Design and Implemention of an Emulation Node for Space Network Protocol Testing

Sichen Zhao, Yuan Fang(✉), Wenfeng Li(✉), and Kanglian Zhao(✉)

School of Electronic Science and Engineering, Nanjing University, Nanjing 210023, China
zsc199405606@outlook.com, {yfang,zhaokanglian}@nju.edu.cn, leewf_cn@hotmail.com

Abstract. This paper proposed a scheme to design and implement an embedded emulation node, which provided an approach to emulate a precision network environment for the measurement of network protocols in the space information network. The design scheme employed the Beaglebone black (BBB) board which owns the approximate hardware performance as a satellite has, ported the real-time operating system RTEMS to BBB board, and then loaded the ION-DTN on it. We established the RTEMS development environment on BBB board, and then designed and implemented the network adapter driver associated with BBB board for RTEMS. The emulation results show that the designed emulation node is stable and reliable. With the DTN protocol stack working properly on emulation node, the function and performance of the designed emulation node can meet the requirements of space network protocol tests.

Keywords: DTN · RTEMS · Satellite emulation node

1 Introduction

A Space Information Network (SIN) is the network system based on space platforms, such as geostationary (GEO) satellites, medium earth orbit (MEO) satellites, low earth orbit (LEO) satellites, stratospheric balloons, etc., which can obtain, transmit and process space information in real time. Its goal is for the increasingly demand of information sharing among the connected heterogeneous networks, and the efficient data delivery in SIN.

Due to the harsh transmission conditions in space communication environment, such as the long propagation delay, high loss of data, frequent link disruptions, dynamic changes of the network infrastructure, the condition or environment of space networking is quite different to that of terrestrial network. It is

This work is supported by the National Natural Science Foundation of China (No. 61401194), the Fundamental Research Funds for the Central Universities (021014380064) and the Priority Academic Program Development of Jiangsu Higher Education Institutions.

X. Gu et al. (Eds.): MLICOM 2017, Part II, LNICST 227, pp. 658–667, 2018.
https://doi.org/10.1007/978-3-319-73447-7_70

necessary to carry out the research on basic theory and key technologies for SIN before they are directly employed in space networking. While some of the existing network simulators can provide flexible, scalable network Emulations, e.g. OpenNet and NS-2, they are just discrete event-driven state simulators which only create an open Emulation environment, and can not generate real data stream inside the network as the actual network does. This greatly limits the emulation to satisfy many experimental requirements of the network applications.

For research on SIN, we can construct a measurement platform to create the scenarios as the environment of space networking. Generally, we employ a PC as a satellite node in the emulation platform. However, the performance of PC hardware is much better than that of satellite hardware which would cause deviations of the emulation results in some measurements.

To eliminate such effects, this paper studies how to design and implement an embedded emulation node to provide an approach to emulate a precision network environment for the measurement of network protocols in SIN.

2 General Scheme Design

As mentioned before, it is important to develop an emulation node with similar properties owned by the satellite, which can improve the authenticity of the space networking environment for SIN protocol measurements.

In this paper, we develop the emulation node in SIN based on Beaglebone Black (BBB) embedded core board [1], which can provide the approximate hardware performance as a satellite does.

BBB board is a low cost embedded core board developed by TI company. The board hardware specification includes a TI AM335x [2] series ARM processor running at 1 GHz, 512 MB of DDR3 RAM and 4 GB of flash memory. Its standard ports include 1 × RJ-45 interface for 100 Mbit LAN, 1 × USB 2.0 host and client, 1 × SDIO (for SD memory cards), etc. Compared with the PC, its hardware resources are closer to those of a satellite. It is more suitable for the hardware platform of the satellite emulation node.

To ensure that the satellite works reliably in the harsh space environment, and can deal with the emergencies instantaneously, stability and timeliness are specific requirements for the satellite operating system. Therefore, real-time operating system replaces Linux or Windows operating system and becomes the priority of the satellite operating system.

RTEMS [3] is a hard real-time embedded operating system developed by the United States military. It has been widely used in communications, aerospace, industrial control, military and other fields. Table 1 shows the performance comparison between RTEMS and other systems.

As listed in Table 1, compared with the time-sharing operating system Linux and the real-time operating system VxWorks, RTEMS works with higher reliability and better real-time nature [4]. So far, RTEMS has been running on THEMIS, REXUS 5 rockets and other aircraft [5]. Hence, RTEMS is chosen as the operation system of the satellite emulation node in our scheme.

Table 1. Performance comparison between RTEMS and other systems [6]

	Interrupt delay		Context transfer	
	System load balancing			
	Max	Average	Max	Average
RtLinux	13.5	1.7	33.1	8.7
RTEMS	14.9	1.3	16.4	2.2
VxWorks	13.1	2.0	19.0	3.1
	Heavy load			
RtLiunx	196.8	2.1	193.9	11.2
RTEMS	19.2	2.4	213	10.4
VxWorks	25.2	2.9	38.8	9.5

It has been proved that TCP protocol works inefficiently when it is directly employed in space networking. Currently, the delay-/disruption-tolerant networking (DTN) originated from the deep space communications is widely recognized as the most suitable technology to be employed in space networking. Many research problems will focus on the studies of DTN protocols and their uses in space networking. Thus, DTN protocol stacks will be ported to the emulation node. Since the ION-DTN developed by the California Institute of Technology's Jet Dynamics Laboratory (JPL) is a implementation of the DTN network protocol architecture, we consider porting the ION-DTN to the operating system of the emulation node.

The general design scheme of the emulation node is shown in Fig. 1. The Beaglebone Black embedded core board is used as the hardware development platform. The satellite operating system selects RTEMS real-time operating system and runs ION-DTN on it to realize the data transmission based on the DTN stack.

Since RTEMS does not provide a network adapter driver for the BBB broad, one of the difficulties in this scheme is to develop the network adapter driver for RTEMS on BBB broad. In addition, the architecture of ION-DTN is more complex, and how to port the ION-DTN on RTEMS is also one of the difficulties. Hence, to complete the design scheme of the emulation node proposed in this paper, the following problems should be solved:

- Development of network adapter driver for BBB board;
- Transplantation of ION-DTN protocol stack;
- Development application for ION internetworking.

3 Establishment of RTEMS Development Environment

The development and operation of embedded applications are carried out on different machines. So first of all, we need establish a cross-development environment. Embedded development environment software is usually composed of operating system, compiler, debugger and so on.

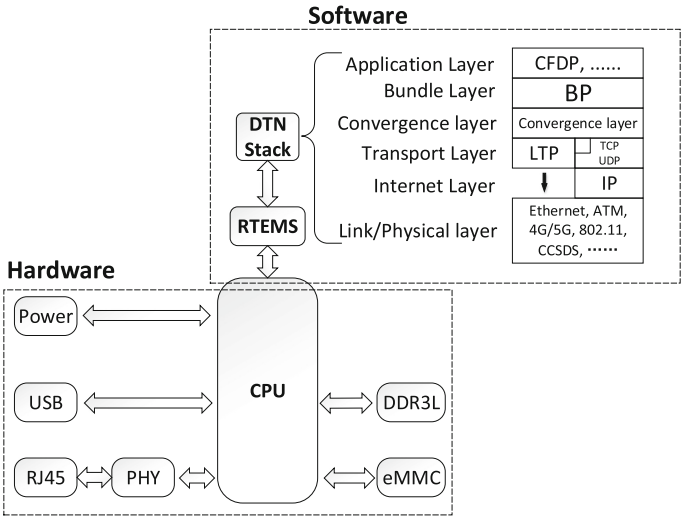

Fig. 1. System general program design

Development environment of the design scheme is based on Ubuntu 16.04 64-bit and RTEMS 4.11.

Download the latest version of RTEMS from official website. Its open source code already contains Beaglebone Black's BSP. The work approach [8] is shown in Fig. 2.

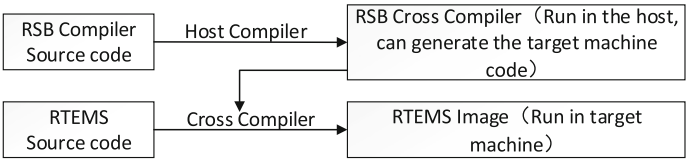

Fig. 2. Development diagram

RTEMS-Source-Build (RSB) is RTEMS official cross compiler. First configure the RSB to match the development environment and the RTEMS. Then run the bootstrap script file in the root directory of the source code. After that, RTEMS kernel trimming and configuration can be implemented by running the configuration file. For example, the parameter "-target" is set to compile the target, and thus "-target = beaglebonblack" indicates the target is BBB board; The parameter "Cenable" indicates that a function is enabled, so that "Cenable-networking" can enable the network protocol stack function.

After the configuration and cross-compiling aforementioned, RTEMS system image for BBB board can be achieved and transplanted on BBB board by the command "Make install" [10].

4 Development of the Network Adapter Driver for RTEMS

Since RTEMS does not have the network adapter driver associated with BBB board, we need design and implement its network adapter driver before other application transplantation. Consider the implementation of U-Boot and FreeBSD driver as references. Although the implementation of U-Boot network driver is simple and easily understood, we cannot duplicate its implementation to RTEMS since it should also obey the open source protocol GPL3. As the network protocol stack of RTEMS is developed from TCP/IP protocol stack of FreeBSD, the API function of RTEMS's protocol is similar to that of FreeBSD. Thus, it is convenient for porting driver. In addition, the development of FreeBSD is according to the BSD open source protocol, which allows the code modification done by the developers arbitrarily. The details of the implementation for the network adapter driver on RTEMS is described in Fig. 3 [11].

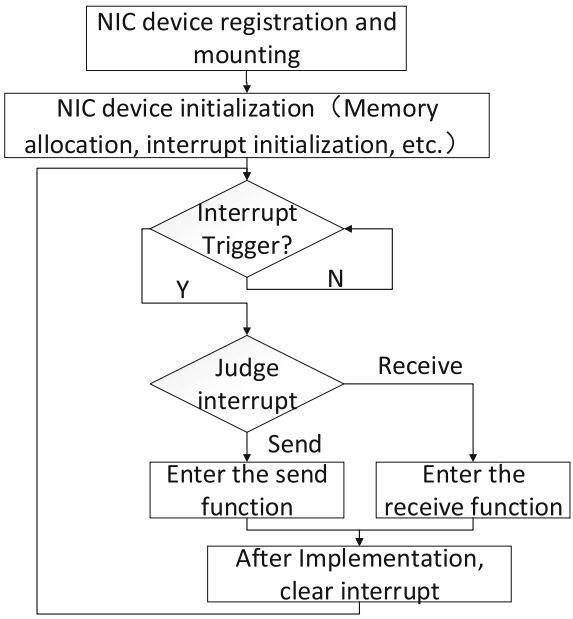

Fig. 3. Network adapter workflow

The transplantation of the network adapter driver is according to the following steps.

- Registration and mounting for network adapter;
- Operation of the initial configuration on network adapter;
- Implemention of interrupt on network adapter;

– Implemention of transceiver on network adapter;
– Function test on network adapter;
– Application test on network adapter.

Some key issues should be noted in the process of development. For example, BBB board fulfills the network adapter management via the TI instruction "CPSW". Then interface function associated with the network adapter registration is named "cpsw attach", and it will be called by the network usage or the system initialization. Usually, CPU is authorised to read data from cache first. However, the transmission method adopted by DMA is directly transmitting data via memory. Thus, it is important for DMA that the operation on the data cache should be disabled while data exchanged with the network adapter.

After the development of network adapter driver, we carry out its functional test by the instruction set "shell" integrated with RTEMS.

5 ION-DTN Transplantation

5.1 DTN Protocol Stack Transplantation

DTN, as an overlay networking architecture, is widely recognized as the most suitable technology to be employed in space networking. Therefore, many researchers focus on the studies of the heterogeneous protocol stacks and overlay networking of DTN for their use in space networking. ION-DTN developed by JPL is one of the implementation for DTN. We thus transplant ION-DTN to RTEMS in our emulation node scheme.

The DTN protocol stack has been described in Fig. 2. As the fact that the source code of ION-DTN already contains the Makefile file and the application for RTEMS, transplantation can be implemented by the Makefile file modification.

First of all, configure the environment variable in the Makefile to make it suitable for the development environment. The objective of Makefile is to generate "ion.exe" file. The source files include ICISOURCES, LTPSOURCES, BPSOURCES, TESTSOURCES, etc. If one wants to add a new function, the source file list needs to be modified, e.g. entering a new source file into the list. Suppose that the node expects to provide the ability of the space router, the function of BP packet transmitting-receiving in Bundle layer will be added on list.

In addition, RTEMS static library should be also added in the Makefile file, because the interface function between Bundle layer and OSI protocol layer needs the support of sockets integrated with RTEMS. After that, the "ion.exe" file is generated once again, and then converted into an image. Burn it to the SD card, BBB board thus can boot from such SD card.

5.2 Application for ION Internetworking

Interplanetary Overlay Network (ION) is an implementation of the space internetworking based on DTN protocol stack. Its data flow processing is shown in Fig. 4 [12].

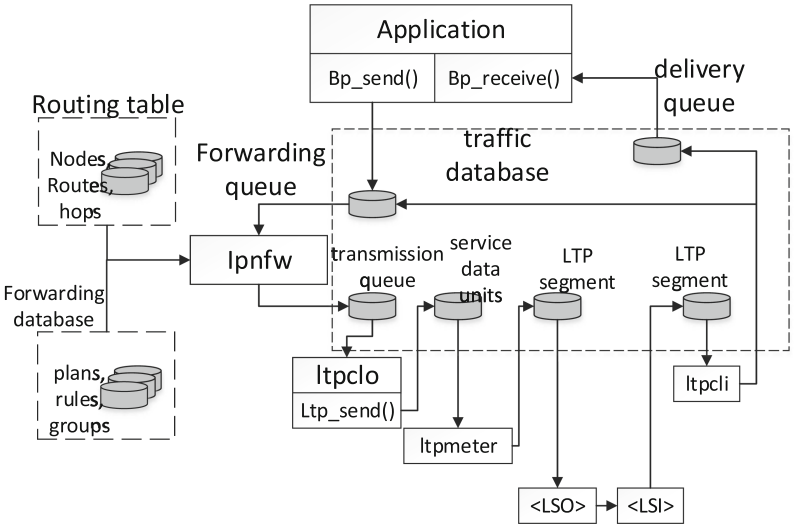

Fig. 4. ION data stream

The source code of ION-DTN contains a demo file, ionrtems.c, which implements the loopback function of the BP package. However, it only transfers data packets between the Bundle layer, which does not involve using the RTEMS network protocol and the physical layer Ethernet frames.

After reconfiguration aforementioned, the pseudoshell function in ionrtems.c file is modified to obtain the command "bpdriver" and "bpcounter" as follows.

```
bpdriver -1000 ipn:1.1 ipn:3.1 -1000
bpcounter ipn:3.1
```

where the first parameter of bpdriver indicates the size of the bp packet, e.g. 1000 is 1000 KB, the second and third arguments indicate the source node and the destination node respectively, and the last argument indicates the number of the bp packets. The parameter of bpcounter indicates the source node. With the help of these two commands, the entire protocol stack is enabled to send and receive BP packets successfully.

6 Emulation Results and Discussion

According to the scheme proposed in this paper, we implement the development of emulation node network adapter driver, ION-DTN protocol stack transplantation and application for ION internetworking.

The performance of our designed emulation node will be evaluated via the following experiments. Most of our emulation parameters are listed in Table 2.

We use ping as the diagnostic tool to test the reachability of our emulation node on the network. Taking our designed emulation node as the destination

Table 2. Emulation parameters

Parameter	Setting/Value
Space Router Hardware	BBB Platform
Bandwidth Limit	100 Mbps
Liunx host PC	Ubuntu 16.04
BP Hosted Transport	No
ACK Packet Size/bytes	70
LTP Block Size/bytes	240
LTP Segment Size/bytes	1400
MTU Size/bytes	1500
Number of samples	Independent test 8 times

BP: Bundle protocol layer MTU: Maximum transfer unit

host, we send Internet Control Message Protocol (ICMP) Echo Request packets to the target host from a Linux host and wait for the ICMP Echo Replies. Figure 5 is the output of running ping on Linux for sending 50 probes to the target node. It lists the statistics of the entire test and the red line in Fig. 5 indicates the value of average round trip time (RTT). In this experiment, the first RTT spent the longest time since the source host at first needs to go through ARP broadcasting via the router to find the physical address corresponding to the target host IP address. The value of other RTTs was fluctuated around the average slightly which indicates the emulation node network adapter working stably.

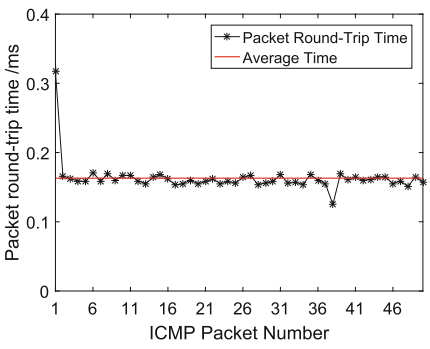

Fig. 5. The Emulation node sends ICMP packets

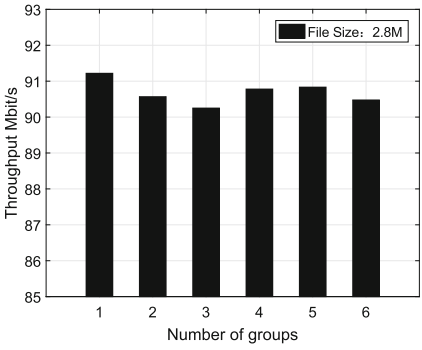

Fig. 6. FTP test

Further, we develop an FTP server program to test the transmission ability of our embedded emulation node. Here, function rtems_initialize_ftpd is called to initialize the ftp, and the structure rtems_ftpd_configuration is used for the corresponding parameters configuration, including ftp port, the maximum number of connections, etc. The tests had been done after setup the FTP server

on RTEMS, and the measurement results are illustrated in Fig. 6. It shows that uplink rate of network adapter on BBB board is about 32 Mb/s and the downlink rate is 90 Mb/s, which are approximate to the rate available at network adapter on BBB board.

As can be seen from Fig. 6, the transfer rate is maintained at 90 Mbps, the network adapter has a good performance.

No.	Source	Destinatior	Protocol	Length	Info
1	10.0.0.3	10.0.0.4	LTP Segme	94	ipn:1.1 > ipn:2.1
2	10.0.0.4	10.0.0.3	LTP Segme	55	Report segment
3	10.0.0.3	10.0.0.4	LTP Segme	60	Report ack segment
4	10.0.0.4	10.0.0.3	LTP Segme	118	ipn:2.0 > ipn:1.0
5	10.0.0.3	10.0.0.4	LTP Segme	60	Report segment
6	10.0.0.4	10.0.0.3	LTP Segme	48	Report ack segment

No.	Source	Destinatior	Protocol	Length	Info
3	10.0.0.3	10.0.0.4	Bundle	84	ipn:1.1 > ipn:2.1
5	10.0.0.3	10.0.0.4	Bundle	181	ipn:1.1 > ipn:2.1
7	10.0.0.3	10.0.0.4	Bundle	181	ipn:1.1 > ipn:2.1
9	10.0.0.3	10.0.0.4	Bundle	181	ipn:1.1 > ipn:2.1

Fig. 7. BP, LTP, CFDP protocol

Finally, we do experiments to test whether the DTN protocol stack working properly on RTEMS. We send data from our emulation node to a Linux host, and both of them load the DTN protocol for data transmission. In the tests, a packet analyzer named Wireshark runs on the Linux host to capture the packets transmitted between two nodes, and some of the results are illustrated in Fig. 7. As can be seen in Fig. 7, two nodes can employ BP and LTP protocol for data delivery successfully. It shows that the embedded emulation node can load DTN protocol effectively and run it properly, which makes our designed emulation node meet the requirements for space network protocol testing.

7 Conclusion

This paper proposed a scheme to design and implement an embedded emulation node, which provided an approach to emulate a precision network environment for the measurement of network protocols in the space information network. The design scheme uses the Beaglebone Black (BBB) embedded core board owned the approximate hardware performance as a satellite has, ports the real-time operating system RTEMS to BBB board, and then loads and runs the ION-DTN on it. The experiment results show that the function and performance of the designed emulation node can meet the requirements for space network protocol testing.

References

1. Beaglebone Black User Manual. https://cdn.sparkfun.com/datasheets/Dev/Beagle
2. AM335x Sitara Processors Users Guide. http://www.ti.com.cn/cn/lit/er/sprz360i/sprz360i.pdf
3. RTEMS Ada Users Guide. https://www.rtems.org/
4. Faxin, Y.: Comparison and analysis of commonly used embedded real-time operation. J. Comput. Appl. **4**, 761–764 (2006)
5. Dong, J., Li, Y., Yang, Q., Zhai, J.: Real time evaluation of embedded operating system oriented to space system. J. Comput. Eng. Des. **1**, 114–120 (2013)
6. Sun, L.: A comparative study of four popular embedded real-time operating systems-VxWorks, QNX, ucLinux, RTEMS. J. Comput. Appl. Softw. **8**, 196–197 (2007)
7. Zhou, J.: Research on key technologies of space integrated information network based on DTN. D. ACM SIGBED Rev. **11**, 20–25 (2014)
8. Yang, H.: Porting of RTEMS embedded operating system. J. Softw. **12**, 108–113 (2015)
9. Bloom, G., Sherrill, J.: Scheduling and thread management with RTEMS. J. ACM SIGBED Rev. **11**, 20–25 (2014)
10. Fan, C., Gui, X.: Development of RTEMS real-time system board support package. J. Appl. Single Chip Microcomput. Embed. Syst. **6**, 35–38 (2005)
11. Huazhong, W., Chen, H.: Design of Network Driver Based on RTEMS. J. Electron. World **2**, 129–130 (2014)
12. ION.pdf. http://ipnsig.org/wp-content/uploads/2015/05/Whats-new-in-ION.pdf