

UDK 681.527.2

Sgadov S.A.¹

¹senior lecturer ZNTU

IMPLEMENTATION OF PROTOCOL OF SERIAL PORT EXCHANGE WITH A HERCULES DEVELOPMENT KIT BOARD

The Hercules Development Kit development board is constructed on the basis of the TMS570LS310 microcontroller and can be used as a base of the control onboard computer. In this regard there is a problem of implementation of debugging by the serial protocol for data exchange with this board.

In this work author offer a software protocol of data which assumes that the microcontroller is an initiator of data exchange. Before data exchange the microprocessor sends a line command through a serial port which defines type and the direction of data. The modeling program on the PC listens to the specified serial port and after receipting of a command performs the corresponding operations on exchange.

The following algorithm of data exchange with the PC was used:

1. The microcontroller sends a request to GET
2. It expects for some time of arrival of data on port
3. If the timeout is exceeded, then repeats from p. 1
4. The accepted data are processed.
5. The microcontroller sends a request to SEND
6. The microcontroller sends data.

Data are transferred in internal representation. At the same time it is necessary to remember that in the TMS microprocessor uses reverse order of bytes both integer and float data types. Development have been provided in C++ and the

class system working with a serial port was created for PC and for the microcontroller.

The class **TTY** is responsible for initialization and byte data transmission through a serial port. For transfer of data types volume more than one byte created the derivative **class EnhTTY**. It expands a base class with the methods oriented to transfer of typified data.

Thus:

- **virtual int ReadInt(void) virtual void WriteInt(int data)** are intended for transfer of integer numbers.

Methods

- **virtual void WriteIntVector** (const int data [], int size);

- **virtual void ReadIntVector** (int * data, int size); - for transfer of arrays of integer numbers.

- **virtual double ReadDouble(void)** and **virtual void WriteDouble(double data)** - real numbers, and

- **virtual void WriteDoubleVector** (const double data [], int size);

- **virtual void ReadDoubleVector** (double data [], int size); - for transfer of arrays of real numbers and structures from real numbers.

- **virtual void ReadString(string & sendestring)** and

- **virtual void WriteString(string recievedstring)** - transfer zero - the named lines.

The library for transfer by serial port consists of modules: armtty.h, armtty.cpp, typedtty.h, typedtty.cpp, proconst.h, typestruct.h.

The architecture of classes for work with a serial port is similar to the classes considered above with the following changes in the class TTY.

- Work with port is based on hardware drivers created by HALCoGen. The Connect method is absent because the initialization of port is made from the constructor implicitly by sciInit call () from sci.h - so the speed of port is established in the 9600th baud.

- **TIMEOUT** is set in terms of cycles of the processor, but not in milliseconds.

- **uint32 _sciReceiveByte** method appeared which receives byte from port at a low level

- There is a member of the class **bool IsTimeout** which is established in true, at achievement of a timeout of input, reset is recommended to be carried out **ClearTimeout method call ()**.

Thus, the developed class library is invariant in relation to using program, and implementation of methods of a class differs for different architecture (the PC and TMS). Therefore, similar approach can be implemented also in relation to other boards and microprocessors.