Mobile Application for Visualisation of External Signal on PDA device in Real Time

Radim Štěrba
Department of Measurement and Control,
Faculty of Electrical Engineering and Computer Science
VSB Technical University of Ostrava,
Ostrava-Poruba, Czech Republic
radim.sterba.st@vsb.cz

***Abstract*— The work is focused on a creating of a mobile application that is able to capture and visualize an external audio signal. The aim was to create applications to which can be build at development of a digital oscilloscope. The application is written for the platform. Comapct NET Framework and Windows Mobile. The entire application is written in C #. Today's digital oscilloscopes are fairly large, heavy and mainly dependent on the electrical energy. They are adapted to work in the laboratory. The motivation for this work was to use a handheld computer to be able to replace the digital oscilloscope at least partially. At the end there was an application that was able to accurately visualize the external audio signal up to frequencies around 3000Hz. This limitation is due to the parameters of the Pocket PC (microphone, A/D converter, sound card and processing power PDA). The capture of the external audio signal is used built-in microphone.**

Keywords – Windows Mobile; .NET Compact Framework; C#; PDA; digital oscilloscope

I. INTRODUCTION

Mobile Electronics brings us a user´s comfort and simplifies our lives. PDAs, mobile phones and various communicators are now a normal part of our lives. They are very powerful today due to great technological developments in the PDA. They have powerful processors with high clock frequencies, large system memory of the hundreds of Mega Bytes and also have a large color displays with high resolutions. According to the communication site PDAs are mostly equipped with USB ports. PDA is able to communicate from the wireless communication thanks to IrDA, Bluetooth, WiFi, GSM, GPRS and UMTS.

Today's PDAs can play videos or music, allow you to surf the Internet, are able to locate and navigate through an integrated GPS receiver. With powerful processors and their large operating memory they can run very demanding applications. But it has not always been that way. The PDAs should help with the planning tasks, organizing time and contact management originally. PDAs can be used for office applications (Microsoft Word, Microsoft Excel, and others) besides multimedia content. [1] Picture No 1 illustrates how PDA may look like.

Picture 1: Example of PDA. [2]

The whole problem of visualization of an external audio signal can be divided into three blocks:

- Reading of an external audio signal
- Audio Signal Processing
- Plot the signal on the PDA green

The first block is the reading of the external audio signal (sound). It is worked with the built-in microphone and sound card PDA here. The external audio signal is read by a microphone and converted to a discrete signal. The second block is a discrete signal processing into a form that is suitable as the input data to draw on the screen. The last third signal block is a rendering on the screen PDA devices. The PDAGraph component takes care about the rendering of the signal, which is described below.

The quality of the visualization depends on the hardware quality. Built-in microphones in the PDA are used primarily to record human speech. Frequency range of the speech is about 300Hz to 4000Hz, so we cannot expect quality

visualization of very low and very high frequencies of the input audio signal. Another limiting factor may be a PDA hardware power, especially clock processor speed and memory size of the RAM.

II. APPLICATION PDASCOPE

PDAScope application is designed for PDA devices running Windows Mobile operation system. It is written in C# programming language and uses the platform .NET Compact Framework, which is designed for mobile devices. The application was developed and tested in a developmental background of Microsoft Visual Studio 2008. Windows Mobile 5.0 or higher and support. NET Compact Framework 3.5 and higher are the minimum of the requirements for this application. Preview applications running in the emulator is shown in Picture No 2.

Picture 2: View of application PDAScope.

Picture No 3 shows a block diagram of an application PDAScope. The first block introduce a scan signal, it means that the work is dealt with the hardware and with sound card and built-in microphone in this case. The microphone is used to record sound which converts the sound into an electrical signal. An algorithm, which works with Microsoft classes for sound processing, provides the audio signal processing. The component PDAGraph provides the rendering, which was created within this project. PDAGraph component is described below.

Picture 3: Block diagram of the application.

The first block of a block diagram introduces a part of the program, which cares about the external signal scanning from built-in microphone. The communication with the microphone is provided with WaveIn.cs class. [3] The external analogue signal is converted to discrete signals in the first block, which range from 0 to 255. The level of the analog signals is represented with discrete values from 0 to 255, the analog zero (there is no signal at the output of the microphone) is represented with the discrete values of the 128th. Discrete values from 0 to 127 represent the negative deflections and the discrete values from 129 to 255 represent positive deflections of the analog signals.

The block of the audio signal processing is represented with a PDAScopeWaveIn.cs class and Record.cs class. The PDAScopeWaveIn.cs class generates from raw data the field of samples. There is created a copy of the field in Form1.cs, which continues to work with. This copy serves as the input for rendering and other calculations.

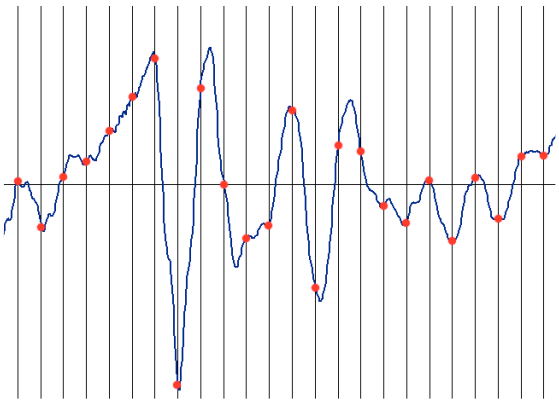
There is a change to render the scanned process over the PDA screen in the last part. The rendering is provided with the user´s component, which is described below. The component contains two methods (AddGraph method and DeleteGraph method) for the rendering. During the redrawing the field of the samples (which is the parameter) is transmitted to the AddGraph method, and then it is depicted. We leave the depicted process on the display for a while (20 ms) and then we use the DeleteGraph method. This method deletes the depicted process. This is cyclically repeated using high redraw speed.

III. SCANNING THE ANALOG SIGNAL

The microphone is connected to the PDA using the sound card. The input signal is the analog signal of course and it must be converted to digital signal by an A / D converter. This conversion is made by using sampling. The signal sampling is a process, when the discretization of the signal happens in its time domain. The current value of the signal (the sample) is recorded at any point in the time. We can see these samples as red circles at the picture No 4. The picture shows that we lose a lot of details by sampling the original signal. Even when we increase the sampling rate we get only a set of discrete points. The Shannon sampling theorem deals with the sampling, which states that an accurate reconstruction of a continuous, frequency-limited signal from its samples will be possible if the signal is sampled with at least twice higher frequency than the maximum frequency of the reconstructed signal, is. The sampling frequency is chosen twice higher plus a small reserve than the maximum transmitted frequency is required in practice. [4]

Man is able to perceive frequencies from 20Hz to 20kHz. Therefore the sampling frequency should be at least 40kHz.

The sampling frequency 44100Hz is usually selected in practice .The recording quality is set to 8 bits per sample in PDAScope application and the sampling frequency is set to 22050Hz. This setting is done in the WaveIn.cs class and can be changed. The PDA hardware options must be respected as a computing power or as a screen display capabilities.



Picture 4: Example of signal sampling [5]

There are many types of microphones in terms of them. Each type is based on a different physical principle and therefore has different properties. Currently, the most common type of microphones is an electrostatic microphone, it is sometimes also known as a condenser. This type is characterized by high input impedance, which is balanced by the input characteristic and high sensitivity. Electrostatic microphones belong between the best for its quality and are often used for a professional recording. There aren´t such high-quality microphones for professional audio recording in the handheld computers. But built-in microphones in handheld computers are good enough for example to capture the spoken word.

The condenser microphone works such way, that acoustic oscillations vibrate the membrane, which is one of the capacitor electrodes, connected to the electrical circuit. The capacitance changes according to the rhythm of the changing the position of the membrane. This change is converted into an electrical signal either powering its own microphone capsule by a very soft source of a polarization tension and the tension is sensed by the preamplifier with a high input impedance, or the capacity of the inserts is used as a part of the high-frequency oscillator, which is disharmonized by the capacity changes, and the frequency signal is demodulated in following circuit [6].

IV. COMPONENT PDAGRAPH

The information about the color of the curve is sent to the component by each visualized signal rendering, the information about the curve thickness and about the look of the curve is sent mainly. This information is presented by a two-dimensional area that contains the coordinates of the points x, y, from which the process is composed.

PDAGraph component is used for rendering the measured data. The component was written specifically for the application of a digital oscilloscope. The component is written in C# language and contains six basic methods for working with the component. All methods are described below.

The component is written that way so it can adapt to the screen of each PDA. Therefore it is always necessary to know the width and height of the panel, which the process is rendered in. These dimensions can be determined by two methods GetPanelWeight and GetPanelHeight. Methods always return an integer value size in pixels.

The rendering of two vertical scales on the right and left side is under way when the process starts. Two methods take care about the rendering, Panel2_Paint method and Panel3_Paint method. Both methods are described below.

AddGraph Method:

AddGraph is a method that directly renders charts. AddGraph has three arguments: an area of points, a line width and color lines. The first argument is an area of points, namely x, y coordinates of the points. The second argument is the width lines, the data type is Integer, and so the entire number is entered. The last argument is the color line, a data type is Color. The method is called: PDA_Graph.AddGraph (points, 3 Color.Blue). Points are an area of points, 3 is the width (the witdh of process will be 3 points) and Color.Blue is the process color (in this case blue). AddGraph method is called every time the screen is redrawing with DeleteGraph method together. The redraw process starts at the first by calling the AddGraph method, which renders the course. Then the fiber, that carries about drawing, is put to the sleep for 20 ms, in order to see a picture of the course and immediately to be deleted by DeleteGrafph method. This is constantly repeated, and thereby we can achieve a continuous rendering.

DeleteGraph Method:

This method deletes the entire contents of the chart. The picture of the process is deleted. The method is called for each redraw. The method has no arguments, so it is enough to call: PDA_Graph.DeleteGraph ().

Panel2_Paint Method:

Method Panel2_Paint draws vertical scale on the left side of the component. The vertical scale is used for better orientation. This method is called immediately at the start that is why the vertical scale is rendered immediately when you start the application.

Panel3_Paint Method:

It is a similar method, such as Panel2_Paint Method is. Panel3_Paint Method depicts the vertical scale on the right side of the component.

GetPanelWidth Method:

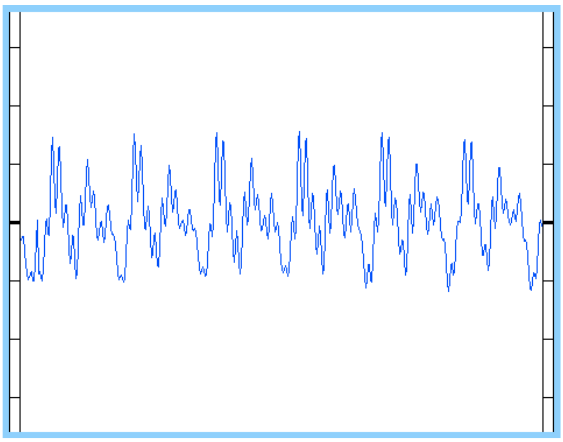
This is a method, which returns the width of the panel, which the process is rendered in. Each PDA has a different screen size and mainly different resolution. And therefore it is necessary to return the size of the component, which is made to render. It is returned both – the width and the height.

Both methods GetPanelWeight and GetPanelHeight are parameter less and always return the entire number as a panel size information in pixels.

GetPanelHight Method:

GetPanelHeight method is used for the return the value of the height of the panel, in which the captured signal is rendered. As it was described before each PDA has a different screen size and therefore it is necessary to dynamically adjust the parameters for rendering.

Preview of the above component is described in Picture No 5. The most important is the white area where the captured processes are rendered. There are vertical scales on sides. Colors, which the process can be rendered by, can be changed in the settings. The blue color is set implicitly.



Picture 5: Preview component PDAGraph

V. THREADS IN THE APPLICATION

The application is written so the threads can be used. Specifically, two threads are in the application, they are responsible for recording the external audio signal, rendering the screen on the PDA.

The synchronization object syncObj EventWaitHandle is defined in the Form1.cs class, which ensures the synchronization of fibers. The context switching thread is manually, it is given by setting EventResetMode.ManualReset. The recording is realized as the first, which is provided by the threads called recTh in a Record.cs class. Class Record.cs contains important MAX_RECORD constant, which determines the time during which the recorded sound will be recorded from the built-in microphone. MAX_RECORD constant is set to 30 ms. The availability of hardware devices (built-in microphone) is checked after the recording initialization. If the built-in microphone is available, the recording starts. The recTh thread is suspended during the recording and the recording is stopped than, the time is given by MAX_RECORD constant. The context switches for thread graphTh after checking the length of the field, where data is stored using the Form1.syncObj.Set (). The GraphTh fibre runs in the class Form1.cs and always waits until the recTh thread stops the recording. The method recTh Form1.syncObj.WaitOne () provides the waiting for a signal from the recTh thread. Then data is copied to a local field using the lock (lock) as the first. It is because not to avoid a duplication access to the memory and because the application thread-safe could exist. Next we work only with local data field. Further the measurement of maximum and the average value calculations are made, and then the actual rendering of the data sample to the PDA display with the specific color is made. The process stays on the display depicted only for specific time, which is selected as MAX_RECORD – 10. The process stays rendered for 20ms. Then the process is deleted and context of the method Form1.syncObj.Reset (); is handed back to the recTh thread and the whole situation is repeated again.

VI. TESTING THE APPLICATION

The functionality of the application was tested in the Microsoft Visual Studio 2008 and mainly with the help of real devices Fujitsu-Siemens N560. The emulator doesn´t have to always behave just like real devices. It is always better to use your real PDA device for the test, because the hardware is accessed in this application.

Producer	Fujitsu-Siemens
Operating system	Windows Mobile 5; Premium Edition
Processor type	Intel PXA270; XScale
Processor frequency	624 MHz
ROM memory	64 MB
RAM memory	128 MB
Display	3,5 "; TFT; 65536 barev
Display resolution	640 x 480
Memory card slot	ano; SD
Microphone	ano; mono
Reproducing unit	ano; mono
USB	ano; USB 1.1
Wi-Fi	ano
Bluetooth	ano; Bluetooth 1.2
IrDa	ano
GSM	ne
GPS	ano
Number of channels GPS	20
GPS chip	SiRF Star III
Battery	Li-ion; 1200 mAh
Weight	160 g
Dimensions	116 x 71 x 14 mm

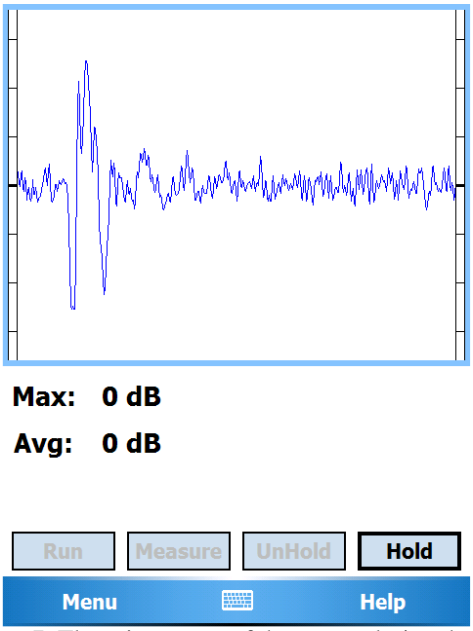
Table 1: Parameters PDA Fujitsu-Siemens N560.

Table 1 shows the parameters of the PDA, where the testing the application PDAScope was performed. The equipment is several years old, and that is why it cannot be compared with today's latest PDA. Very small RAM memory is probably the most limiting parameter.

The first phase of testing consists in the visualization of pre-undefined signal, such as human speech. It was verified with this test that the PDAScope application really somehow visualize the external audio signal. Testing method is illustrated in Picture No 6.

Picture 6: The first phase principles of testing PDAScope application

There is a printscreen of the PDA screen during the testing on the picture below. The signal, depicted in Picture No 7, corresponds to the human voice.

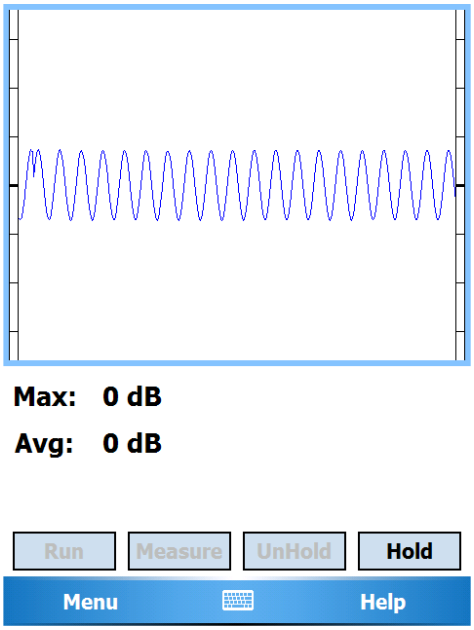


Picture 7: The print screen of the screen during the first phase of testing.

The second phase of testing was aimed at signals testing, about which we know how their process is over time. A sine function is an example. A mobile phone was used for the test. An audio file in MP3 format was released from the mobile phone, which featured the sine wave process in a given frequency. The frequency of 1000Hz was chosen in this case. Mobile phone was used as a signal source (signal generator).

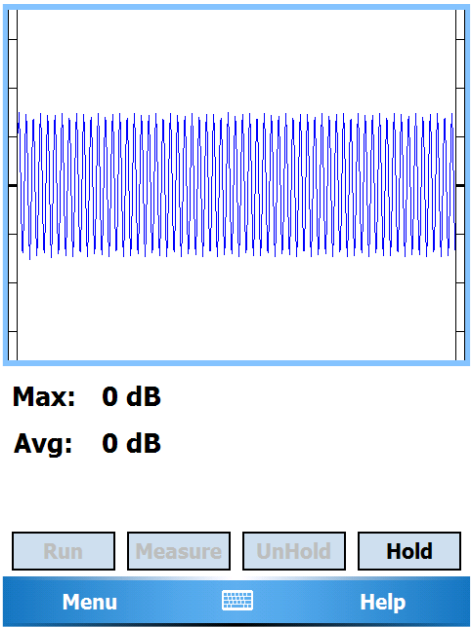
Applications for PDA visualized the sine wave with a small error always on the left edge of the screen, is at the beginning of each scanning cycle. It wasn´t found why such distortion existed all over the entire period of application development and application testing. The process distortion is shown in Picture No 9, which is a sine wave at a frequency of 1000Hz.

Picture 8: The second phase principles of testing PDAScope application.



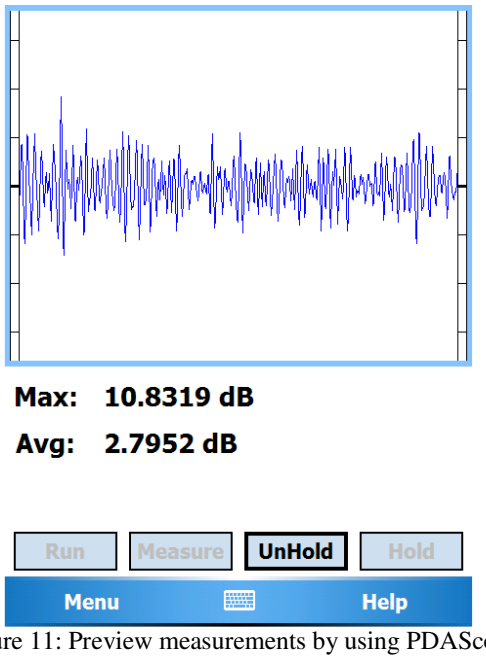
Picture 9: The print screen of the screen during the second phase of testing.

Several other frequencies were tested to be able to find which sinusoidal signal frequency the application is able to accurately visualize. The testing was performed gradually by increasing the frequency by 1000 Hz. Sinusoidal signals were tested in frequencies 1000Hz, 2000Hz, 3000Hz, 4000Hz and 5000Hz. The signal was almost unreadable already at a frequency of 4000Hz. The test follows that the application is able to visualize the signals in the frequency of about 3000Hz. The distortion, described in this charter, is not already identifiable by the visualization of such high frequency. The print screen of the screen, when the sinusoidal signal of frequency 3000Hz is visualized, is shown below in Picture No 10.



Picture 10: Visualization of the signal with a frequency of 3000Hz.

Picture No 11 shows the final test, which consists in a measure testing, is Measure button. The value of maximum and average values of the displayed in decibels are calculated when you press this button.



Picture 11: Preview measurements by using PDAScope.

VII. Conclusions

The application of the digital oscilloscope, that was able to visualize the external audio signal, was realized through this work. The PDAGraph component for the graphs rendering on mobile devices was created together with this application. A benefit of created component is its applicability in other projects.

The application testing found that the maximum frequency, which was an application able to truly visualize, was about 3000Hz. It was used the PDA Fujitsu-Siemens N560 for the testing.

References

[1] Personal Digital Assistant; http://cs.wikipedia.org/wiki/Personal_Digital_Assistant (Date of quotation 02. 05. 2011)

[2] Example of PDA; http://blogs.totalpda.co.uk/2009/06/work-and-play-with-the-fujitsusiemens-pocket-loox-n560/ (Date of quotation 02. 05. 2011)

[3] Recording and Playing Sound with the Waveform Audio Interface; http://msdn.microsoft.com/en-us/library/aa446573.aspx (Date of quotation 03. 05. 2011)

[4] Vzorkování; http://cs.wikipedia.org/wiki/Vzorkov%C3%A1n%C3%AD (Date of quotation 02. 05. 2011)

[5] Example of signal sampling; http://upload.wikimedia.org/wikipedia/commons/a/a0/Vzorkov%C3%A1n%C3%AD.png (Date of quotation 02. 05. 2011)

[6] Mikrofon; http://cs.wikipedia.org/wiki/Mikrofon (Date of quotation 02. 05. 2011)