Reconfigurable Acceleration of Microphone Array Algorithms for Speech Enhancement

Ka Fai Cedric Yiu[1], Chun Hok Ho[2], Nedelko Grbric[3], Yao Lu[1], Xiaoxiang Shi[1] and Wayne Luk[2]

[1]The Hong Kong Polytechnic University, Department of Applied Mathematics Kowloon, Hong Kong, China

[2] Imperial College London, Department of Computing, London, England

[3] Blekinge Institute of Technology, School of Engineering 37225 Ronneby, Sweden

[1]macyiu@polyu.edu.hk , [2]cho@doc.ic.ac.uk, [3]ngr@bth.se

Abstract

Microphone arrays play an important role in noise reduction and speech enhancement. Their algorithms are based on beamforming, which reduces the level of localized and ambient noise signals while minimizing distortion to speech from the desired direction via spatial filtering. This paper describes a class of subband beamforming algorithms. The similarity between different algorithms is discussed. To enhance computational efficiency, the algorithms are implemented in frequency domain. A hardware architecture, with bitwidth optimization, is proposed to support the algorithms. An implementation with 7 instances on a Xilinx XC4VSX55 FPGA at 175MHz can run 41.7 times faster than the corresponding pure software implementation on a 3.2GHz Pentium 4 PC.

1 Introduction

Voice input systems with various functionality have become a part of our daily life. Apart from mobile communication devices, there are a wide range of other applications including teleconferencing, voice over IP, speech recognition devices and voice telematic system in cars. For such applications, it is crucial to have a good acoustic interface in order to provide accurate voice control or smooth hands-free audio communication. An essential technique to enhance the received signal is to employ a microphone array so that beamforming and noise-cancelling techniques can be applied. For instance, Philips has begun to employ a two-microphone system in their new mobile phone design for improved noise reduction [1].

Beamforming techniques exploit fundamental properties about the spatial and/or temporal distribution of both the speech and noise sources, in order to enhance perception [2]. If the geometry of the problem is known, there are several ways of designing a beamformer. One way to adapt the beamformer for specific applications is to use sequences of calibration signals [3]. The algorithm can be efficient if subband processing is employed.

There are several studies about the implementation of beamformer on reconfigured devices. A time-delay sonar beamforming has been reported [7]. The beamformer can achieve six times speed up over commodity DSP systems. Another beamformer implementation involves delta-sigma modulation, and the beamformer is applied to medical ultrasonic application [8]. However, these studies do not consider subband processing and have not been applied to acoustic applications.

While beamformers can be implemented as an application-specific integrated circuit (ASIC), it would be difficult to adapt such device to different environments while maintaining the quality of the output. For instance, in an environment with mild noise, a rather short filter might be sufficient to suppress it. On the other hand, in a very noisy industrial environment, very long filters would be required to obtain reasonable speech enhancement. As a result, filter length should be varied to adapt to the change of environment. An ASIC implementation may fail to operate under this environment unless the filter length can be reconfigured in someway. A field programmable gate array (FPGA) implementation, however, can be dynamically reconfigured to increase the filter length while reducing the number of filter instances. In addition, a speech recognizer can be integrated into the FPGA to form an system-on-a-chip (SoC) solution. Such SoC platform can be further configured to recognize different application-specific voice commands.

This paper covers optimal methods to find beamformers based on sequences of calibration signals. This includes an efficient frequency domain modified recursive least squares adaptive algorithm (CWRLS) [4], and the maximization of signal-to-noise ratio beamforming algorithm. The algorithms share a common structure and have a lot of simi-

1-4244-1898-5/08/$20.00 ©2008 IEEE

larity in the implementation. It includes an adaptive frequency domain structure consists of a multichannel analysis filter-bank and a set of adaptive filters, each adapting on the multichannel subband signals. The output of the beamformers are reconstructed by a synthesis filter-bank in order to create a time domain output signal. Information about the speech location is put into the algorithm by a recording performed in a low noise situation, simply by putting correlation estimates of the source signal into a memory. The recording only needs to be done initially or whenever the location of interest is changed. The adaptive algorithm is then run continuously and the reconstructed output signal is the extracted speech signal.

In order to achieve real-time performance while maintaining better flexibility in different environments, the implementation of the algorithms on a high-end FPGA is studied. The algorithm CWRLS is used as an illustration of the hardware design. First, to achieve computational efficiency, first of all, a frequency domain implementation (or subband processing) is employed to speed up the convergence of the beamformer. Then, the complete architecture is simulated in hardware to aim for real-time operation of the final beamformer. To summarize, the key contributions of this paper include:

1. The first FPGA-based hardware architecture for a class of microphone array algorithms based on the use of calibrated signals together with subband processing. The proposed design can efficiently reduce noise and enhance speech quality in a time critical environment.
2. Optimization based on bitwidth analysis to explore suitable bitwidth of the system. The optimized integer and fraction size using fixed point arithmetic can reduce the overall circuit size by up to 80% when compared with a direct realization of the software onto an FPGA platform.
3. An evaluation of our approach, including a comparison with a software version running on a 3.2GHz Pentium 4 machine, showing that the FPGA-based implementation at 175MHz with 7 instances can achieve speedup of 41.7 times.

2 Background

The source is assumed to be a wideband source, as in the case of a speech signal, located in the near field of a uniform linear array of I microphones. The beamformer uses finite length digital linear filters at each microphone. The output of the beamformer is given by,

$$y[n] = \sum_{i=1}^{I} \sum_{j=0}^{L-1} w_i[j]\, x_i[n-j] \tag{1}$$

where $L-1$ is the order of the FIR filters and $w_i[j]$, $j = 0, 1, \cdots, L-1$, are the FIR filter taps for channel number i. The signals, $x_i[n]$, are digitally sampled microphone observations and the beamformer output signal is denoted $y[n]$.

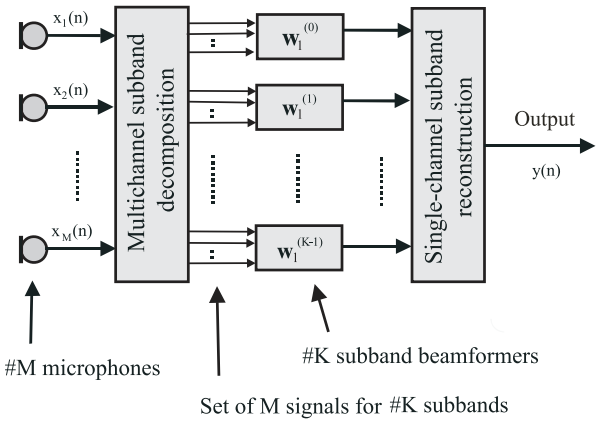


Figure 1: Subband beamforming structure. The number of subbands is K and the number of microphones is M.

These FIR filters need to be high order to capture the essential information, especially if they also need to perform room reverberation suppression. By using a subband beamforming scheme, the computational burden will become substantially lower. The subband beamforming scheme used in this study is presented in Figure 1. In this case each microphone signal is filtered through a subband filter. A digital filter with the same impulse response is used for all channels, thus all spatial characteristics are kept. This means that the large filtering problem is divided into a number of smaller problems.

The signal model can also be described in the frequency domain[1], and the filtering operations become multiplications with number I complex frequency domain representation weights, $w_i^{(f)}$. For a specific frequency, f, the output is given by

$$y^{(f)}[n] = \sum_{i=1}^{I} w_i^{(f)} x_i^{(f)}[n] \tag{2}$$

where the signals, $x_i^{(f)}[n]$ and $y^{(f)}[n]$, are time domain signals as specified before but they are narrow band, containing essentially components with frequency f. The observed microphone signals are given in the same way as

$$x_i^{(f)}[n] = s_i^{(f)}[n] + \sum_{d=1}^{D} \nu_{id}^{(f)}[n] + v_i^{(f)}[n] \tag{3}$$

and the optimization objective will be simplified, due to the linear and multiplicative property of the frequency domain

[1]The representation is made on a finite grid that can be dense. This operation can be an FFT or a filter-bank.

representation, as

$$\forall f \begin{cases} max \left[y^{(f)}[n] \cong s^{(f)}[n] \right] \\ min \left|\left| \sum_{i=1}^{I} w_i^{(f)} \left[\sum_{d=1}^{D} \nu_{id}^{(f)}[n] + v_i^{(f)}[n] \right] \right|\right|. \end{cases} \quad (4)$$

If a least-squares criterion is used, the objective is formulated in the frequency domain as a least squares solution defined for a data set of N samples. However, since the reference source signal information is not directly available, a calibration sequence gathered in a quiet environment is used instead. This calibration signal will represent the temporal and spatial information about the source.

The objective is to calculate

$$\mathbf{w}_{ls,opt}^{(k)}(N) = \left[\hat{\mathbf{R}}_{ss}^{(k)}(N) + \hat{\mathbf{R}}_{xx}^{(k)}(N) \right]^{-1} \hat{\mathbf{r}}_s^{(k)}(N) \quad (5)$$

where the real frequency $f = F_s k/K$, with F_s the sampling frequency and K the total number of subbands, and where the array weight vector, $\mathbf{w}_{opt}^{(k)}$ for the subband k is defined as

$$\mathbf{w}_{opt}^{(k)} = [w_1^{(k)}, \quad w_2^{(k)}, \quad \dots \quad w_I^{(k)}]^T. \quad (6)$$

The source correlation estimates can be pre-calculated in the calibration phase as

$$\hat{\mathbf{R}}_{ss}^{(k)}(N) = \frac{1}{N} \sum_{n=0}^{N-1} \mathbf{s}^{(k)}[n] \mathbf{s}^{(k)\,H}[n] \quad (7)$$

$$\hat{\mathbf{r}}_s^{(k)}(N) = \frac{1}{N} \sum_{n=0}^{N-1} \mathbf{s}^{(k)}[n] s_r^{(k)\,*}[n] \quad (8)$$

where

$$\mathbf{s}^{(k)}[n] = [s_1^{(k)}[n], \quad s_2^{(k)}[n], \quad \dots \quad s_I^{(k)}[n]]^T$$

are microphone observations when the calibration source signal is active alone, while the observed data correlation matrix estimate $\hat{\mathbf{R}}_{xx}^{(k)}(N)$ can be calculated similarly.

3 The CWRLS Algorithm

We assume that the estimated correlation matrix, $\hat{\mathbf{R}}_{ss}^{(k)}$, and the estimated source signal cross correlation vector, $\hat{\mathbf{r}}_s^{(k)}$, are made available from an initial acquisition, for each subband. Additional disturbing and known sources with fixed location, both point sources and others, are assumed to be available during the acquisition phase. The source cross correlation estimates must be acquired when only the source signal of interest is active. For each subband signal, $k = 0, 1, \cdots, K-1$, where for each subband the corresponding normalized frequency is $f = 2\pi k/K$ and each sample instant n, the observed microphone signals in subband number k are denoted $x_i^{(k)}[n]$, $i = 1, 2, \cdots, I$. The number of available samples in the acquisition phase is N.

The algorithm can be summarized as follows:
Initialize $\mathbf{P}_0^{(k)}$ as the inverse of the correlation matrix.
For $n = 1, 2, \cdots$

$$\begin{aligned} \mathbf{x}_n^{(k)} &= [x_1^{(k)}[n], \quad x_2^{(k)}[n], \quad \dots \quad x_I^{(k)}[n]]^T \\ \mathbf{P}^{(k)} &= \lambda^{-1} \mathbf{P}_{n-1}^{(k)} - \frac{\lambda^{-2} \mathbf{P}_{n-1}^{(k)} \mathbf{x}_n^{(k)} \mathbf{x}_n^{(k)\,H} \mathbf{P}_{n-1}^{(k)}}{1 + \lambda^{-1} \mathbf{x}_n^{(k)\,H} \mathbf{P}_{n-1}^{(k)} \mathbf{x}_n^{(k)}} \\ \mathbf{P}_n^{(k)} &= \mathbf{P}^{(k)} - \frac{\gamma_p (1-\lambda) \mathbf{P}^{(k)} \mathbf{q}_p^{(k)} \mathbf{q}_p^{(k)\,H} \mathbf{P}^{(k)}}{1 + \gamma_p (1-\lambda) \mathbf{q}_p^{(k)\,H} \mathbf{P}^{(k)} \mathbf{q}_p^{(k)}} \end{aligned} \quad (9)$$

where index $p = (n \, mod \, I) + 1$,

$$\mathbf{w}_n^{(k)} = \alpha \mathbf{w}_{n-1}^{(k)} + (1-\alpha) \mathbf{P}_n^{(k)} \hat{\mathbf{r}}_s^{(k)}$$

The output from each subband is then:

$$y^{(k)}[n] = \mathbf{w}_n^{(k)\,H} \mathbf{x}_n^{(k)}$$

n is increased by one, return.

The operation phase consists of continuous decomposition of the microphone signals into discretized frequencies, by the subband decomposition structure. The subband weights are updated by making use of both the memorized correlation estimates and the actual microphone observations. The output from each subband signal is reconstructed with the reconstruction filter-bank, and the time domain output consists of the estimate of the sound source of interest. The algorithm is adapting continuously once the correlation estimates are placed into memory.

The algorithm contains a step where a rank one update of the correlation matrix is performed using scaled eigenvectors, one eigenvector for each new input data vector. This step adds correlation estimates from the source signal and in this way, the information gathered in the acquisition phase, will remain as a constant part of the correlation matrix while the contributions from the environmental noise will be subject to the forgetting factor in the estimates. This procedure forces full rank properties into the matrix inverse, independent of actual data properties.

4 Hardware Architecture and Design

The optimal beamformer algorithm described in Section 3 can be implemented under ergodic signal property assumptions, where the expectation operator may be interchanged with time averaging estimations. Further, the source signal and the noise/interference- signal components have to be accessible separately so that we may estimate the source- and the noise/interference correlation estimates individually. In the following it is assumed that these constraints are fulfilled.

In the time domain, the main operations of the CWRLS beamformer involves those given by equations (5) and (9).

These operations are greatly reduced by carrying out the actual filtering in the frequency domain, and transforming the results back to the time domain described using the dataflow shown in Figure 2. A hardware structure is designed to support the calculations. The design contains two main processing cores: the complex matrix-vector product core and FFT/IFFT core. The FFT core transforms the input signal from the time domain to the frequency domain, and the IFFT core transforms the signal back to time-domain. The multiplication core performs the filtering in the frequency domain. The proposed hardware design performs the following calculations:

1. Analyse the input signal to their frequency domain representations via FFT;
2. Filter the subband signals by the subband impulse response estimates. The multiplication itself is a complex matrix and vector product circuit;
3. Synthesize the impulse response estimates back to the time domain via IFFT (inverse FFT).

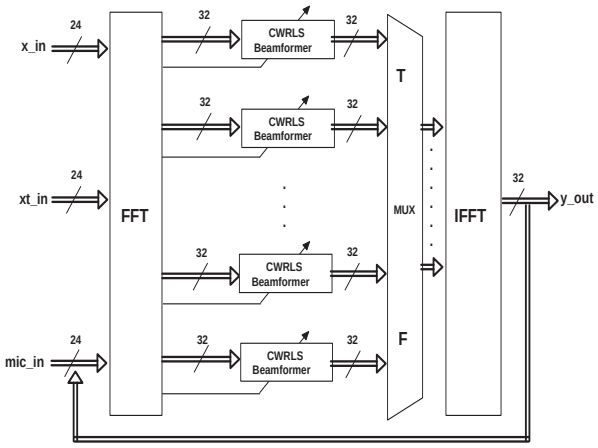


Figure 2: Dataflow of the main operations.

Once the full-band impulse response estimate is reconstructed, the output signal can easily be calculated by software implementation.

During profiling, time-consuming operations can be determined for implementation in hardware using FPGAs. A shared memory architecture is adopted to enable efficient communication and exchange of a large chunk of data between the CPU and the hardware accelerator.

In contrast to software development, hardware development provides an opportunity for bitwidth optimization: varying the width of processing and storage units in the design to achieve the best trade-offs in size, speed and output numerical properties, such as the signal-to-noise ratio.

Our bitwidth optimization is based on fixed-point representation with saturation arithmetic to handle overflow conditions [5]. In saturation arithmetic, if the result of an operation is greater than a given maximum, it is set to that maximum. A set of fixed-point library is developed in software where the size of integer and fraction can be adjusted. This library allows exploration of how bitwidth affects the quality of the signal during beamforming.

Bitwidth analysis is used to identify a near-optimal bitwidth for the hardware to ensure the quality of the result while reducing the area requirement. Two steps are involved, dealing respectively with the integer width and the fraction width in the fixed-point representation. First, dynamic range analysis techniques are used to find the integer width. From an initial estimate and a representative collection of input values, the integer width of the arithmetic operations in the design is varied systematically to find the smallest bitwidth such that the result remains valid. Speech distortion and noise suppression performance measures are used to quantify the difference in performance for different integer sizes. Second, a similar procedure is adopted in precision analysis to determine the fraction width. As the fraction width decides the accuracy of the result, it can be determined by comparing the output speech signal with double-precision floating-point arithmetic.

The internal hardware architecture is described in Figure 3, where it is depicted at logic block level. The core contains an operation unit, a Direct Memory Access (DMA) read/write master pair and an On-chip Peripheral Bus (OPB) register slave. In order to maximize system performance, the FFT/IFFT and the complex multiplier are implemented using the core generator provided by the vendor tools. The FFT/IFFT component used in this project is a Xilinx '128-point Pipelined' 24-bit FFT, that allows continuous data processing and achieves best transform time. The complex multiplier has been configured to support 32-bit input and 64-bit output. The final result has to be truncated to 32-bit so that it can be placed onto the On-chip Memory Bus (OCM). Scaling and saturation mechanism are also implemented to avoid overflow happening. In addition to the FFT/IFFT and complex multiplier cores, four different configurations of FIFOs are also generated by Xilinx core generator as I/O buffers. By taking advantage of Xilinx IP cores, project development time can be greatly reduced and the quality of the design can be guaranteed. The architecture has been implemented on an FPGA platform using VHDL. It is synthesized using Synplify Pro 8.1, placed and routed on the Xilinx XC4VSX55-12-FF1148 [6] FPGA devices using Xilinx ISE 9.1i FPGA design package.

Multiple instances of our CWRLS beamformer can be packed in a single FPGA to boost the performance, which would be useful especially when the design has multiple channels. This technique can fully utilize the resource on the FPGA to gain massive speedup.

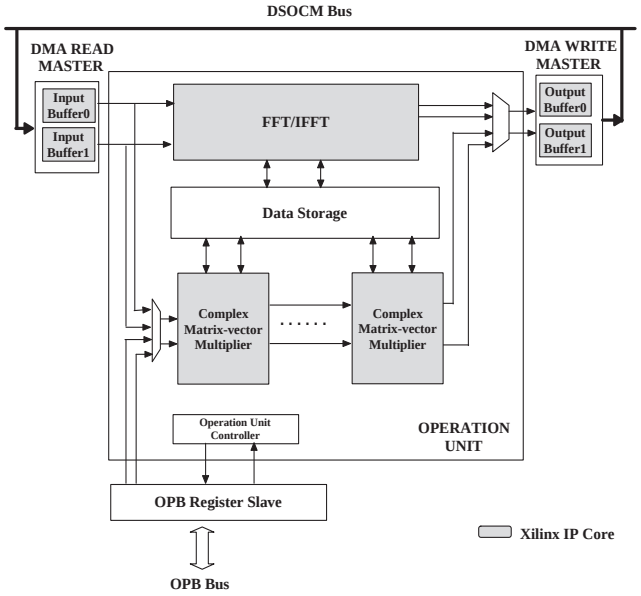


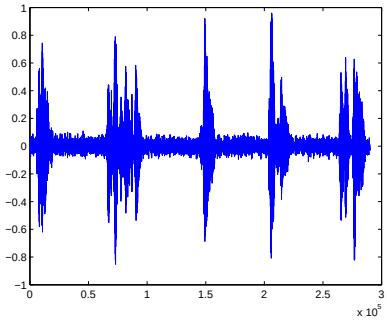
Figure 3: Block diagram of the hardware architecture for CWRLS beamformer.

5 Results

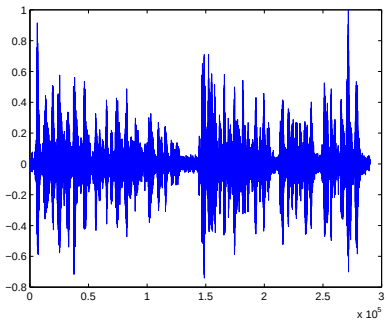
Our approach is evaluated in a simulation experiment with four microphones, where the total number of subbands $M = 128$ with a decimation factor $D = 64$. Figure 4 illustrates the inputs and outputs of the CWRLS beamformer. Given an input speech signal and a noise signal as shown respectively in Figure 4a and 4b where the noise signal is introduced from sample 1 to 290,000, the CWRLS beamformer produces a filtered signal with reduced noise.

The bitwidth analysis method in Section 4 is used to determine the width of integer and fraction. First, the optimal integer width is found to be 12 bits; a further increase does not improve the results significantly. Since overflow may occur occasionally, saturation arithmetic helps to minimize the impact. Second, the optimal fraction width is found to be 20 bits without unacceptably distorting the output speech signal. Figure 4c and 4d show the filtered signals produced by the CWRLS beamformer using double-precision floating-point arithmetic and the 32-bit fixed-point arithmetic that we use, illustrating their similarity. Moreover, in our current simulation the amplitude of the noise and the speech signals are designed to be similar. In reality, the noise would be much weaker than the speech, so the CWRLS beamformer can perform even better.

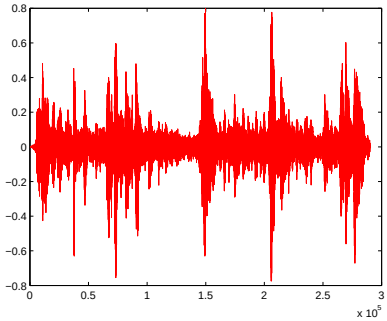
The proposed beamformer architecture is implemented on Xilinx Virtex 4 device. The design is described in VHDL and Xilinx ISE 9.1i is used in the design flow. The implementation results of single beamformer core are shown in Table 1. An estimation of the area usage of a direct single precision floating point implementation is about 29,000



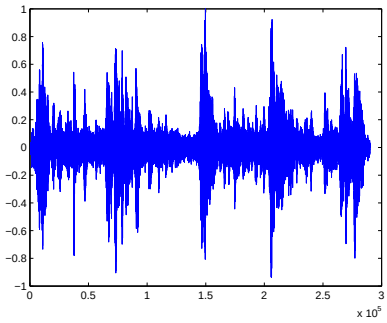
(a) input speech signal



(b) input noise signal



(c) filtered signal using double precision floating-point representation for CWRLS



(d) filtered signal using fixed-point representation for CWRLS, integer size = 12, fraction size = 20

Figure 4: Input signal and results of the CWRLS beamformer.

slices. So an optimized fixed point implementation can reduce the area requirement by 80%. In addition, fixed point implementation provides better clock cycle time and reduce the total number of clock cycle required for the beamforming operations which improves the performance of the beamformer.

An estimation has been made to evaluate the performance of the FPGA-based CWRLS beamformer. Assuming one block of data contains 256 samples under a 16kHz sampling rate, the number of clock cycles required for processing the block of data in the frequency domain is measured to be 1,031,774. Therefore, given that the period of one clock cycle is $1/(175MHz) = 5.71ns$, the FPGA-based CWRLS beamformer can perform one step of speech enhancement in $5.89ms$, or 43,463 samples per second.

Software version is developed in ANSI C and compiled to native machine code using the Linux compiler GCC. It should be noted that the algorithm compiled using GCC has the optimization feature that is particularly useful with vector and matrix computations, which is used intensively in the CWRLS beamformer. A test is performed by providing 290,000 samples to the program and measuring the time required to finish all the calculations. The test is performed on a Pentium 4 3.2GHz machine with 1GB memory, and it takes an average of 36.6 seconds to finish the calculations. Therefore, the software performance is 290,000/36.6 = 7,293 samples per second. It shows that the FPGA-based CWRLS beamformer can achieve 6.0 times speedup even with one instance of subband calculation, when compared with software running on a 3.2GHz PC.

Table 2 summarizes the results when adding more instances of the filter in an XC4VSX55-12-FF1148 FPGA chip and shows how the number of CWRLS beamformer instances affects the speedup. While it is expected the frequency is deteriorated when more cores are instantiated, we found that the frequency is not decreased even if the utilization rate is 84%. So the speedup would scale linearly with the number of CWRLS beamformer instances. An XC4VSX55-12-FF1148 device can pack at most 7 instances of the CWRLS beamformer, so the speedup will be 41.7 times.

FPGA device	XC4VSX55-12
Slices used	5937 (12%)
DSP48 used	72 (14%)
Block RAM used	8 (2%)
Frequency (MHz)	175

Table 1: Summary of the CWRLS beamformer.

Instances	Frequency (MHz)	Slices	DSP	Speedup
1	175	12%	14%	6.0
3	175	36%	42%	17.9
5	175	60%	70%	29.8
7	175	84%	98%	41.7

Table 2: Multiple instances of the CWRLS beamformer on an XC4VSX55-12-FF1148 FPGA.

6 Conclusions

A class of microphone array algorithms are studied, which involve designing beamforming filter weights by using a sequence of calibrated signals. In particular, a typical implementation, namely the subband calibrated weighted recursive least squares algorithm, is studied in detail. A hardware implementation of the algorithm on a high-end FPGA is described. The complete architecture is realized in hardware and results show that an overall improvement of 41.7 times over software on a 3.2GHz Pentium 4 Processor can be achieved, when an FPGA-based coprocessor performs the critical part of the algorithm.

Further work includes dynamically reconfiguration of filter length by using an embedded speech recognizer as a performance indicator. Also, optimizations for reducing power and energy consumption, and extensions to exploit the reconfigurability of FPGAs to support run-time customization for adaptive beamforming, are interesting topics for future research.

References

[1] NXP, "Philips new LifeVibes family of acoustical solutions improves the quality of mobile communications," *http://www.nxp.com/news/content/file_943.html*, March, 2003.

[2] B.D. Van Veen and K.M. Buckley, "Beamforming: a versatile approach to spatial filtering," *IEEE ASSP Magazine*, 5(2):4-24, April 1988.

[3] S. Nordholm, I. Claesson and M. Dahl, "Adaptive microphone array employing calibration signals: An analytical evaluation," *IEEE Trans. Speech Audio Processing*, 7:241-252, 1999.

[4] N. Grbić and S. Nordholm, "Soft constrained subband beamforming for hands-free speech enhancement," *ICASSP-02*, I-885-888.

[5] G.A. Constantinides, P.Y.K. Cheung, and W. Luk, "Synthesis of saturation arithmetic architectures," *ACM Transactions on Design Automation of Electronic Systems*, 8(3):334–354, 2003.

[6] Xilinx Inc. Virtex-4 Family Overview. *http://direct.xilinx.com/bvdocs/publications/ds112.pdf*, 2004.

[7] Paul Graham and Brent Nelson, "FPGA-Based Sonar Processing," *Proc of Field Programmable Gate Arrays*, pp. 201–208, 1998.

[8] B. G. Tomov and J. A. Jensen, "A new architecture for a single-chip multichannel beamformer basedon a standard FPGA," *IEEE Ultrasonics Symposium*, 2:1529-1533, 2001.