the message body is the specific content of the message. The protocol includes the following message types.

(1) Network port data receive format

The data flow obtained by the Ethernet port could be separated by the host computer through the data receive format (see Table 1). The host computer can obtain the data of each serial port. In the data receive format, the message body comprises a receive flag, serial port byte, and serial port data. The length of the message body data can be determined via the length from serial port 1 (COM1) to serial port 6 (COM6). Serial port byte and serial port data are sequentially stored from COM1 to COM6. If data are present in a few COM, then the COM data before COM are directly followed by the COM data after COM. The received data of each serial port does not exceed 8 bytes each time.

Table 1. Network Port Receive Data Format

Field name	Number	Type	Description
Header_Flag	4	string	"DATA" represents the head of message package
Receisend_Flag	1	string	Flag of receive and send data
COM1_Length	1	int	Length of serial port 1 data, 0 represents no data
COM2_Length	1	int	Length of serial port 2 data, 0 represents no data
COM3_Length	1	int	Length of serial port 3 data, 0 represents no data

COM4_Length	1	int	Length of serial port 4 data, 0 represents no data
COM5_Length	1	int	Length of serial port 5 data, 0 represents no data
COM6_Length	1	int	Length of serial port 6 data, 0 represents no data
Data	0–48	string	Stored order of each serial port data block

Consider the following hexadecimal notation data as example:
44 41 54 41 01 08 00 00 04 00 00 31 32 33 34 35 36 37 38 30 30 30 30
44, 41, 54, and 41 are the message header flags (String DATA).
01 is the flag of the receive/send data, whereas 01/00 is the receive/send data.
08 is the number of bytes of serial port 1 (i.e., 8 bytes).
00 is the number of bytes of serial port 2 (i.e., 0 bytes).
00 is the number of bytes of serial port 3 (i.e., 0 bytes).
04 is the number of bytes of serial port 4 (i.e., 4 bytes).
00 is the number of bytes of serial port 5 (i.e., 0 bytes).
00 is the number of bytes of serial port 6 (i.e., 0 bytes).
31, 32, 33, 34, 35, 36, 37, and 38 are equal to 1, 2, 3, 4, 5, 6, 7, 8 that are received from serial port 1.
30, 30, 30, and 30 are equal to 0,0,0,0 that are received from serial port 4.

(2) Serial port baud rate setting format

BAUD	Serial port number	Configured number of baud rate

BAUD is the message header. The serial port number has values that range from 1 to 9. The configured number of the baud rate has values that range from 0 to 9. For example, "BAUD 4 0" indicates that the baud rate of serial port 4 is set to 0 (The configured number of the baud rate can be set based on the specifications of the devices. For example, 0 represents 4,800, 1 represents 9,600, 2 represents 38,400, 3 represents 57,600, and 4 represents 115,200.).

(3) Serial port baud rate response format

RSPB	Serial port number	Configured result of baud rate

RSPB is the message header. The configured result of the baud rate value is 0 or 1, where 0 and 1 represent the failure and success, respectively, of the configurations.

(4) Serial port switch set format

UART	Serial port number	Switch state

UART is the message header. The state of the switch value is 0 or 1, where 0 and 1 represent close and open, respectively.

(5) Serial port switch response format

RSPU	Serial port number	Configured result of switch

RSPU is the message header. The configured result of the switch value is 0 or 1, where 0 and 1 represent the failure and success, respectively, of the configurations.

(6) Serial port status query format

QURY	Serial port number

QURY is the message header. The serial port number has values that range from 0 to 9.

(7) Serial port status query response format

RSPQ	Serial port number	Configured result of baud rate	Switch state

RSPQ is the message header. The serial port number has values that range from 0 to 9. The configured number of the baud rate has values that range from 0 to 4. The state of the switch value is 0 or 1, which represent close and open, respectively.

The server system rule states that the data component of the datagram that is sent each time should not exceed 512 bytes. To ensure real-time data transmission, if the time is

required to contract, then the server will also send the packets whether the server reaches 512 bytes. Thus, the system particularly regulates the data capacity and time to ensure that the requirements of the specific application for real-time operation can transmit substantial data.

3.2 Data Analysis

We can obtain specific information on the navigation and the latitude, longitude, direction, and depth from the original data stream from the Ethernet port via a two-step data analysis. A worker thread and data buffer can be created for each serial port by setting the device type of each serial port, which is the corresponding resolved driver of the respective devices. The original Ethernet data flow will be converted to the specific navigation and positioning data that can be used for calculation in a manner that is lossless, real-time, and considerably efficient via a two-step analysis.

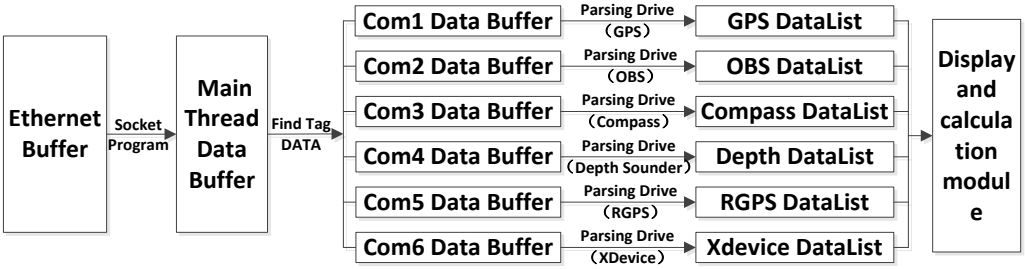

Figure 5. Data Analysis Process

First, the data flow from the Ethernet buffer can be obtained through socket programming in a worker thread. The data flow is sent to the main thread through message notification or function callback. Meanwhile, the main thread seeks the DATA head section in its data buffer (e.g., the head section of the network interface data format) and places the int value thereafter, which the last 6 bytes of the DATA is converted into, into an integer array. The array is used to store the number of bytes of each serial port data per message. Subsequently, traversing the array elements is performed to successively divide the following data in accordance with the element values. These data are stored into the corresponding buffer of the device to refresh the main buffer.

Second, the first step in the analysis indicates that the data are continuously added in the corresponding device buffer of each serial port. Meanwhile, a worker thread continuously runs in the device object. A complete data section is separated out in the thread mainly based on the delimiter of the device data. Thereafter, the data are refreshed in the device buffer. Data, such as time, longitude, latitude, orientation, and depth, are extracted by using the analytic function. Eventually, the data are packaged in the structure and sent to other modules for calculation, updating, and display.

4. System Simulation and Testing

The device communication client (see Figure 6) of the proposed OBS-emplaced navigation and positioning system mainly include a serial port control, serial port configuration, connection management, and source data testing. The client establishes a connection with the server by specifying the IP and port numbers. An individual can manage and configure each serial port through this client and test the sending and receiving cases of each serial port data. In addition, the client supports the functions of opening or closing any serial port and the real-time display information of the connection status and other functions.

The system can set the device name, device type (i.e., corresponding analytic functions of the device), baud rate, and other parameters for each serial port through the

configuration window of the serial port (see Figure 7). Furthermore, the parameters of the specific type of device (e.g., header type of the GPS that must be addressed, single/dual frequency mode of the depth gauge, angle correction value of the compass, and buffer size of the storing track points, among others) can be set. In addition, the configuration window supports deletion for each device connected through the serial port.

The system uses the serial port network equipment to access the multi-channel serial data. Because the serial port networking equipment is independent and has a user-defined data transfer protocol so that the serial port Parameter adjustment is more transparent and the data transmission is more efficient and secure. It's better than traditional serial port expansion card. To verify the function and performance of the system, we performed tests on the entire system (see Figure 8) by using the NmeaWiz and c# dotTrace test tools. As reference, we set a serial port that lacks data transmission. The NmeaWiz test tools simulate 5 different format types of device data via the other five serial ports to which uninterrupted data are sent based on the period of 100, 200, 300, 500, and 1000 ms within 24 h. The results of the test are as follows: 1) the system runs stably, 2) it meets the requirement of real-time operation, and 3) no errors occurred. The source data test window mainly displays the source data information of each serial port. For example, serial ports 1 and 3 receive the GPS and OBS data, respectively. Serial port 2 does not send data. Serial port 4 receives the depth gauge data (including DA and DB). Serial port 5 receives the compass instrument data. Serial port 6 receives the RGPS (trailer) data. Data that are analyzed through the two-step analysis are displayed on the right side window of the test window in the text. Under the different network load, the key parameters of network communication, such as receiving frequency, decoding time, unpacking time and so on, are tested. The test results show that the performance of the equipment network connectivity is excellent (see Table 2).

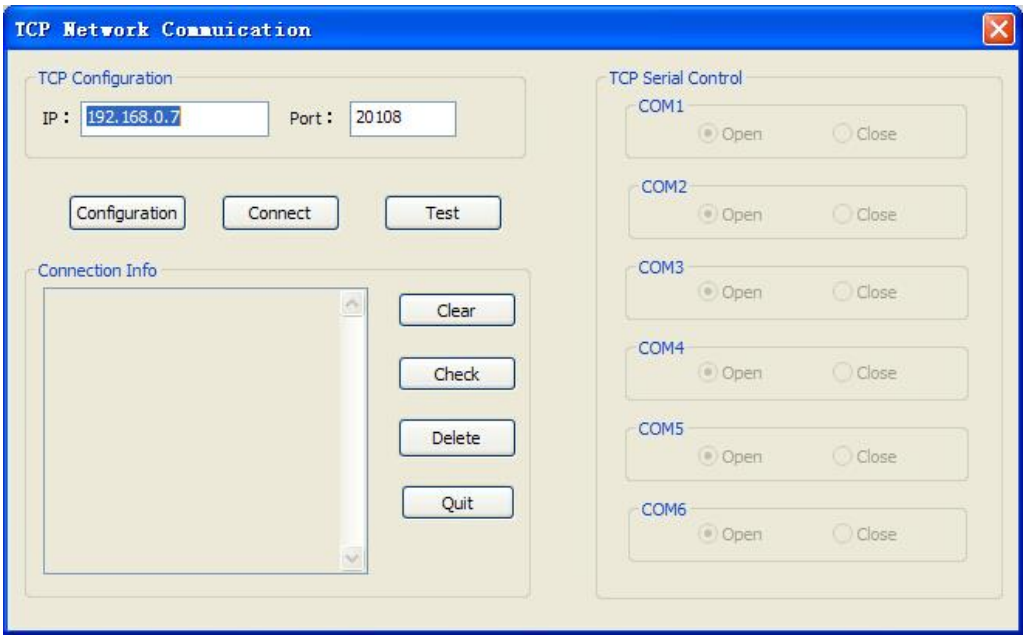

Figure 6. TCP Network Communication Manager

Figure 7. Serial Port Configuration

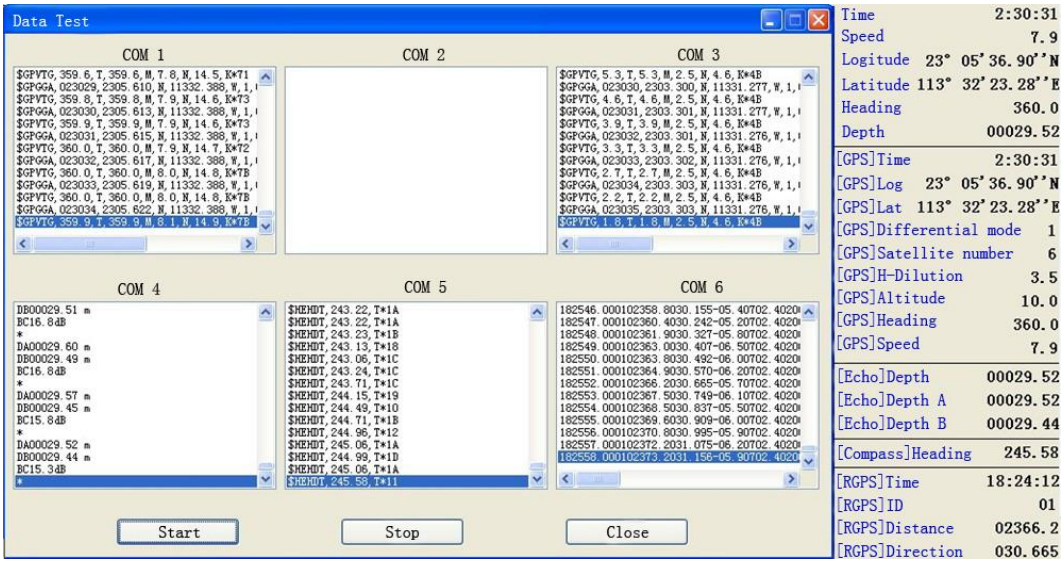

Figure 8. Data Communication Test

Table 2. Equipment Network Connectivity Performance Test

Equipment number	Receiving time (ms)	Decoding time (ms)	Unpacking time (ms)	Average receiving time (ms)	Receiving frequency (ms)	Total time (ms)
1	15	1	842	0.65	23	22807
2	94	2	1498	2.35	40	40438
3	66	2	1635	2.36	28	27882
4	67	1	1360	3.53	19	18776
5	45	1	1855	3.21	14	14094

5. Conclusion

This study proposed the use of computer networks and software and hardware technology in the context of OBS detection combined with the current status of the use of an exploration ship's equipment to provide a set of network solutions for a multi-port network for real-time data communication of multi-channel navigation equipment (e.g., DGPS, OBS, compass, and echo sounding). An OBS data communication subsystem was created based on the TCP/IP high-speed network communication. Sea trials of the system demonstrated that the proposed solution presented in this study can provide accurate, stable, and reliable real-time data for various functional modules of the OBS-emplaced navigation and positioning system. This system runs stably and can meet the requirement of real-time operation, thereby considerably facilitating the remote control and unified management of a variety of serial port devices.

References

[1] J. F. Hu and J. L. Jin, "Design and implementation of OBS emplaced navigation and positioning system", Science of Surveying and Mapping, vol. 39, no. 4, **(2014)**, pp. 68-73.
[2] J. J. Yan, J. F. Hu and J. G. Shang, "Optimized approximate position computing for OBS multi-point pseudo range measurement", Science of Surveying and Mapping, vol. 38, no. 16, **(2013)**, pp. 96-98.
[3] L. Zhang, "Correction of OBS position and recent advances of 3D seismic exploration in the Central Sub-Basin of South ChinaSea", Earth Science-Journal of China University Geosciences, vol. 38, no. 1, **(2013)**, pp. 33-34.
[4] W. T. Zhan, L. Wang and J. G. Sun, "Research on transforming between multi-serial ports and Ethernet", Aeronautical Computing Technique, vol. 41, no. 4, **(2011)**, pp. 102-108.
[5] W. Huang, S. L. Gu and W. Lu, "Design and implementation for ECDIS development library compliant with international standards", Navigation of China, vol. 32, no. 4, **(2009)**, pp. 17-21.
[6] P. Zhou, C. Huang and N. Jiang, "A research of communication resolutions combining serial port with network", Computer & Network, no. 15, **(2011)**, pp. 68-70.
[7] W. Fan and H. Z. Xu, "Design of embedded serial port network gateway based on ARM and TCP/IP protocol", Computer Engineering and Design, vol. 29, no. 1, **(2008)**, pp. 80-82.
[8] L. J. Li and D. F. Ye, "Implementation of network communications based on the C/ S construction in Winsock ActiveX", Computer engineering & science, vol. 31, no. 2, **(2009)**, pp. 20-23.
[9] Y. J. Wang and J. G. Liu, "Technology research on linux applied in embedded system", Application Research of Computers, vol. 22, no. 5, **(2005)**, pp. 102-104.

Author

GuiZhou Zheng is a professor of the Faculty of Information Engineering of China University of Geosciences. His main research areas include WEBGIS, 3S technology, geographic calculation, 3D visualization, and GIS application project. He has been in charge of several research projects, such as OBS navigation and positioning system, real-time OBS 3D visualization software development, double-boat real-time status observation system, 3D mine visualization, aerial survey database management system, and geological disaster forecast system. He has published over 40 journal papers and 3 books.

