MEASUREMENT DATA PROCESSING FOR AUDIO SURROUND COMPENSATION

M. Karlsson, W. Kulesza, B. Johansson, A. Rosengren
Department of Technology, University of Kalmar, Sweden
E-mail: magnus.karlsson@hik.se, wlodek.kulesza@hik.se

Abstract: Attractive implementations of DSP tools for measurement can improve students' interests for deeper study of signals theory. The experiment set-up is based on standard equipment that the students can find at home. The application of DSP tools should show them importance of random signal analysis and hardware implementation issues.
Keywords: measurement data, crosscorrelation, matched filter, FPGA

1 INTRODUCTION

The students' interest is of greatest importance to gain understanding of and engagement in any subject. Attractive and real implementation of data processing tools for measurement was an aim of the laboratory development for the courses The Signals and Systems and The System Development with VHDL.

Music and sound processing have been chosen as the theme of the laboratory. In a part of the laboratory, an acoustic signal has been applied to measure a distances between speakers and listener in an acoustic surround system.

Due to different distances between the listener and the speakers, the sound from different speakers is listened with certain relative time delays. The information about the distances can simplify compensation system, which gives opportunity to listen properly the sound from all speakers when the listener is moving around in the room.

2 EXPERIMENT SET-UP

In the one dimensional experiment set-up, we assume that, the system is applied to measure distances among the speakers, the listener and the wall. The listener uses a microphone and that the signal is sent back to the music system via wireless data-communication, e.g. the Blue-Tooth system, Figure 1. The distances can be measured using a test signal, the delay between the test signal, which is sent, and the signal, which is registered by microphone, can be estimated by calculating a cross-correlation between the sent and received signals. For the purpose a matched filter can also be applied. In the two dimensional space, different test signals can be used for each of the speakers or time multiplexing of the same signal can be performed.

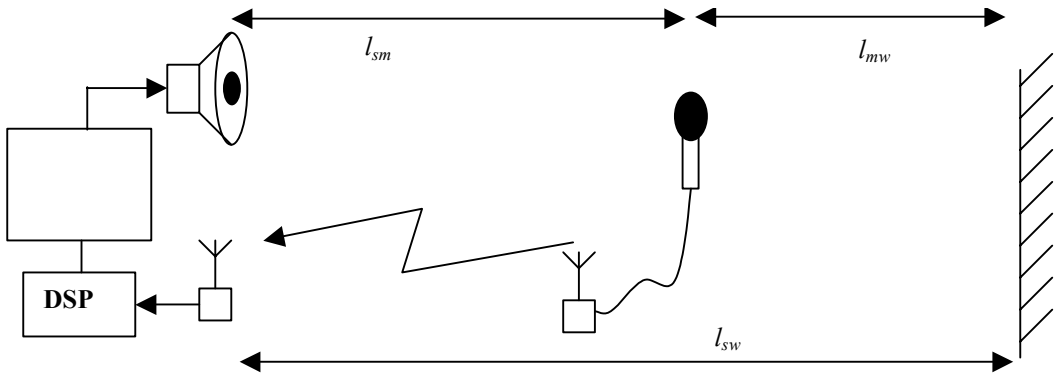

Figure 1. Experiment set-up

Several experiments have been performed to find, which measurement set-up could be the best to estimate distance among speakers, microphone and the wall by measuring sound delay. The test signal, u, which has been sent via the speaker, contains few periods of 5 kHz sinusoidal signal added to a music signal, c_m, which can be treaded as random one. The reason of the real random signal application is to show benefit of using a **c**ross**c**orrelation **f**unction, CCF, and a matched filter, to recover the test signal within a measured signal.

3 USED ALGORITHMS AND ITS COMPUTATION

Two different mathematical algorithms, CCF and matched filter are proposed for the problem solution, [1], [2]. Both of them are very useful to recover a signal contaminated by random noise. The received signal, $y[n]$, contains several components, direct test signal, $u[n]$, and its attenuated and delayed reflections as well as the sent and reflected music signal, $c_m[n]$.

$$y[n_r] = \sum_i \alpha_i \, u[n_t + \beta_i] + c_m[n_r] \tag{1}$$

where α_i is a attenuation constant and β_i is a delay between the transmitting instant, n_t, and receiving instant, n_r of the i-th reflection.

Important assumption of the experiment is that the music signal, $c_m[n]$, and the test signal , $u[n]$, are not correlated.

3.1 Algorithms description

3.1.1 The cross correlation function

The cross correlation function is a measure of similarity in time of two signals. In our case we estimate the instants when the microphone registers the test signal and its reflection.

Handling the CCF as a block-based algorithm with acquisitioned N samples, the unbiased cross correlation function between the test signal, u, and the measured received signal, y, is defined as

$$r_{uy}[m] = \frac{1}{N - |m|} \sum_{n=0}^{N-1} u[n] y[n + m] \tag{2}$$

It is easy to prove using properties of the signal $y[n]$, that the last equation can be rewritten as

$$r_{uy}[m] = \frac{1}{N - |m|} \left\{ \sum_{n=0}^{N-1} \sum_i u[n] \alpha_i u[n + \beta_i + m] + \sum_{n=0}^{N-1} u[n] c_m[n + m] \right\} = \sum_i \alpha_i r_{uu}[m + \beta_i] \tag{3}$$

From properties of the autocorrelation function, it can be concluded that r_{uy} will reach maxima for $m=-\beta_l$, i.e., the instants when the reflections are registered by the microphone.

3.1.2 The matched filter algorithm

Matched filtering is commonly used to detect time recurring signals buried in noise. The main underlying assumption in this method is that the signal is *time limited* and has a known waveshape, what is the case of this experiment. The impulse response of a matched filter is the time-reversed replica of the signal to be detected. The digital matched filter can be represented as an FIR filter and its output, $z[m]$, is defined as the convolution of the impulse response, $h[m]$, and the input signal, $y[m]$.

$$z[m] = h[m] * y[m] = \sum_{n=0}^{M-1} h[n] y[m - n] \tag{4}$$

For the test signal, $u[n]$, of length M samples, the impulse response, $h[m]$, for the causal matched filter, is defined as

$$h[n] = \begin{cases} u[-n + (M-1)] & n \in [0, \ldots, M-1] \\ 0 & elsewhere \end{cases} \tag{5}$$

Hence, the matched filter output yields

$$z[m] = \sum_{n=0}^{M-1} u[-n + (M-1)] y[m - n] \tag{6}$$

using the variable substitution $k = -n + (M-1)$ yields

$$z[m] = \sum_{k=0}^{M-1} u[k] y[m + k - (M-1)] \tag{7}$$

The factor $(M-1)$ is a time shift required to get a causal filter; the variable substitution $m' = m - (M-1)$ compensates the time shift and yields to

$$z(m') = \sum_{k=0}^{M-1} u(k)y(m'+k) \tag{8}$$

In the same way as for CCF algorithm, using properties of the input signal, $y[n]$, it can be proved that the maxima of the signal, $z[m]$, will occur for $m'=-\beta_i$.

3.2 Computation of the algorithms

3.2.1 The CCF properties

Even if the two algorithms, the CCF, and the matched filter are similar, their computation properties are different. The CCF is a block-based algorithm, i.e., blocks of *N* samples are acquisitioned and processed. The algorithm requires a lot of memory, *N* cells, and yields a large latency. The time delay to get the result, is equal to

$$CCF_{Latency} = N \cdot T_{sample} \tag{9}$$

Hence, the CCF is suitable for off-line analysis of the acquisitioned data using a virtual tool implemented in a PC.

3.2.2 The matched filter properties

The matched filter algorithm is a sample-based algorithm, i.e., after each input sample an output sample is produced. Using direct mapping of each operation in the algorithm yields to an implementation with maximum throughput and a minimum amount of memory, [4]. Hence, the used test signal is limited to a short period of time thus an FIR filter of order *M* can be used to implement the matched filter, which requires *M-1* memory cells.

The small arithmetic workload yields no additional delay, since all computations can be performed during one sample interval. Hence, the latency is equal to

$$Matched\ Filter_{Latency} = M \cdot T_{sample} \tag{10}$$

Therefore, the matched filter algorithm is more suitable and easier to use than the CCF algorithm in a hard real time measurement system with on-line analysis implemented in hardware, e.g., **F**ield **P**rogrammable **G**ate **A**rray, FPGA.

3.2.3 Arithmetic workload

In the CCF algorithm, the number of multiplication and additions for each output sample is, *N and N-1*, respectively. However, the test signal, *u,* is only non-zero for *M* samples, what reduce the number of multiplications and additions to, *M*, *M*-1, respectively. The normalization factor can be incorporated in the multiplications since the signal length, N, is known in advance.

In the case of matched filter the number of multiplications and additions are *M* and *M*-1, respectively. Hence, the arithmetic workload for these algorithms is about the same.

However, the arithmetic workload can be significantly reduced by a smart choice of the test signal, *u*. One obvious choice is a sinusoidal test signal with a frequency that is an even fraction of the sampling frequency, i.e.

$$f_{signal} = \frac{1}{T_{sample} R} \quad R \in [2,3,4,\ldots] \tag{11}$$

By exploiting the symmetries in the test signal, *u*, the number of non-trivial multiplications can be reduced significantly. Solutions with only trivial multiplications time ± 1 are to use frequencies equal to the Nyquist frequency, *R*=2, or preferably half of the Nyquist frequency, *R*=4. The large number of adders in the FIR filter's adder tree is handled by using a Carry Save Adder tree, which is faster and requires less hardware than an ordinary adder tree with Carry Ripple Adders, [5].

3.3 Comparison

The matched filter algorithm is similar to the CCF algorithm except for the time shift due to the causal filter and the normalization of the CCF. The lack of this normalization in the matched filter results in a transient of *M*-1 samples, which is not significant due to the short time duration of the test signal in relation to the measured signal.

The matched filter has a much lower latency and requires less memory than the CCF algorithm but the arithmetic workload is about the same.

4 EXPERIMENT DATA

4.1 Experiment part one

During the first experiment, the test signal, *u,* is sent to the speaker simultaneously with a music signal. Therefore, the signal received from the microphone, *y,* contains the music and information originated from the test signal, *u*. The microphone is held at a constant distance from the speaker during the experiment.

From the plots, Figure 2, the delays can be measured. From the left figure it is determined the time instant when test signal was sent. In the right figure the result of the CCF/matched filter is shown. The zero in the time scale is the time instant when the test signal was sent. The delay of the first and second peaks of CCF/matched filter are 60 ms and 140 ms. Therefore the calculated distances are respectively l_{sm}=2.04 m and l_{sw}=3.40 m. Note, that the first peak (direct wave) is lower in amplitude than the second peak which is from a reflected wave. This is due to the fact that the microphone was directed towards the wall.

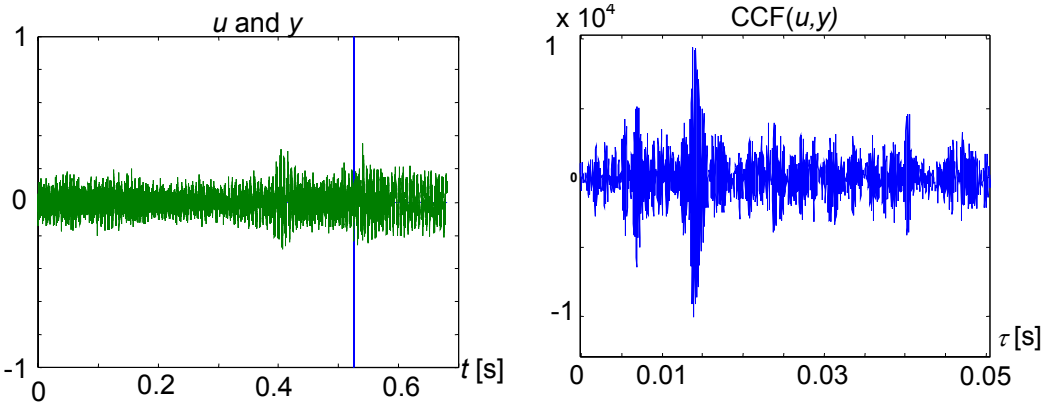

Figure 2. The plots of first experimental data

4.2 Experiment part two

The second experiments test signal, *u,* has the same form like in the experiment part one. It is repeated every 0.5 s. To simplify the task, no music is used. The distance from the microphone to the speaker varies during the experiment. The task is to estimate the position of a listener with sampling rate equal a half-second. The plots, Figure 3, show the result of the experiment.

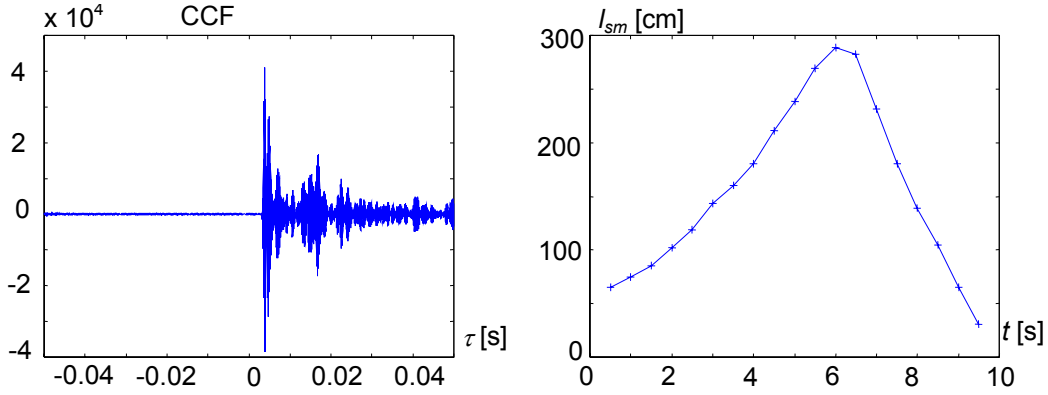

Figure 3. The plots of second experimental data

4.3 Accuracy of distance measurement

The accuracy of the distance measurement depends mainly of the sampling frequency, which is directly correlated with the resolution of the length measurement. If the maximum detection from the CCF is correct the time accuracy is ±1 sample, an estimation of the accuracy using a sample frequency of 48 kHz is hereby.

$$distance\ accuracy = 2T_{sample} \cdot v_{air} = 2\frac{1}{48kHz} \cdot 340m/s = 14mm \qquad (12)$$

The amplitude dynamic range in the computation can easily be made large enough to not interfere with the measurement accuracy. A detail discussion of the measurement accuracy is of great importance but is out of the scope of this paper.

5 ALGORITHMS' IMPLEMENTATION

5.1 Virtual tool implementation

Figure 4, shows the block diagram for the virtual implementation. The measured data are AD converted and sent directly to the Correlation Block, which output is sent to the Level Detector, which detects the time instants when the test signal is recovered in the signal from the microphone. The output of the Correlation Block is visualised using a Scope.

The Level Detection, which is important function of the system. In this case, the used Level Detector has a static threshold, which is set by the user and is also displayed on the Scope. More advanced Level Detectors using filtering and adaptive threshold could be applied in order to increase the robustness of the maximum detection. The logic output from the Level Detector controls the distance estimator, which transforms the time unit into distance unit.

The Test-signal Generator, which signal is sent to the DAC and the Correlation Block, keeps the synchronisation of the measurement. It also generates a control signal *go*, which is used to compensate different delays in the measurement system.

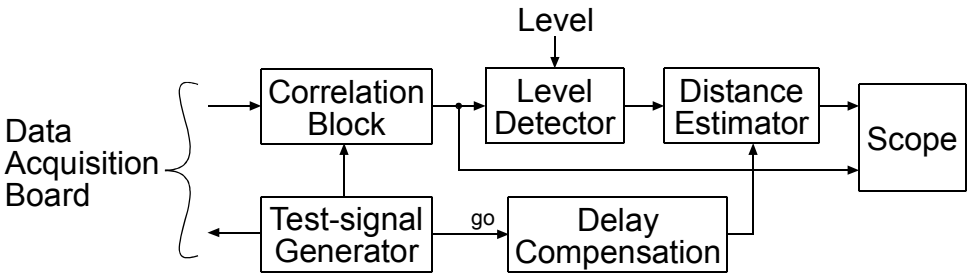

Figure 4. The block diagram of the algorithm for the virtual implementation

For implementation of the virtual tool a software choice is Simulink/Matlab™, [3], using the Data Acquisition Toolbox together with a suitable Data Acquisition Board, which block diagram is shown in Figure 5.

Another common and easy to use and suitable software implementation is LabView™ environment.

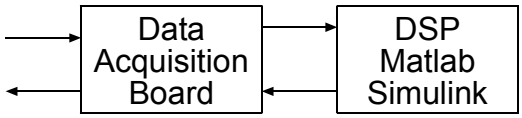

Figure 5. The block diagram of the system

It is easy and commonly to use Matlab for the signal processing part and for the presentation of the data and results. The Simulink environment is easy to adapt to the task and is user friendly.

5.2 Real tool implementation

A hard real time system can easily be implemented in hardware using a **F**ield **P**rogrammable **G**ate **A**rray, FPGA. The block diagram of the implementation is shown in Figure 6. This block diagram is similar to the virtual block diagram in Figure 4, but the Matched Filter replaces the Correlation Block.

The output of the matched filter is visualised by a VGA-monitor, in order to observe the result. The detection threshold is set by the user using a standard keyboard and is also displayed on the VGA-monitor. The position of the listener in the room is displayed using LED displays. The Test-signal Generator keeps the synchronisation of measurement, which signal that is, sent to the DAC. Note, that the test signal is not connected to the matched filter, since the information of the test signal exists in the coefficients of the filter.

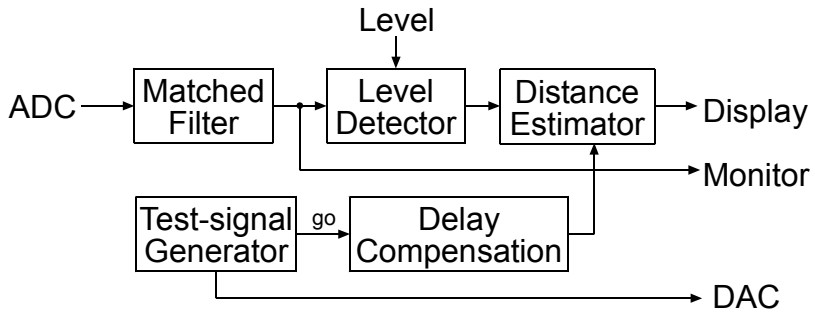

Figure 6. The block diagram of the algorithm for the real implementation

5.2.1 Algorithm design

The design of the algorithm is performed using Simulink/Matlab™. The design of the algorithm is divided into two steps.

Firstly, the DSP block-set, enables the designer to use a large arithmetic precision using Matlab's 64-bits floating-point during the design of the filter and Level Detector. The design of the matched filter is simple using the Filter Design Tool, which produces models of the filter using several different structures.

Secondly, is the design of the algorithm using limited accuracy with fixed-point arithmetic. The trade-off between accuracy and amount of hardware is simple using the Filter Design Tool together with the Fixed-point block-set. The data and coefficient word lengths can be easily adjusted and compared with the high accuracy model during simultaneous simulation of the models.

5.2.2 Algorithm implementation

The modelling and construction of the hardware is performed in several steps shown in Figure 7, [4]. Firstly, the algorithm is manually mapped on a suitable architecture, which is described using a **H**ardware **D**escription **L**anguage HDL, in our case VHDL. In view of the fact that the system is heterogeneous and the number of multiplications is small, the direct mapping method is used, [4]. We are using the VHDL design tool, FPGA Advantage™ from Mentor Graphics, [6]. Each synthesis step in the design flow is verified through simulation and comparison of the different models, this is important in order to avoid mistakes during the design.

The next synthesis step is the choice of arithmetic, e.g., bit-parallel, digit-serial or bit-serial arithmetic can be used. With the aim of simplify the design and control the bit-parallel approach has been chosen at the expense of larger area of the final implementation.

The next synthesis step to the **R**egister **T**ransfer **L**evel, RTL, is also manual. The description of the hardware is now at the level of pipelined data-paths, i.e. HDL code describing Boolean functions and registers.

The next stage is the choice of hardware technology, in our case the FPGA was chosen. The HDL code at the RTL level is automatically synthesised using a logic synthesis tool, Leonardo Spectrum, which is included in the FPGA Advantage Package. The output from the tool is a netlist of logic gates and flip-flops, which can be simulated for control of logic function using the HDL simulator. It also performs a first coarse static timing estimation of the logic delays. This timing estimate is used as a guide if a redesign is required due to that the delays are not fulfilling the specifications or is to close to them. The logic synthesis tool is using four different heuristics methods for the synthesis and optimisation of area and/or delay of the logic nets.

The final step in the design flow is placement of the logic gates and routing between them. The Place & Route tool performs this. This tool is technology dependent; we are using FPGA circuits from Xilinx Inc, and the Place & route tool, Xilinx Alliance Series, [7]. After placement and routing a better static delay estimation can be performed, since the delay due to wires is known. Finally, programming of the FPGA and the first test run can be carried out. Using the development boards from Xess Corporation, [7], which is easily programmed using the PC parallel port. The board also has a Codec circuit for HiFi, which contain both the ADC and a DAC and is well suited for our application.

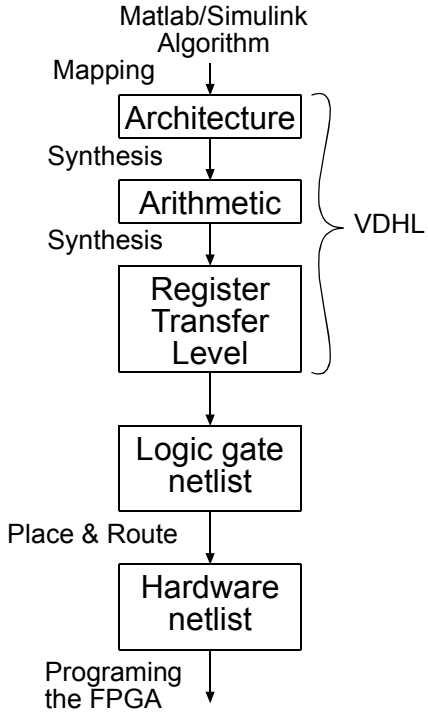

Figure 7. The VHDL/FPGA design flow

6 CONCLUSIONS

To mix different tools, real and virtual, for measurement data processing, which is an important issue of the engineering education is one of the laboratory aims. The application of signal processing tools using music and sound signals as a theme in the laboratory gains the students' interest and engagement in performance of the laboratory tasks.

Real and virtual implementations of measurement instruments make the laboratory exciting for students from different education programs such as computer science as well as electronics.

Implementation of attractive and fun applications in hardware using VHDL is of great significance to gain students' interest in the subject.

The integration between two different courses is also an important issue; since this gives the students better understanding of theory and practice of the subjects.

The proposed experiment shows that quite sophisticated measurement can be performed using standard tools what gives opportunity to perform the experiment even at home.

The accuracy and robustness of the measurement should be improved using more advanced test signal and level detector. The minimal level of the test signal is one of interesting issues of the experiment.

REFERENCES

[1] J. McGhee, W. Kulesza, I.A. Henderson, M.J. Korczynski, *Measurement Data Handling*, ACGN Lodart, Lodz, 2001.
[2] E.C. Ifeachor, B.W. Jervis, *Digital Signal Processing - A Practical Approach*, Addison-Wesley, 1996.
[3] Matlab Tools Manuals, MathWorks, www.mathworks.com
[4] L. Wanhammar, *DSP Integrated Circuits*, Academic Press, Feb. 1999.
[5] M. Karlsson, A generalized carry-save adder array for digital signal processing, IEEE conf., NORSIG-2000, Kolmården, Sweden, June 13-15, 2000.
[6] FPGA Advantage, Mentor Graphics, www.fpga-advantage.com
[7] Xilinx Inc. Alliance series, www.xilinx.com
[8] Xess Corporation, www.xess.com