Development of vibration spectrum analyzer using the Raspberry Pi microcomputer and 3-axis digital MEMS accelerometer ADXL345

Conference Paper · April 2017

DOI: 10.1109/MEMSTECH.2017.7937525

6 authors, including:

Marek Iwaniec
AGH University of Science and Technology in Kraków
86 PUBLICATIONS 255 CITATIONS
SEE PROFILE

Andriy Holovatyy
Lviv Polytechnic National University
24 PUBLICATIONS 86 CITATIONS
SEE PROFILE

Vasyl Teslyuk
Lviv Polytechnic National University
276 PUBLICATIONS 695 CITATIONS
SEE PROFILE

Mychajlo Lobur
Lviv Polytechnic National University
198 PUBLICATIONS 561 CITATIONS
SEE PROFILE

Development of Vibration Spectrum Analyzer Using the Raspberry Pi Microcomputer and 3-Axis Digital MEMS Accelerometer ADXL345

MarekIwaniec[1], Andriy Holovatyy[2], Vasyl Teslyuk[2], Mykhaylo Lobur[2], Kostyantyn Kolesnyk[2], Marta Mashevska[3]

1. Department of Process Control, AGH University of Science and Technology, POLAND, Krakow, al.AdamaMickiewicza 30, E-mail: iwaniec@agh.edu.pl
2. CAD Department, Lviv Polytechnic National University, UKRAINE, Lviv, S. BanderySt. 12, E-mails: aholovatyy@yahoo.com, vasyl.m.teslyuk@lpnu.ua, mlobur@polynet.lviv.ua
3. Department of Information Systems and Technologies, Institute of Enterprise and Advanced Technologies, UKRAINE, Lviv, GorbachevskyySt. 18, E-mail: marta.ippt@gmail.com

Abstract - **In the paper, the spectrum analyzer of vibration accelerations using the Raspberry Pi microcomputer and 3-axis digital MEMS ADXL345 accelerometer has been developed. The system includes accelerometer driver for Raspberry Pi, software for data acquiring and processing from the acceleration sensor, conversion of the vibration acceleration signals from time domain into frequency domain using DFT, graph plotting of the vibration accelerations and their spectra.**

Keywords –**vibration acceleration monitoring and spectrum analysis system, 3-axis MEMS accelerometer ADXL345, single-board Raspberry Pi microcomputer, vibration acceleration, spectral analysis, discrete Fourier transform (DFT), computer numerical control (CNC).**

I. INTRODUCTION

Modern technological processes require the continuous monitoring of many parameters of technological equipment. The most important parameters are mechanical parameters such as periodic mechanical vibrations. Such control is needed in the various fields: in the semiconductor electronics (vibration control in crystal growing systems), in the microelectronics industry (vibrations in photolithography equipment), in the mechanical engineering (vibrations in machinery and beatings of machine parts), in the automobile industry (vibration control of individual car units and the whole car in general), in the railway transport (sensors for detecting a train approaching), in the power industry (vibration monitoring of gas turbine blades) in the aircraft industry (control of turbine beats) and etc. [1,2].

Vibration monitoring systems make possible to solve a lot of the problems coupled with vibrations that appear during the technological equipment operations (for example CNC machine operation) [1-4]. Thus, development of vibration monitoring and spectrum analysis systems for technical objects is an important task.

II. HARDWARE DESIGN OF VIBRATION ACCELERATION MONITORING AND ANALYSIS SYSTEM

Nowadays, for developmentof automated monitoring and control systems the single-board microprocessor or microcontroller based platformsare often used. They are based on the different hardware architectures such as Arduino, STM32, Raspberry Pi and others [5]. The hardware of our system is built on the single-board computer Raspberry Pi 3 Model B (Fig.1) and 3-axis digital accelerometer ADXL345 (Fig.2) [6,7]. TheADXL345accelerometer is mounted on the monitoring object (for example on the milling head of CNC milling machine) and connected via I2C bus to the Raspberry Pi microcomputer. The microcomputer acquires (collects) data from the sensor and processes them. The ADXL345 accelerometer is used as a sensor for vibration measurements. ADXL345 is a miniature three-axis digital accelerometer from Analog Devices Inc. It has low power consumption, high resolution (13 bits) and adjustable measurement ranges up to ±16 g. Moreover, the measurement range can be selected from the following numbers: ±2g, ±4g, ±8g and ±16g. The measurement results can be read via SPI (3 and 4-wire) or I2C bus.

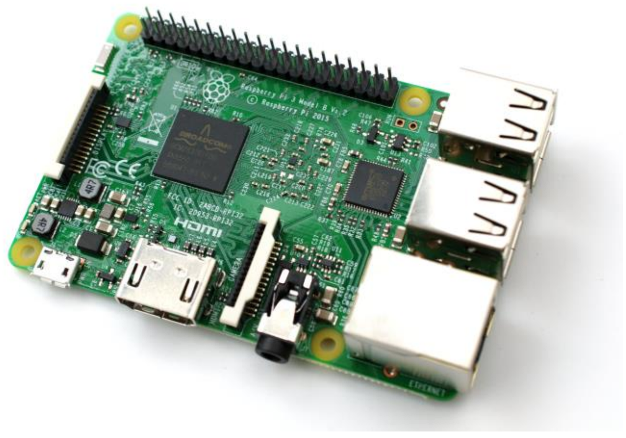

Fig. 1 Single-board computer Raspberry Pi 3 Model B

ADXL345 is the capacitive accelerometer with the bandwidth of 0.05 ... 1600 Hz. This device is ideal (very convenient) for measuring dynamic acceleration, low vibrations, static acceleration of gravity (gravitational force), the movement and tilt angles (inclinometer). Bandwidth describes the ability of the sensor to detect acceleration changes that occur with a high frequency (eg, vibration at a frequency of 1000 Hz). On this characteristic the sample frequency of the embedded ADC of the accelerometer influences which has to be at least twice larger than sample frequency. The maximum sample frequency for ADXL345 is 3200 Hz [7].

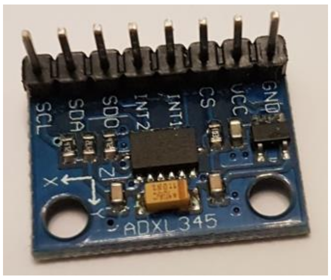

Fig. 2 Three-axis digital MEMS accelerometerADXL345

Accelerometer Adjustment

The ADXL345 accelerometer is connected to the Raspberry Pi microcomputer via I2C bus (Fig. 3). The I2C supports a standard data rate of 100 kHz and also higher 400 kHz. The ALT_ADDRESS line is responsible for the accelerometer address on the I2C bus. At the high logic level on the line the 7-bit address with a value of 0011101 (0x1D) is assigned to the accelerometer, at the low level - 1010011 (0x53). For the correct interface operation the SDA and SCL lines have to be tied to the supply voltage through the pull-up resistors Rp (10kOhm). Because of the limited data rate of 400 kHz, the maximum output data rate of acceleration measurement should not exceed 800 Hz.

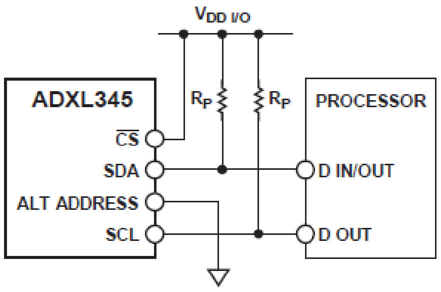

Fig. 3 Connection diagram of ADXL345 via I2C bus (address 0x53)

The information exchange with the accelerometer is as follows. Firstly, Raspberry Pi sends via I2C bus the register address to be read from or written to. Then, the data are sent for writing to this register or for reading the specified number of bytes from the device.

Initialization of the ADXL345 accelerometer consists of three steps:

1. Measurement mode activation by setting 3rd bit in the POWER_CTL register (address 0x2D). Write a value of 0x08.
2. Data format definition in the DATA_FORMAT register (address 0x31).
3. Offset – values writing OFSX, OFSY, OFSZ into the OFFSET - registers. The addresses of the OFFSET - registers are 0x1E (OFSX), 0x1F (OFSY), 0x20 (OFSZ).

Using the DATA_FORMAT register we set the output data format in the registers DATAX, DATAY, DATAZ. Using the FULL_RES bit of the DATA_FORMAT register we configure the resolution measurement of acceleration. In establishing the bits included enhanced resolution mode, which depends on limits set by bits Range, the measurement result is calculated based on the coefficient of 3.9 mg/LSB. Zero value bit sets a fixed resolution of 10 bits, and the measurement result and the calculated ratio will depend on the chosen limit. Justify DATA_FORMAT bit register sets the alignment method of measurement result in the registers DATAX, DATAY, DATAZ, bit values equal to 1, sets left alignment, 0 - right.

Table 1.g Range Setting

Setting		g Range
D1	D0	
0	0	±2g
0	1	±4g
1	0	±8g
1	1	±16g

In the BW_RATE register we adjust the output data rate.

Register 0x2C – BW_RATE (Read/Write)

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	LOW_POWER	Rate			

The Rate bits are used for setting the output data rate, Table 2.

Table 2. Bandwidth and Output Data Rate Setting in the BW_RATE register (address 0x2C)

Output Data Rate (Hz)	Bandwidth (Hz)	Rate Code, D0…D3 bits in the BW_RATE register
3200	1600	1111
1600	800	1110
800	400	1101
400	200	1100
200	100	1011
100	50	1010
50	25	1001
25	12.5	1000
12.5	6.25	0111
6.25	3.13	0110
3.13	1.56	0101
1.56	0.78	0100
0.78	0.39	0011
0.39	0.20	0010
0.20	0.10	0001
0.10	0.05	0000

In our case, we configure measurement range by setting 3 bits D3, D1 and D0. When the FULL_RES bit is set to 1, the device operates in the highest resolution 3.9 mg/LSB. It does not matter what measurement range is determined, one bit represents acceleration in 3.9 mg. If the FULL_RES bit is not set to 1, the ADXL345 accelerometer will run in 10-bit mode and bits D1 and D0 range will determine a number of mg/LSB. The values of mg/LSB for different range configurations are given in Table 3.

Table 3.mg/LSB in different range configurations

Range	mg/LSB
All g-range when bit FULL_RES bit = 1	3.9
±2g, 10 bit mode	3.9
±4g, 10 bit mode	7.8
±8g, 10 bit mode	15.6
±16g, 10 bit mode	31.2

Writing the value of 0x0B into the DATA_FORMAT register, the range of ±16 g with the maximum 13-bit resolution will be selected. Now the operating accelerometer is ready, but without defined offset values. The offset values can be set for each axis in the OFFSET- registers (OFSX, OFSY, OFSZ) or programmatically. The offsets must be individually calculated for each accelerometer. Each accelerometer has its own unique offset values. The initialization code is as follows:

```
fd = wiringPiI2CSetup(ADXL345_ADDRESS);
temp = ADXL345_GetRegisterValue(ADXL345_DEVID);
if (temp != ADXL345_ID)  status = 0;
printf("Device ID: 0x%x, Status: %u\n", temp, status);
ADXL345_SetRegisterValue(ADXL345_BW_RATE,
(0x00|ADXL345_RATE(0x0A)));
ADXL345_SetRegisterValue(ADXL345_DATA_FORMAT,
(0x00|ADXL345_RANGE(ADXL345_RANGE_PM_16G)|ADXL345_FULL_R
ES));
ADXL345_SetRegisterValue(ADXL345_FIFO_CTL,
(0x00|ADXL345_FIFO_MODE(0x2)|ADXL345_SAMPLES(0x1E)));
ADXL345_SetRegisterValue(ADXL345_INT_MAP,(0x00|~ADXL345_
WATERMARK));
ADXL345_SetRegisterValue(ADXL345_INT_ENABLE,(0x00|ADXL34
5_WATERMARK));
ADXL345_SetPowerMode(0x1);
```

The raw values of the measured accelerations for each axis X, Y, Z are stored in the read-only registers DATAX0, DATAX1, DATAY0, DATAY1, DATAZ0 and DATAZ1. The registers DATAX0, DATAX1, DATAY0, DATAY1, DATAZ0, DATAZ1 are 8-bit data registers with the addresses from 0x32 to 0x37. The value of the acceleration for each axis is represented by two bytes: DATAx0 contains LSB and DATAx1 – MSB, where x – measurement axis X/Y/Z. The registers DATAX0 (0x32) and DATAX1 (0x33) contain the data on the X-axis, DATAY0 (0x34) and DATAY1 (0x35) - the data on the Y-axis, DATAZ0 (0x36) and DATAZ1 (0x37) - the data on the Z-axis. It is recommended to read all the registers 6 bytes per session to prevent possible data loss, because transactions between the individual contents of registers may change. The data reading function is as follows:

```
void ADXL345_GetXyz(int16_t* x, int16_t* y, int16_t* z)
{
*x = ADXL345_GetRegisterValue(ADXL345_DATAX1) << 8;
*x += ADXL345_GetRegisterValue(ADXL345_DATAX0);
*y = ADXL345_GetRegisterValue(ADXL345_DATAY1) << 8;
*y += ADXL345_GetRegisterValue(ADXL345_DATAY0);
*z = ADXL345_GetRegisterValue(ADXL345_DATAZ1) << 8;
*z += ADXL345_GetRegisterValue(ADXL345_DATAZ0);
}
```

Now, the data in the 16 - bit format have to be converted into a value of g. For this, we multiply them a fixed-rate constant, which varies depending on the specific measurement range and resolution accelerometer. In our case the mode used with a maximum resolution of 13 bits, has a coefficient of 3.9 mg/LSB. So this means that data must be multiplied by 0.0039 to convert them to acceleration value in units of g (1g = 9.80665 m/s^2).

```
void ADXL345_GetAccXYZ(double* acc_x, double* acc_y,
double* acc_z)
{
int16_t x, y, z;
ADXL345_GetXyz(&x, &y, &z);
*acc_x=(double)x*ADXL345_MG_LSB_FULL_RES/1000*9.80665;
*acc_y=(double)y*ADXL345_MG_LSB_FULL_RES/1000*9.80665;
*acc_z=(double)z*ADXL345_MG_LSB_FULL_RES/1000*9.80665;
}
```

Accelerometer Calibration and Setting Offset Values

The accelerometer is a mechanical structure with elements that can freely move. These moving elements are very sensitive to mechanical stress (shocks, shaking) and even much more sensitive than its electronics. The offset is an important accelerometer metric because it determines a threshold for measuring a real acceleration. Additional measurement errors can also occur when mounting an accelerometer system. These errors can be caused by stress in the printed-circuit board while mounting and using various compounds to the component. Therefore, accelerometer calibration is recommended to perform after the system installation.

One of the well described and accurate calibration methods is to use twopointsperaxis. The ADXL345 accelerometer has a three-axisdesign. Therefore, weneed to make measurementsinsixpoints. Our developed application has the function forreading theaccelerometermaximumsandminimums. Withthesevaluestheoffsetvaluesandgainfactorscan be calculated. The calculated offset values and gain factors give the newcalibratedaccelerometerreadings.

Beforetakingthesemeasurements, wehaveto mount theaccelerometerwiththe Z axisparalleltotheupdirection. Forexample, ifthe accelerometerison a table, the ADXL345 breakoutboardhas tobeorientedin such a way that the Z dataisconstant.Eachtimein ordertomeasure a differentaxis, theblock on whichthe ADXL345 breakoutismounted, is turned. An X-Y-Z axissymbolonthebreakoutboardhelpswithorientingineachdirection.The measurementshave to be taken ineachaxisdirectionin a +1 g and −1 g field.

Theoffsetvaluesandgainfactorsarecalculatedfrom the measured outputs usingthefollowingequations [8]:

$$A_{OFF}[g] = 0.5 \times (A_{+1g} + A_{-1g}) \tag{1}$$

$$Gain = 0.5 \times \left(\frac{A_{+1g} - A_{-1g}}{1g} \right) \tag{2}$$

The new values of the real acceleration in the X, Y and Z – axis are calculated by the following formulas:

$$accX = (x - A_{OFF\_X}) / GainX \tag{3}$$

$$accY = (y - A_{OFF\_Y}) / GainY \quad (4)$$

$$accZ = (z - A_{OFF\_Z}) / GainZ \quad (5)$$

The calculated offset values for each axis direction are programmed into the OFFSET-registers OFSX, OFSY and OFXZ, respectively. As with all registers in the ADXL345accelerometer, the valuesin the OFFSET-registers are not savedwhen power is off. The power off and on to the accelerometer returns OFFSET-registers to their default values of 0x00. Thus, the offset values have to be recalculated again or can be stored in the Raspberry Pi memory.

III. Software Development for Data Acquisition and Processing from ADXL345 Accelerometer

For communication with the ADXL345 accelerometer the device driver and special software have been developed. The device driver adjusts the sensor according to user requirements, reads and processes data from the accelerometer. The obtained acceleration values for each axis are saved in the specified files ax.dat, ay.dat and az.dat, respectively. The ADXL345 driver is written in C using the wiringPiI2C library and compiler - gcc [9].

For converting a signal of the vibration acceleration from time domain into frequency domain the special software has been developed. The developed software for DFT calculation uses the software library FFTW [10,11]. Spectral data obtained from DFT of vibration acceleration signals for each axis are saved in the files: spectrum_ax.dat, spectrum_ay.dat and spectrum_az.dat, respectively.Shell-scripts accel_plot.sh, spectrum_plot.sh and spectrum_plot_all.sh are written for plotting graphs. The software has been developed under OS Raspbian Jessie. The developed software provides the user interface of the system, data acquiring, processing and saving in the specified files on the microSD memory card, and their visual demonstration. The software can process data from different sensors that allows performing of multi-channel measurements.

Some functions of the developed driver for ADXL345 accelerometer and special software are as follows:

```
/*! Read a value (byte) from the register */
unsigned char ADXL345_GetRegisterValue(unsigned char
registerAddress);
/*! Write a value (byte) into the register */
void ADXL345_SetRegisterValue(unsigned char
registerAddress, unsigned char registerValue);
/*! ADXL345 initialization on I2C */
unsigned char ADXL345_Init(char commProtocol);
/*! Setting standby mode/measure mode */
void ADXL345_SetPowerMode(unsigned char pwrMode);
/*! Read output data on each axis */
void ADXL345_GetXyz(int16_t* x, int16_t* y, int16_t* z);
void ADXL345_GetAccXYZ(double* acc_x, double* acc_y,
double* acc_z);
/*! Enable/disable of activity detection */
void ADXL345_SetActivityDetection(unsigned char
actOnOff, unsigned char actAxes, unsigned char actAcDc,
unsigned char actThresh, unsigned char actInt);
/*! Offset-values setting */
void ADXL345_SetOffset(unsigned char xOffset, unsigned
char yOffset, unsigned char zOffset);
/* DFT calculation */
void adxl345_spectrum_analysis(unsigned intno_samples,
unsigned intfs);
```

IV. Research Results and Their Analysis

The research results are graphically presented in Fig. 4 - 9. In Fig. 4-6 the change of the vibration accelerations has been shown which are received from the accelerometer on the x-, y-, z - axes, respectively. The graphs in Fig. 7-9 show the spectrums of the vibration accelerations for each axis (X, Y and Z).

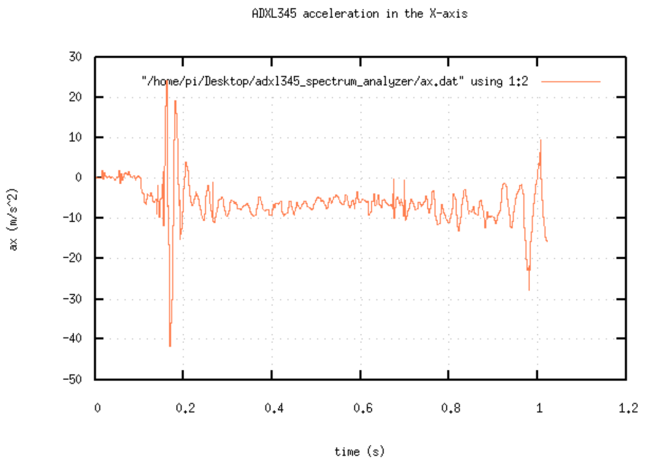

Fig.4Change of the vibration acceleration in the X-axis

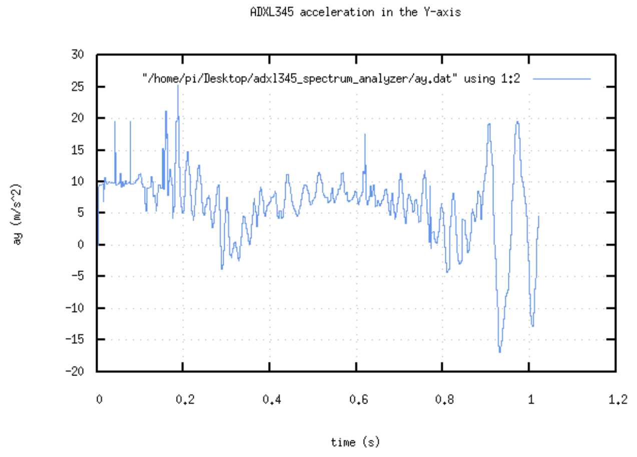

Fig. 5 Change of the vibration acceleration in the Y-axis

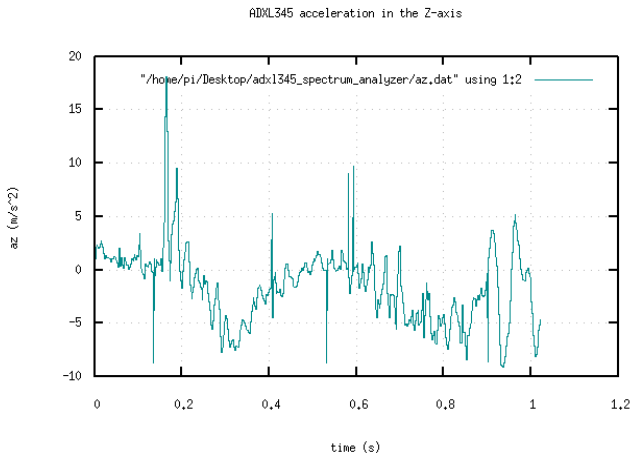

Fig. 6 Change of the vibration acceleration in the Z-axis

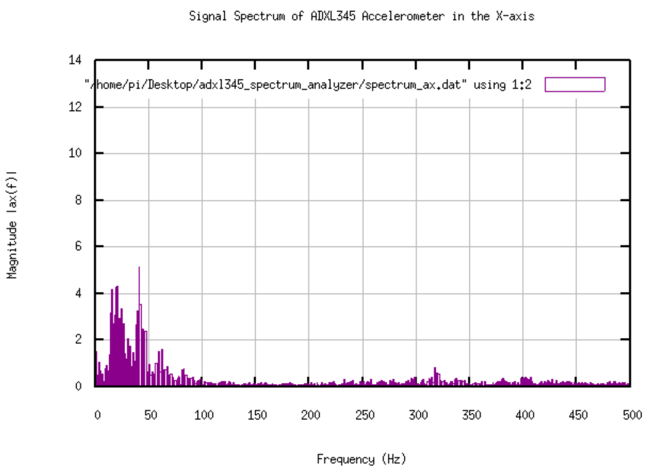

Fig. 7 Spectrum of the vibration acceleration in the X-axis

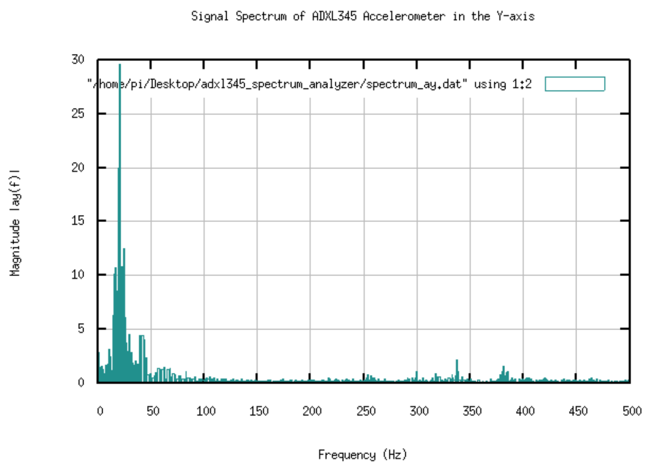

Fig. 8 Spectrum of the vibration acceleration in the Y-axis

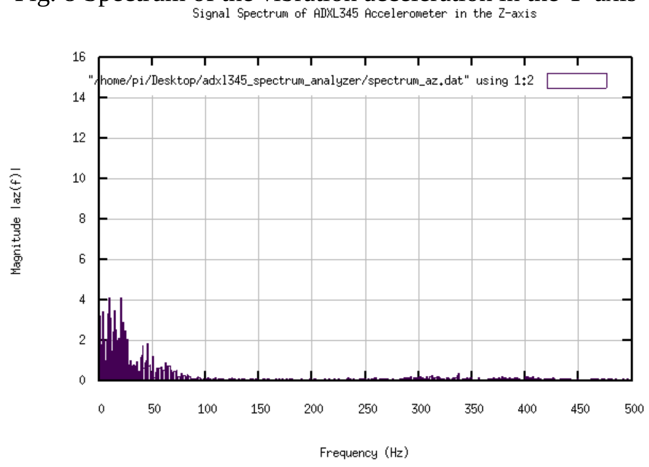

Fig. 9 Spectrum of the vibration acceleration in the Z-axis

V. Conclusion

Thespectrum analyzer of vibration accelerations using the Raspberry Pi microcomputer and three-axis digital accelerometer ADXL345 has been developed. The system can operate in real-time mode and allows us to perform vibration signal analysis in the time and frequency domains.

The driver for the ADXL345 accelerometer has been developed which performs the sensor adjustment, data acquisition and processing. Software for data processing, DFT calculation of the vibration acceleration signal from the time domain into frequency domain and graph visualization of the data has been developed.

References

[1] N. S. Marine, Prof. Dr. S. Nagmode, Prof. R. D. Komati. Vibration Measurement System with Accelerometer Sensor Based on ARM. – Intern. Journal of Emerging Technology and Advanced Engineering. – Vol. 4 Issue 3, April 2014 – pp. 5.

[2] Adam Hjort& Mans Holmberg. Measuring mechanical vibrations using an Arduino as a slave I/O to an EPICS control system. – Uppsala University, Sweden. –June 2015. - pp. 25.

[3] SubimalBikashChaudary. Vibration Monitoring of Rotating Machines Using MEMS Accelerometer. – Intern. Journal of Scientific Engineering and Research. – Vol. 2 Issue 9, September 2014 – pp. 7.

[4] Системимоніторингу і контролю вібраційhttp://www.omative.com/КонтрольВибрации.html

[5] TarasTeslyuk, PavloDenysyuk, AndriyKernytskyy, VasylTeslyuk.Automated Control System for Arduino and Android Based Intelligent Greenhouse // Proceeding of the XIth International Conference "Perspective Technologies and Methods in MEMS Design", MEMSTECH’2015, 2-6 September 2015, Polyana, Lviv, Ukraine. 2015. – P. 7 – 10.

[6] Raspberry Pi 3 Model B https://www.raspberrypi.org/-products/raspberry-pi-3-model-b/.

[7] 3-Axis, ±2 g/±4 g/±8 g/±16 g Digital Accelerometer Data Sheet ADXL345.http://www.analog.com/media/en/-technical-documentation/data-sheets/ADXL345.pdf.

[8] Electronic resource:Application Note from Analog Devices.

[9] Electronic resource: wiringPi library for Raspberry Pi: http://wiringpi.com/reference/i2c-library/

[10] Prots’ko I., Teslyuk V. AlgorithmofefficientcomputationDST I-IVusingcyclicconvolutions //WSEAS TRANSACTIONS on SIGNAL PROCESSING. – 2014, Vol.10. – P. 278 – 288.

[11] Electronic resource: FFT/DFT library: http://www.fftw.org/