Implementation of the USB interface in the instrumentation for sound and vibration measurements and analysis

Conference Paper · October 2003

DOI: 10.1109/IDAACS.2003.1249539 · Source: IEEE Xplore

3 authors, including:

Andrzej Podgórski
Warsaw University of Technology
9 PUBLICATIONS **45** CITATIONS

Implementation of the USB Interface in the Instrumentation for Sound and Vibration Measurements and Analysis

Andrzej Podgórski, Rafal Nedwidek, Michal Pochmara
Warsaw University of Technology, Faculty of Electronics and Information Technology, Institute of Radioelectronics, 00-665 Warsaw, ul. Nowowiejska 15/19, A.Podgorski@ire.pw.edu.pl

Abstract: *The USB (Universal Serial Bus) interface is now commonly used in computer hardware. It assures asynchronous serial data transmission with the speed which is much bigger than that in RS 232 interface. In the paper the essay of an USB implementation in the unit intended for sound and vibration measurements and analysis is presented. The aim of the USB interface implementation was to assure fast data transmission between the unit and a PC. The USBN9602 chip of National Semiconductors was used as an interface controller. The software was developed in the measurement unit and in a PC for testing the application. The different modes of data transmission were tested. The achieved results of the tests performed in different operating systems are presented. The conclusions concerning required modification of the implementation in order to achieve faster transmission are drawn.*

Keywords: - measurement instrumentation interfacing, Universal Serial Bus, fast data acquisition, acoustic and vibration signal processing

1. INTRODUCTION

The USB interface is now more and more common in computer hardware. It assures asynchronous serial data transmission with the theoretical speed up to 10 Mbits/s. It enables one to connect up to 127 units to one bus controller, so-called *host*. It assures +5 V powering in two-line standard cable. Now, the most commonly used is the USB in version 1.1, which was specified in 1998 [1]. The new version, signed as 2.0, enables one to achieve the transmission speed up to 400 Mb/s.

The novelty, presented in the paper, is the essay of an implementation of the USB interface in a sound and vibration meter and analyser [2]. We wanted to verify the thesis that with the USB interface it is possible to use the autonomous measurement unit as an acquisition card in a PC. In order to fulfil the requirements of such application in the 16-bits analyser taken into account – SVAN 912AE – the transmission speed of the interface should not be lower than 96 kBytes/s (the sampling frequency of the analyser is equal to 48 kHz).

2. USB INTERFACE DESCRIPTION

2.1. PHYSICAL STRUCTURE OF THE INTERFACE

In the USB 1.1 specification two classes of units are defined: *low speed* - working with the speed from 10 kbits/s to 100 kbits/s (mouse, keyboards, joysticks) and *full speed* - working with the speed from 500 kb/s to 10 Mb/s (loudspeakers, microphones, cameras). The USB interface is compatible with the *PC Plug and Play* standard and it assures the possibility of the connection between a unit and a working system without any risk of damage (the *Hot Swap* function). The *master-slave* USB bus enables simultaneous transmission to or from a few units but it is not possible to make a transmission between the units. The master controller – *host* – is used for servicing all transmission requests. Units are connected to the host in a star structure using four line cables ended with a standard connector (cf. Fig. 1).

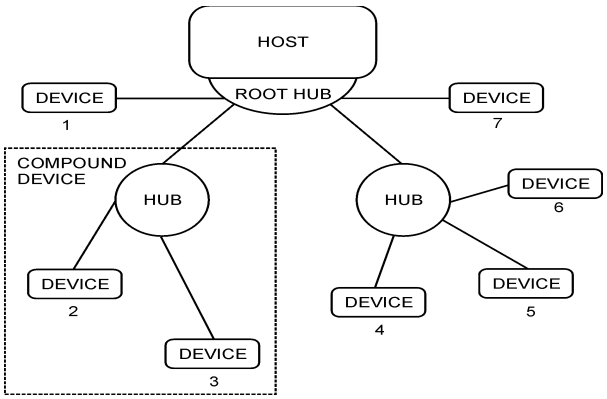

Fig. 1 - Typical topology of USB interface.

2.2. LOGICAL STRUCTURE OF THE INTERFACE

The transmission channel from an application working on the host to the unit is named as *pipe*. It is the realisation of the direct host-unit connection. On the host side the pipe is ended by an input or output data buffer, while on the unit side – by so-called *endpoint* (cf. Fig. 2), which is an abstraction of the certain function fulfilled by the unit. The set of all pipes in the unit is called interface. More than one pipe can be used in one unit (example: camera with a built-in microphone is connected to the host using three pipes).

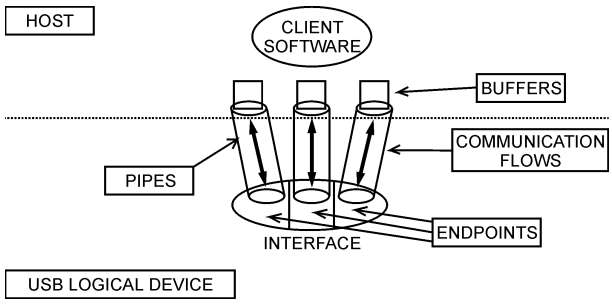

Fig. 2 - Host – unit communication structure.

The USB can be modelled by three layer structure (cf. Fig. 3): applications using the USB units constitute the upper layer, the controllers of these units create the middle one and the USB bus realising the physical transmission forms the lower one. The upper layers use

the lower layers for the communication purposes. The direct communication is possible only with the physical USB bus. The communication channel between the middle layers is called *control pipe*, while the channels for the upper layer are called *data pipes*.

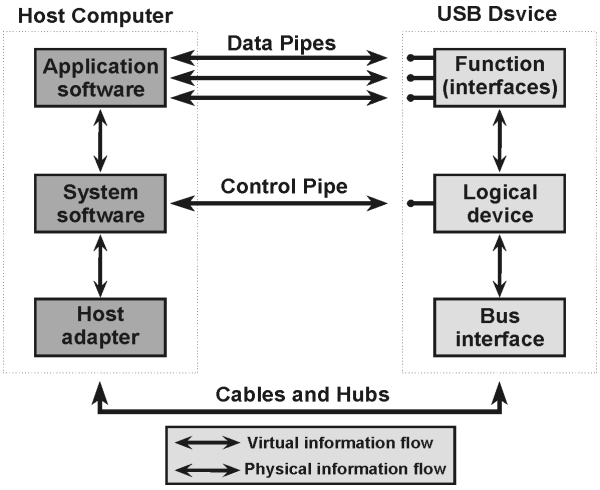

Fig. 3 - Block diagram of three layer USB structure.

2.3. DATA TRANSMISSION

The data transmission between an USB unit and a host is divided into requests. These requests constitute the orders of sending a portion of data. Each request is then divided into series of transactions. The USB controller sends data in portions called frames. Each frame starts every one millisecond. Different transactions coming from different units (different requests) could be placed in one frame. Such structure enables the user to transmit the data simultaneously with different devices. Each transaction is divided into phases and it usually contains *token phase*, *data phase* and *acknowledgement phase*. However, these phases can differ slightly for different types of transactions (four types are defined in the USB standard):

- *Control transfer* used for controlling the communication between a unit and the host. In one frame 8, 16, 32 or 64 bytes are transferred. This type of transmission should be performed without fail. An error is generated when three consecutive data transmission are ill-success. Each unit has the endpoint marked as 0, which is always active, even when the unit is not yet configured.
- *Bulk transfer* used for a transmission of big amount of data, where time is not a critical parameter. The data which is sent can not be lost. However, the capacity of the transfer is not guaranteed, because the data is sent only if other types of transmission do not use the full output of the connection.
- *Interrupt transfer* used for a transmission of small amount of data in a short time. The scheme of the transmission is the same as in the case of bulk transfer.
- *Isochronous transfer*, which guarantees the transfer of a steady number of bytes in the amount of time. It enables one to send up to 1023 bytes in one frame. This type of transmission guarantees the capacity and the periodicity of the transfer but the data can be lost in the case of errors (the handshake phase is not implemented).

3. HARDWARE DESCRIPTION

The USB interface described briefly above was implemented in a sound and vibration meter and analyser controlled by two digital signal processors (DSP). The USBN9602 integrated chip of National Semiconductor [3], [4] (cf. Fig. 4) was used as a host of the USB interface. This chip fulfils the requirements of 1.0 and 1.1 USB standards. It contains all necessary circuits: transceiver of the signals, voltage regulator, serial interface engine, FIFOs endpoint / control and microcontroller interface. It has also two different interfaces: parallel 8-bits and serial, called MICROWIRE. The parallel interface can work in two addressing modes: multiplexed or normal. In the described implementation the mentioned above, serial MICROWIRE, interface was used, because the existing hardware of the analyser does not have the possibility of the parallel port controlling. The parallel ports of the DSP (in this case DSP56303 of Motorola [5]) used in the measurement instrument were already engaged.

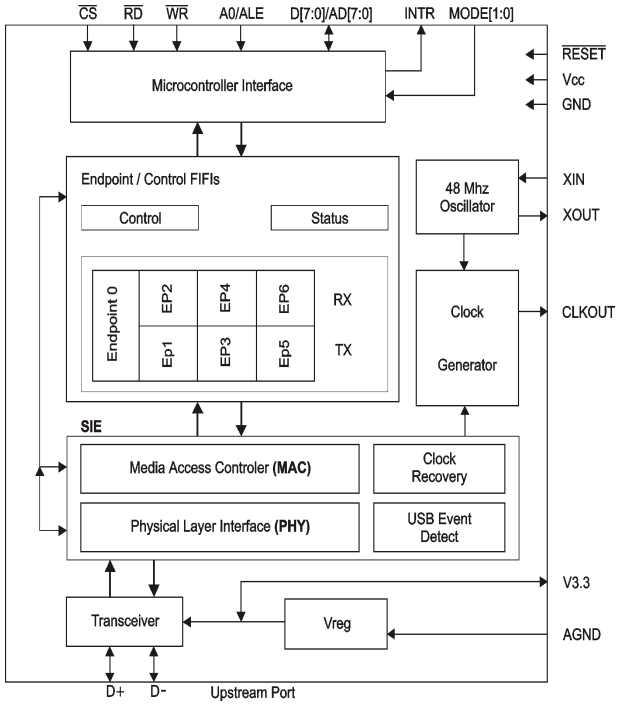

Fig. 4 - Block diagram of the USBN9602 controller.

The Microwire / Plus interface (cf. Fig. 5) enables one to connect USBN9602 chip to the DSP using serial port. Four lines are used for this purpose (Serial Input, Serial Output, Chip Select and Serial Clock). The Microwire interface ensures double sided transmission between the USBN9602 and a unit using three transmission modes: Interface Basic Read, Interface Standard Write and Interface Burst Write. Each mode contains the control / address field defining all parameters of the transmission. Transmitted data are formed, stored and controlled by the FIFOs endpoint / control part of USBN9602.

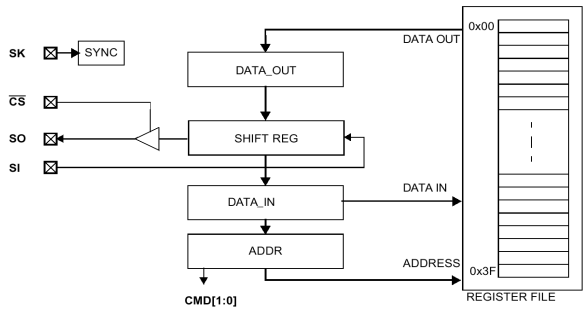

Fig. 5 - Block diagram of the Microwire interface.

The Microwire interface, according to the data specification published up to the beginning of the year 2003, could work with up to 3 MHz clock. The transmission of 1 byte requires 10 clock pulses and it should last 3.33 μs (cf. Fig. 6). In this case the interface should assure the transmission speed up to 300 kBytes/s. After the hardware implementation of USBN9602 the investigation performed proved that the Microwire interface can work properly only with up to 1.5 MHz clock. It decreases the maximum possible transmission speed to 150 kBytes/s. After the interpellation to the producer the National Semiconductor changed the data specification concerning the mentioned above frequency of the Microwire interface clock.

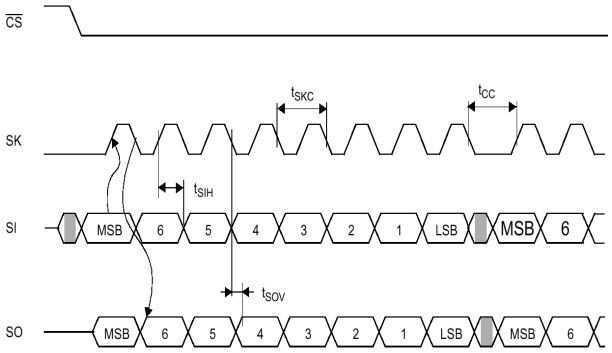

Fig. 6 - Microwire interface timing.

4. DESCRIPTION OF THE SOFTWARE

4.1. SOFTWARE DEVELOPED IN THE MEASUREMENT UNIT

The software was developed to control the USBN9602 chip and the communication between a host and DSP56303 of the SVAN 912AE analyser. The software controller has to implement so-called *Standard Device Requests* defined in the USB standard. Thanks to them the basic communication between the device with the USB interface and the host is performed. It should assure the proper identification, configuration and survey of the unit with the USB. This software contains two parts: main programme which initiates host, the communication parameters and waits for upcoming interrupts and a set of procedures serving the interrupts from USBN9602 and controlling data transfer. Main programme, at the beginning, defines all registers and flags used by DSP56303, ports of DSP used for the communication with USBN9602, host registers and host functional states as well as requests which can appear on the USB bus. Next, ESSI1 serial port of DSP56303, USBN9602 and the descriptors of DSP are initiated and the interrupts in DSP are enabled. Finally, main programme stays in a loop waiting for interrupts. Interrupt procedures identify the reason of the request, serve it and transfer data between the analyser and a PC.

4.2. PC APPLICATION SOFTWARE

The application software was also developed to handle data transmission between the analyser and a PC. This software contains universal USB driver from Thesyscon [6]. The written programme enables the user to configure the analyser as the USB unit connected to a PC. The proper *Vendor* request should be sent to the measuring instrument in order to transfer data to or from the unit. Three types of Vendor requests were implemented in the instrument: single data read out, data write down, constant data read out. The application for testing data transmission speed in different transaction modes (bulk, interrupt and isochronous) was also written. It enables the user to get or set the parameters of the device, open it or close, get the device and configuration descriptors, get and set the configuration, get and set the parameters of the interface, etc.

5. RESULTS OF THEINVESTIGATIONS

5.1. TEST PERFORMED FORARTIFICIAL DATA

In these tests [7] data were prepared and stored in the unit's memory. On each *Vendor* request 64 bytes were transferred. The maximum transmission speed which was achieved in this case was equal to 32 kBytes/s. This speed is limited by the USB protocol. The control frames on the USB bus begin each 1 ms, so in 1 second it is possible to send to the unit up to 1000 requests. In fact, data are transferred not in the same frame in which comes *Vendor* request but in the next one. It theoretically enables one to send up to 32 kBytes/s. In addition, it is possible to notice that other devices connected to the same USB bus slow down the transmission speed.

In order to speed up the transmission data were transferred continuously in 64-byte packages. It requires the USB controller event register checking of the proper pipe and loading in the proper time the consecutive data to the output queue. Additionally, in order not to loose time which is required for sending out data package, it is possible to use two FIFO buffers. In time when one buffer is sent out, the second one is filled by data. The tests were performed using three different transfer modes: *Interrupt*, *Bulk* and *Isochronous*. The best results were obtained in the case of *Bulk* transfer mode (cf. Table 1).

Table 1. Results of data transmission speed tests for different transfer modes

Transfer mode / Analyser with USB	***Interrupt***	***Bulk***	***Isochro-nous***
connected alone to USB bus	34.5 kBytes/s	82 kBytes/s	2 kBytes/s
connected to USB bus with USB camera	32 kBytes/s	27 kBytes/s	2 kBytes/s

Additionally, the speed tests with the artificial data were performed for different operating systems of Microsoft installed on a PC. The results of the investigations are given in Table 2. In Windows XP the frames start every 0.5 ms, while in Windows 98 and Windows 2000 every 1 ms. In Windows XP the reaction for the host requests is faster but in the same time more time is required to send the control packages. This results in bigger speed in *Interrupt* mode and smaller in *Bulk* mode.

Table 2. Results of data transmission speed tests for different operating systems

Transfer mode / PC operating system	***Interrupt***	***Bulk***	***Isochro-nous***
Windows 98 of Microsoft	34.5 kBytes/s	82 kBytes/s	2 kBytes/s
Windows 2000 of Microsoft	31 kBytes/s	80 kBytes/s	2 kBytes/s
Windows XP of Microsoft	42 kBytes/s	76 kBytes/s	2 kBytes/s

The influence of the data package length on the transmission speed was also checked. The tests performed for data packages of 16, 32 and 64 bytes proved that this parameter does not change the speed. The deviations were not bigger than 0.2 kBytes/s.

5.2. TEST PERFORMED FOR MEASUREMENT DATA

In the tests with the measurement data the input signal was sampled with the frequency up to 48 kHz. It was not possible to change smoothly the sampling frequency because the tests were performed in the unit with the installed clock. The dividers by 2, 3, 4 and 5 were used in order to slow down this frequency. In performed tests the *Bulk* mode was used with transferring of one 64-byte package on each *Vendor* request and with the transmission of data stream (2048 bytes were sent on one *Vendor* request). The test were performed in order to assess the sampling frequency of the input signal using which it is possible to transfer continuously the data stream. The Fast Fourier Transform was calculated on the transferred measurement data coming from the generator of the sinusoidal signal. The observation of the averaged spectrum enabled the user to find out any discontinuity of the input signal.

In the case of one 64-byte package transfer on each *Vendor* request the highest sampling frequency of the measurement instrument which does not cause the discontinuity of the input signal is equal to 9.6 kHz (48 kHz sampling frequency divided by 5). For the higher sampling frequencies the data in a package are continuous but the discontinuities appear between the packages (cf. Fig. 7 and 8).

In the case of data transmission stream on *Vendor* request the highest sampling frequency which does not cause the discontinuity of 16-bits input signal, coming from the measurement unit, is equal to 24 kHz (cf. Fig. 9). Some discontinuities appear in the data stream for 48 kHz sampling frequency (cf. Fig. 10).

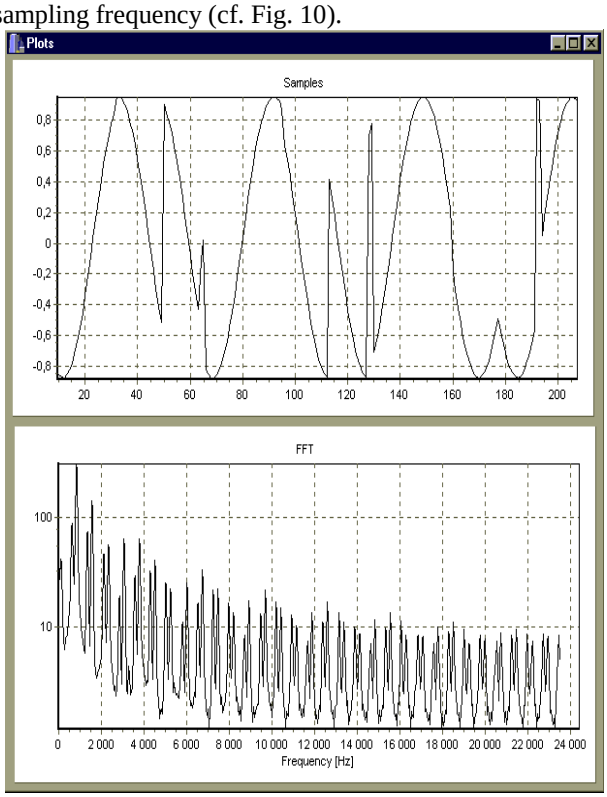

Fig. 7 - Time history and spectrum of 1 kHz input sinusoidal signal sampled with the frequency equal to 48 kHz in the case of one 64-bytes package transfer on each *Vendor* request.

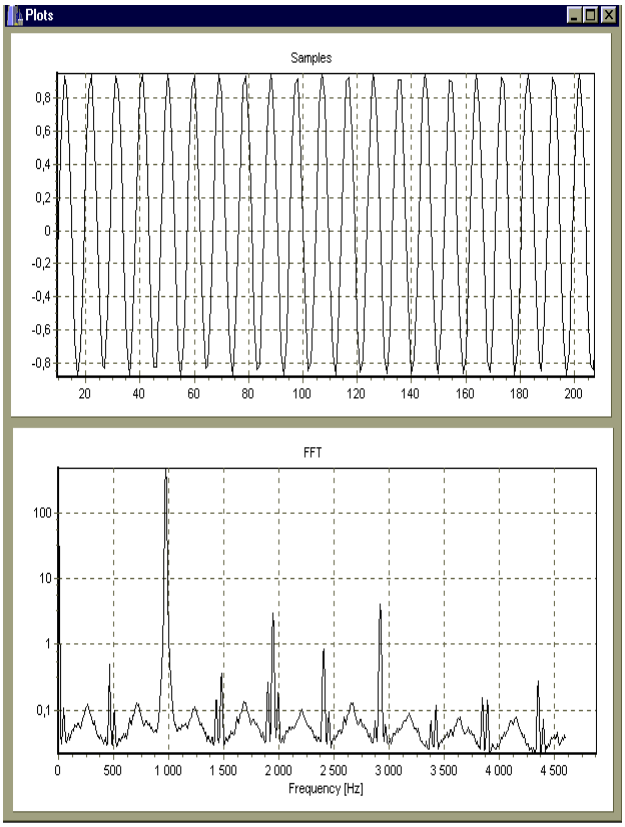

Fig. 8 - Time history and spectrum of 1 kHz input sinusoidal signal sampled with the frequency equal to 9.6 kHz in the case of one 64-bytes package transfer on each *Vendor* request.

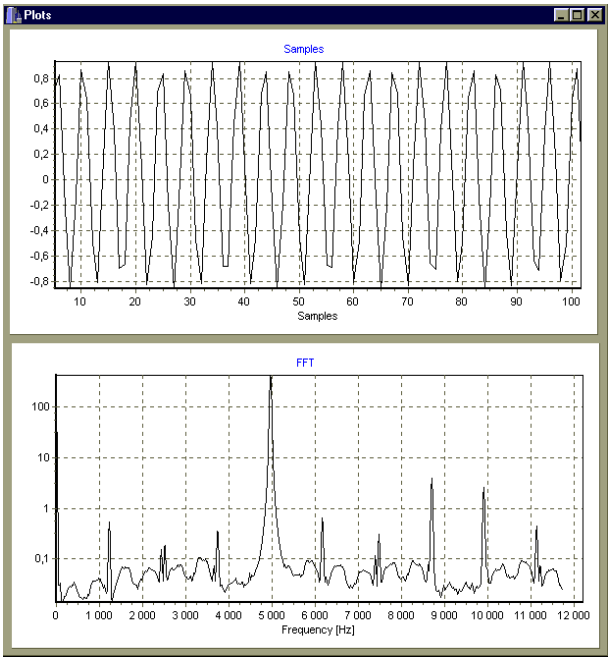

Fig. 9 - Time history and spectrum of 5 kHz input sinusoidal signal sampled with the frequency equal to 24 kHz in the case of 2048 bytes transmission on *Vendor* request.

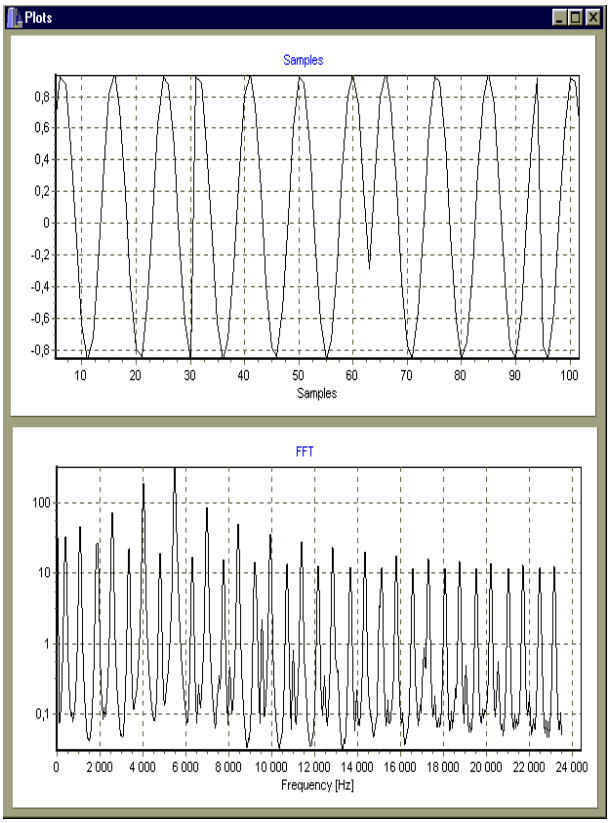

Fig. 10 - Time history and spectrum of 5 kHz input sinusoidal signal sampled with the frequency equal to 48 kHz in the case of 2048 bytes transmission on *Vendor* request.

Taking into account the results obtained for artificial data it is possible to draw out the conclusion that the highest sampling frequency achieved in the described implementation which enables the continuous transfer of 16-bits measurement data is not bigger than 41 kHz. The performed tests proved that the obtained results do not depend on the operating system installed on a PC (the differences were smaller than 5 %). However certain disturbances on the averaged spectrum were observed on the PCs with slower processors (e.g. Celeron with 333 MHz clock). They were probably caused by the tasks with the higher priorities performed on a PC and the number and the kind of tasks executed in a background on a PC.

6. CONCLUSIONS

The results of the research show that the communication between the instrument and a PC was established. There is no problem in sending or receiving small amount of data but the achieved speed is not satisfactory. The obtained results depend mainly on the transaction mode used for data transfer and the set of the USB units connected to a PC. The results do not depend on the operating system installed on a PC. The best results were achieved when in the host the *bulk* transfer was implemented. This speed was equal to 82 kBytes/s. It is about 8 times faster then it is possible to obtain using the RS 232 interface but not enough for the current requirements for data transfer from the analyser to a PC. In the application of the analyser as the front-end of a PC the speed of 96 kBytes/s is needed.

In order to fulfil these requirements some new investigations should be conducted. First of all, instead of USBN9602 controller, some other products with the serial interface have to be found, implemented in the measurement instrument and tested. Another solution is to abandon the usage of the serial interface and check the transmission speed which is assured by the parallel interface already existing in USBN9602. Unfortunately, in order to perform this attempt, more complex redesign of the current measuring instrument is required.

As the first attempt the USB controller AT89C5131 from ATMEL was selected for the future investigations and tests [8]. The serial interface of this unit, according to the data sheet, should be few times faster than that of USBN9602.

7. REFERENCES

[1] *Universal Serial Bus Specification*, Revision 1.1, 1998

[2] *SVAN912AE User's Manual*, Svantek Ltd, 2000.

[3] USBN9602 (Universal Serial Bus) Full Speed Function Controller with DMA Support, *National Semiconductors.*

[4] USBN9602 Firmware Description, *National Semiconductors.*

[5] *DSP56303 24-bit Digital Signal Processor User's Manual*, Motorola.

[6] *Thesyscon Universal USB Device Driver for Windows 98, Windows Millennium and Windows 2000*, Reference Manual.

[7] R. Nedwidek, M. Pochmara. Implementation of the USB interface in the measurement unit, *Master Thesis*, Institute of Radioelectronics, Faculty of Electronics and Information Technology, Warsaw University of Technology, Warsaw, 2003 (in Polish).

[8] Atmel Corporation. AT89C5131 USB controller data sheet (pdf file), *www.atmel.com/dyn/products/*