THE IMPLEMENTATION OF TCP/IP STACK AT THE LABORATORY DEVELOPMENT BOARD SDK 2.0

R.A. Nurmuhametov, A.N. Pushinskaya, T.S. Chernyaeva
Scientific advisor:PhD, Associate professorY.A. Chursin
Language advisor: Senior teacher N.V. Demyanenko
Tomsk Polytechnic University, Russia, Tomsk, Lenin str., 30, 634050
E-mail: nurruslan@yandex.ru

Annotation: The article deals with the implementation of TCP / IP stack at the board SDK 2.0. The article contains some information about the structure of laboratory board SDK 2.0, the interface Ethernet, the organization of TCP/IP stack. In summing up the authors present a practical example of the implementation of TCP / IP stack.

The purpose of the work is the implementation of data transfer between board SDK 2.0 and a personal computer through the organization of TCP / IP stack.

The training laboratory board SDK 2.0 is designed to study the principles of organization of microprocessor systems, structure and functioning of basic components (memory, input-output controllers, memory subsystems, etc.), obtaining programming skills of the embedded systems for various applications. Laboratory board SDK 2.0 is a functionally complete device, which has a high-performance processor core ARM7, a great variety of communication interfaces (RS232C, RS485, CAN 2.0, Ethernet, IEEE 802.15.4), a collection of input-output devices and means of communication with the operator.

The one of the communication subsystem of SDK 2.0 including a controller Ethernet LAN91C111 is the interface Ethernet. Ethernet is a family of computer networking technologies for local area networks (LANs). Ethernet was commercially introduced in 1980 and standardized in 1983 as IEEE 802.3. Since its commercial release, Ethernet has retained a good degree of compatibility. The Ethernet frame format has influenced other networking protocols. A wide range of remote communication features are possible when Ethernet connectivity is added to the embedded designs. End users benefit through cost and time savings since they can centrally monitor, control and service their embedded systems over the Internet instead of physically being there.

Each piece of information transmitted on an Ethernet network is sent in something called a packet. A packet is simply a chunk of data enclosed in one or more wrappers that help to identify the chunk of data and route it to the correct destination.

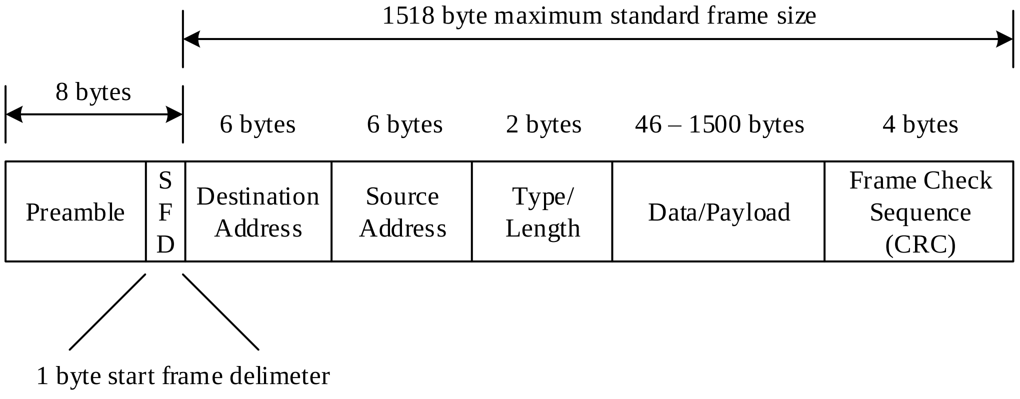

Figure 1 – Ethernet frame format

The 802.3 specification divides the preamble into two sections. The first section is a 56 bit (7 byte) field plus a 1 byte field called the starting frame delimiter (SFD). The preamble is not typically used in modern Ethernet networks as it's function is to provide signal startup time for 10Mbps Ethernet signals. Modern 100Mbps, 1000Mbps or 10Gbps Ethernet use constant signaling, which avoids the need for the preamble.

Destination and Source Address: these two sections of the frame are likely the most commonly understood in that they contain the MAC address for the source "transmitting system" and the destination "target system". The type / length field is used to identify what higher-level network protocol is being carried in the frame (example: TCP/IP) The data / payload field is what we typically consider most important as it is the data in which we are transmitting. The diagram specifies a range between a minimum of 46 bytes and maximum of 1500 bytes. The end of the frame contains a 32 bit field which is a Cyclic Redundancy Checksum (CRC). This is a mechanism to check the integrity of a frame upon arrival at its destination. The CRC is generated by applying a polynomial to bits which make up the frame at transmission. This same polynomial is used at the receiving station to verify the contents of the frame have not changed in transmission.

TCP/IP (Transmission Control Protocol/Internet Protocol) is a protocol family that is used world-wide as the standard for network communications between user processes. TCP/IP is a collection of protocols that include TCP, IP, UDP, ICMP, ARP, RARP, and others. To send data over a TCP/IP network requires four steps or layers:

- Application. Encodes the data being sent;
- Transport. Splits the data into manageable chunks, adds port number information;
- Internet**.** Adds IP addresses stating where the data is from and where it is going;
- Link. Adds MAC address information to specify which hardware device the message came from, and which hardware device the message is going to.

The data exchange between the computer and SDK 2.0 is required the implementation of two sockets, one for the target host, the other for the sending host. In Ethernet domain socket is a combination of IP address and port number, which uniquely defines a single network process. To create the TCP - server the programming language C++ was chosen, and TCP - client is implemented using uVision Keil 3.0.

Figure 2 represents a general view of connection SDK2.0 and a PC to implementation the data exchange between them.

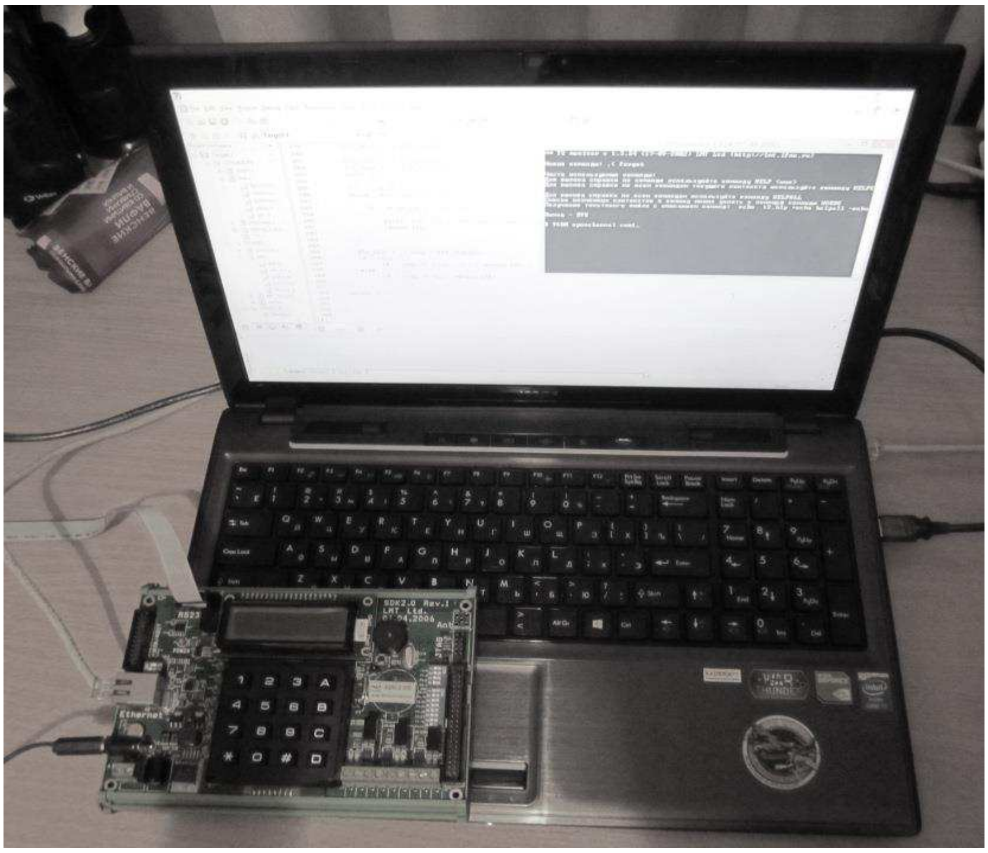

Figure 2 – General view of connection PC and SDK 2.0

As a result of this work the research of interface Ethernet was carried out. The organization of TCP/IP stack was provided by connection SDK 2.0 to PC. The data transmission between SDK 2.0 and computer performed similarly conversation between two persons. If two persons start talking at the same time , they soon discover it (collision detection). In this case, they fall silent and wait for some time, and then one of them starts talking again. Another person waiting for the first person finish talking and then he can start talking. Everyone has their own name (analog unique Ethernet-address). Every moment when someone starts talking, he called the name of the person with whom there was talk, and his name, followed by the transmission of the message.

In the completion of this work implementation of the Web server is expected. It is contribute to decision of one problem faced by embedded systems that run remotely or unattended is that of monitoring the system's status.

References:

1. Douglas E. Comer. Internetworking with TCP/IP. Vol I: Principles, protocols, and architecture. Fourth Edition. New Jersey, 1995.

2. Charles M. Kozierok. THE TCP/IP GUIDE. A comprehensive, illustrated Internet protocols reference. San Francisco, 2005.

3. William Stallings. High-Speed Networks: TCP/IP and ATM Design Principles. Prentice-Hall, 1997.

4. Fact Pandit. A TCP/IP Server written in C#. [Network resource]. – Access mode: http://www.codeproject.com/Articles/5733/A-TCP-IP-Server-written-in-C