FPGA Implementation of Dual-Microphone Delay-and-Sum Beamforming for In-Car Speech Enhancement and Recognition

Harvey Ye, Jim Whittington

Department of Electronic Engineering
La Trobe University
Melbourne, Victoria, Australia
{h.ye, j.whittington}@latrobe.edu.au

Ivan Himawan, Tristan Kleinschmidt, Michael Mason

Speech and Audio Research Laboratory
Queensland University of Technology
Brisbane, QLD, Australia
{i.himawan, t.kleinschmidt, m.mason}@qut.edu.au

***Abstract*—In an automotive environment, the performance of a speech recognition system is affected by environmental noise if the speech signal is acquired directly from a microphone. Speech enhancement techniques are therefore necessary to improve the speech recognition performance. In this paper, a field-programmable gate array (FPGA) implementation of dual-microphone delay-and-sum beamforming (DASB) for speech enhancement is presented. As the first step towards a cost-effective solution, the implementation described in this paper uses a relatively high-end FPGA device to facilitate the verification of various design strategies and parameters. Experimental results show that the proposed design can produce output waveforms close to those generated by a theoretical (floating-point) model with modest usage of FPGA resources. Speech recognition experiments are also conducted on enhanced in-car speech waveforms produced by the FPGA in order to compare recognition performance with the floating-point representation running on a PC.**

***Index Terms*—Field programmable gate arrays, array signal processing, speech enhancement, speech recognition.**

I. Introduction

The operation of various in-car devices such as mobile phones, navigation systems, entertainment systems and climate controls are known sources of driver distraction, which increase the potential of accidents while driving. In various countries, it is illegal to operate mobile phone handsets whilst driving and only the use of hands-free telephones are permitted. The desire to develop hands-free operation of all these devices has lead to an interest in using speech as a natural and less distracting means of interacting with the car.

In the automobile environment interfering noises generated by things such as the car's engine, road conditions and other traffic are a major impediment to acquiring high quality speech signals. This situation makes the operation of hands-free telephones less comfortable as the noise superimposed with the speech makes the conversation difficult to hear. Devices which rely on using speech recognition for control also suffer due to the generally poor quality of speech acquired directly. Speech enhancement techniques which reduce the levels of noise in the signal are beneficial for both in-car telephony and devices that require speech recognition for control.

Speech enhancement techniques can generally be broken into two classes based on the number of microphone signals considered. Single channel techniques utilise the signal from only one microphone and have historically included a variety of algorithms, some of the most common being adaptive filter techniques (Lim & Oppenheim 1979) and spectral subtraction techniques (Boll 1979), (Berouti *et al.* 1979), (Ephraim & Malah 1984). Some of the limitations associated with single channel techniques include the introduction of musical noise and increased signal distortion as signal-to-noise ratios (SNR) decrease.

Multi-channel speech enhancement techniques refer to algorithms which combine acoustic signals from two or more microphones to perform spatial filtering. One benefit of multi-channel enhancement techniques is the ability to adjust or steer the microphone array (beam), using signal processing, in order to focus the signal acquisition on a specific location where the target source is. Multi-channel techniques are also expected to offer the ability to enhance signals at lower SNR due to the inclusion of multiple independent transducers (Johnson & Dudgeon 1993). These benefits are offset by the obvious increased costs involved in having multiple microphones and the processing required for these additional signals. Two-microphone speech enhancement has started to gain momentum as it provides some of the benefits of multi-microphone

This work was supported by the Australian Cooperative Research Centre for Advanced Automotive Technology (AutoCRC).

methods, while not being too costly to implement (Aarabi & Shi 2004), (Ahn & Ko 2005), (Beh *et al.* 2006).

To realise a speech processing system in an automotive environment, an appropriate hardware platform is required. Cost is a key factor in the highly competitive automotive environment, yet speech processing techniques such as delay-and-sum beamforming (DASB) require some digital signal processing (DSP) capability. Thus, the hardware solution must be cost effective while still providing relatively high-performance DSP.

FPGA based signal processing can significantly outperform equivalent DSP processor solutions (Halupka *et al.* 2007), (Bagni & Zoratti 2007). As demonstrated by Halupka *et al.* (2007) in their implementation of a phase-based speech enhancement technique, a fixed-point DSP processor may not be capable of providing sufficient real-time processing power. Research done in conjunction with this work has successfully implemented the single channel spectral subtraction method on FPGA (Whittington *et al.* 2008).

Xilinx, a leading FPGA vendor, has developed the Xilinx Automotive (XA) product family specifically for automotive applications. The Xilinx XA Spartan-3A DSP FPGA is a lower cost member of the Xilinx XtremeDSP™ device portfolio, which includes larger and higher performance FPGAs, such as the Virtex-4 SX. With well over one million system gates, plus memory and XtremeDSP™ slices, the Xilinx XA Spartan-3A DSP FPGAs are the ultimate target for our work (Kitagawa 2008), (Xilinc Inc 2007).

The design is initially developed in a high-end device (i.e. Virtex-4 SX), and gradually reworked towards a lower-end device solution (i.e. Spartan-3A). This is possible due to similarities in their architecture, particularly the XtremeDSP™ slices (also called DSP48 slices) in Virtex-4 devices, designs can be ported between XtremeDSP™ devices in a reasonably straight-forward manner (Xilinc Inc 2008).

This paper focuses on the FPGA design of a dual microphone delay-and-sum beamformer for speech enhancement specifically designed toward cheaper and efficient solutions for in-car environments. In Section II, the dual-microphone enhancement method using DASB is presented. Section III describes a fixed-point FPGA implementation of the algorithm. Experimental results verifying the FPGA design are presented in Section IV. This is followed by discussion and conclusions in Section V.

II. DELAY-AND-SUM BEAMFORMING

Beamforming is a method of spatial filtering which differentiates desired signals from noise and interference based on their location. The direction where the array of microphones is steered is called the look direction. The simplest beamforming algorithm is the delay-and-sum beamformer which works by compensating signal delay to each microphone appropriately before they are combined using an additive operation. The outcome of this delayed signal summation is reinforcement of the desired signal while the noise in each microphone tends to cancel each other.

The illustration of the dual-microphone DASB is given in Fig. 1. Consider a desired signal received by N omni-directional microphones sampled at discrete time k, in which the input to each microphone is an attenuated and delayed version of the desired signal $a_n s(k-\tau_n)$ and noise v_n given by:

$$x_n = a_n s(k-\tau_n) + v_n(k) \tag{1}$$

In the frequency domain via Fourier transform, the array signal model is:

$$\mathbf{X}(\omega) = S(\omega)\mathbf{d} + \mathbf{V}(\omega) \tag{2}$$

where **d** represents the array steering vector which depends on the actual microphone and source locations. For a source located near the array, the wavefront of the signal impinging on the array should be considered a spherical wave and the source signal is said to be located within the near-field of the array instead of a planar wave commonly assumed for a source located far from the array. In the near field, **d** is given by (Bitzer & Simmer, 2001):

$$\mathbf{d} = [a_1 e^{-j\omega\tau_1}, a_2 e^{-j\omega\tau_2}, \ldots, a_N e^{-j\omega\tau_N}]^T \tag{3}$$

$$a_n = \frac{d_{ref}}{d_n},\ \tau_n = \frac{d_n - d_{ref}}{c} \tag{4}$$

where d_n and d_{ref} denote the Euclidian distance between the source and the microphone n, or the reference microphone,

respectively, and *c* is the speed of sound.

To recover the desired signal, each microphone output is weighted by frequency domain coefficients $w_n(\omega)$. The beamformer weights are designed to maintain the beam at the look direction to be constant (e.g. $\mathbf{w^H d} = 1$). For a dual microphone case, the beamformer output is the sum of each weighted microphone:

$$Y(\omega) = \sum_{n=1}^{2} w_n{}^{H}(\omega) X(\omega) \tag{5}$$

The inverse Fourier transform results in time domain signal *y(k)*.

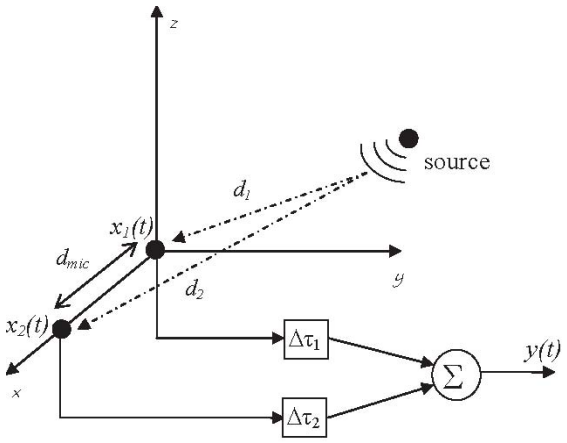

Fig. 1. Dual-microphone delay-and-sum beamforming.

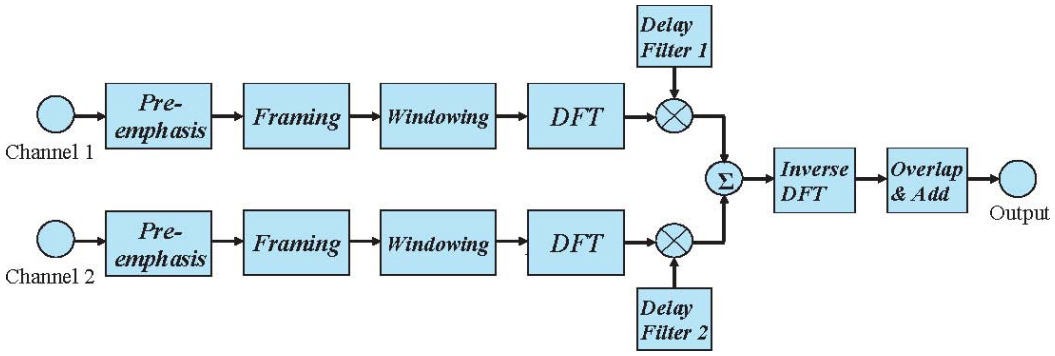

Fig. 2. The block diagram of the dual-microphone DASB system.

III. FPGA IMPLEMENTATION

In this section, the FPGA implementation of the dual microphone DASB technique is described. The target device for the implementation was a Xilinx Virtex-4, however the need to be able to realise the design on a low-cost FPGA device motivated a number of the design decisions discussed below.

A. Structure of the Two-Channel DASB System

The delay filters of DASB are realised in the frequency domain due to the ease of expressing the unity gain and linear phase shift transfer function directly. In the time domain this would require a fractional delay filter. The realisation of frequency domain filtering leads to the overall structure for the two-channel DASB system as shown in Fig. 2.

From Fig. 2, the common speech analysis blocks of pre-emphasis, overlapped framing, Hamming windowing and Discrete Fourier Transform (DFT) are required to transform each frame to the frequency domain. Once in the frequency domain, delay filtering is performed by multiplying each frame of the signal with the coefficients of the corresponding delay filter. After delay filtering, the two resultant signals are summed and transformed back into the time domain. The final signal is

reconstructed by overlapping and adding successive output frames.

B. Overall Design Strategy

1) 3-FFT design: This is the most straightforward strategy. Its overall structure is almost identical to Fig. 2, except that the DFT and Inverse DFT blocks in Fig. 2 are replaced by the standard FFT/IFFT blocks. The most obvious advantage of the 3-FFT design strategy is that there is almost no need for buffering because there are no blocks in the system that are shared by different processing tasks. However, one obvious disadvantage of this design strategy is that three FFT/IFFT blocks – which are high on DSP48 resource usage – are employed. While this DSP48 resource requirement may not be too high for the Virtex-4 device, it most probably is for a low-cost device. This is the reason for considering a 1-FFT design strategy.

2) 1-FFT design: As shown in Fig. 3, an alternative strategy employs only one FFT/IFFT block. This design shares the FFT/IFFT block, using it three times (twice for FFT, once for IFFT) during each frame period. Although this arrangement requires more general resources for buffering, one instead of three FFT/IFFT blocks will significantly reduce the DSP48 resources, which are very limited in a low-cost FPGA device such as a Spartan-3A DSP. According to this observation, the 1-FFT design strategy is chosen for this implementation.

C. Timing Arrangement

In general, the timing arrangement for the 1-FFT system must satisfy the following two requirements:

1) There is no time overlap between shared blocks – that is, data should not be fed into the block while it is still processing previous data.
2) The entire system remains a real-time system where there is no significant accumulated processing delay.

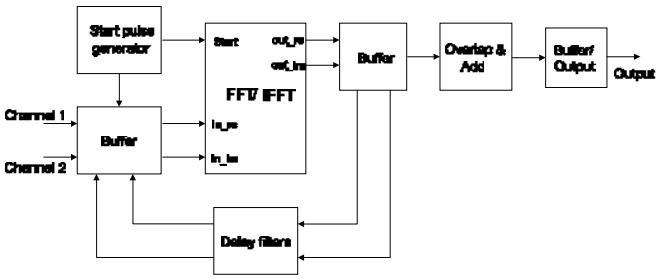

Fig. 3. The overall structure of the 1-FFT design strategy.

The key to a correct timing arrangement is the construction of the start pulse signal which is connected to the start pin of the standard FFT/IFFT block. The rising edges of this pulse signal determine the times at which the FFT/IFFT operation starts. To meet the first requirement above, the start pulse for the second channel data must not occur before the FFT processing of the first channel data is finished and the start pulse for the IFFT operation must not occur before the output data is ready. To meet the second requirement, all three FFT/IFFT operations must be finished before the next frame of data is fed into the buffer.

This system uses 512 samples in each frame and there is a 50% overlap between the neighbouring frames, so all three FFT/IFFT operations need to finish within a 256 data sample period. For the FFT/IFFT block used, the output data will be ready exactly 5210 clock cycles after the rising edge of the start pulse. With a 50 MHz clock for the FPGA hardware, this means that the rising edge of the next start pulse must be at least 104.2 μs later.

The real-time processing requirement can be easily satisfied for the input data sampling frequency of 16 kHz (i.e. sample period of 62.5 μs), as 153 FFT/IFFT operations could be performed within the 16ms 256 data sample period. To speed up development it was desirable to make use of some of this unused processing bandwidth by allowing for a data sample rate of 80 kHz. This factor of five increase enabled Xilinx System Generator (XSG) simulations to run faster by roughly the same order of magnitude. Setting the gap between the three start pulses (for each data frame) to ten times the increased data sample period of 12.5 μs easily meets the FFT/IFFT latency requirement of 104.2 μs. The resultant start pulse sequence is shown in Fig. 4, which works for both 16 kHz and 80 kHz data sampling rates.

D. Other Aspects of the Implemented System

1) The pre-emphasis filters: The common practice to re- move spectral tilt using a pre-emphasis filter is characterised by the following difference equation:

$$y(i) = x(i) - 0.97x(i-1) \tag{6}$$

where *x(i)* and *y(i)* are the i^{th} input and output samples, respectively. Its implementation requires a delay block, a multiplier and an adder.

As shown in Fig. 2, the conventional position for the pre-emphasis filter is before any other processing blocks. However, since the delay filter for each channel is applied later in the system, the whole process can be regarded as a cascade of the pre-emphasis filter and the delay filter. Therefore, these two filters can be combined and applied in the frequency domain. This will save the DSP48 resources that would otherwise be required for the two pre-emphasis filters.

2) Hamming windowing: This can be easily implemented with a block of Read Only Memory (ROM) (for the Hamming window values) and two multipliers (one for each channel).

3) Delay filters: This can be implemented in a similar manner to the Hamming window, using the frequency domain representation of the transfer function.

4) Overlap-and- add reconstruction: The processed time domain frames must be properly overlapped and added to produce the final reconstructed speech signal. This reverses the framing process performed at the start of the algorithm. This is implemented using an addressable shift register and an addition block.

5) The buffer/output block: The output from the overlap- and-add block consists of bursts of 256 samples at the system clock rate and there are long delays between these bursts. To make sure that the system emits output samples at a uniform rate, we include a buffer/output block.

6) Buffers: These are realised using addressable shift registers.

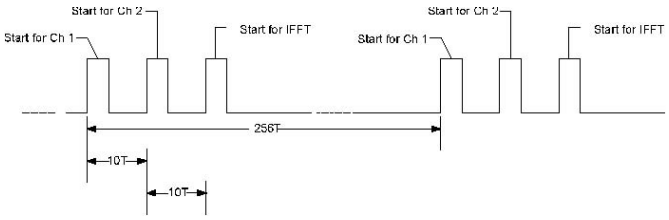

Fig. 4. The start pulse sequence for the 1-FFT system. The parameter T represents the test input data sampling period which is one fifth of the operational value.

E. Design Process

The XSG was used as the primary tool for developing the implementation of the DASB hardware design. XSG is an FPGA development environment that sits above the MATLAB and Simulink software packages (Xilinc Inc 2008). The XSG package contains predefined blocks that can be readily compiled into a hardware description language (HDL) and subsequently synthesized for specific Xilinx FPGAs. Designers can also incorporate their own HDL descriptions into the design. The XSG tool does not provide fully optimised FPGA solutions, however through the access to predefined blocks, it greatly reduces the prototyping time.

The DASB design was originally developed as MATLAB scripts using high precision, complex floating-point arithmetic. To produce an FPGA hardware solution, fixed-point equivalents for the complex operations were required. Furthermore, minimal resource utilisation is necessary to reduce costs. As quality and cost, in terms of resource usage, tend to go hand-in-hand, a multi-step process is usually required with considerable testing and evaluation at each stage. Our approach was to build the DASB XSG model block-by-block based on similar sections of the floating-point algorithm. On completion, each block was tested to ensure correct operation before the next block was developed. Some optimisation of resource usage was also carried out at this time. To verify each stage of the design, the XSG model was simulated (via the inbuilt XSG simulator) applying known inputs and comparing outputs to that of the floating-point algorithm, with a close match indicating an appropriate implementation.

Once all sections of the initial XSG design were complete, they were integrated and the entire design was converted into HDL code, synthesised using Xilinx ISE 9.2 tools and implemented on a Xilinx Virtex-4 SX FPGA. For initial hardware development work, the ML 402 development board (containing a Virtex-4 SX35 device) was used. The FPGA design was then tested against the complete floating-point algorithm, the results of which lead to further refinement and optimisation of the design.

IV. VERIFICATION OF FPGA DESIGN

A. System Verification

Testing involved passing selected input files to the FPGA DASB design and collecting the output as a file for analysis. This was performed using a test harness developed for work with a spectral subtraction design (Whittington *et al.* 2008), modified to provide two channel inputs. The basic block diagram of the test harness is shown in Fig. 5.

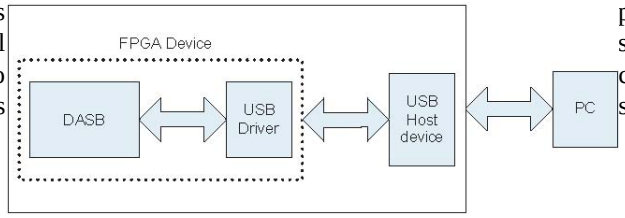

Fig. 5. Block diagram of test harness used in system verification.

To verify that the designed system worked correctly, the ramp signals shown in Fig. 6 were used. The system was setup to have symmetrical delay filters, i.e. distances from the two microphones to the speaker are the same. In this case, the ideal output should be a constant zero.

The XSG simulation and FPGA outputs are shown in Fig. 7, whilst Fig. 8 shows the difference between the two outputs. The maximum difference in amplitude is within $1x10^{-4}$ representing approximately 4-bits of quantisation error.

Another test to verify the design was using modulated chirp signals as shown in Fig. 9. A modulated chirp signal is a signal in which frequency increases with time (up-chirp) which is modulated by another chirp so that the amplitude also varies with time. The purpose of this signal is to provide frequency sweeps across time to test the frequency response of the designed filters so that any frequency related errors can be quantified.

The system was setup to use asymmetrical delay filters. The floating-point MATLAB and FPGA outputs are in Fig. 10 with the amplitude difference between the two outputs shown in Fig. 11. The maximum difference is again in the region of 4-bits quantisation error.

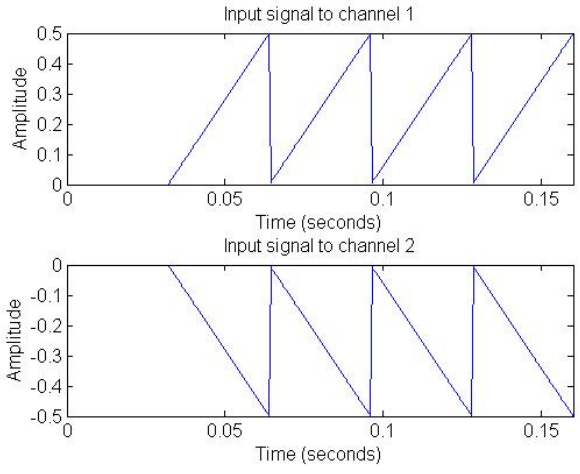

Fig. 6. Ramp signals used to verify the designed system.
A ramp signal was fed to Channel 1 and its inverse fed to Channel 2.

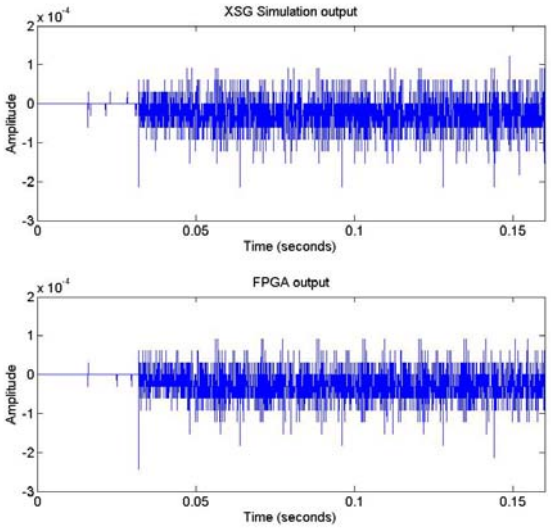

Fig. 7. The output of XSG simulation and FPGA using ramp signals as input. The accuracy of the FFT/IFFT block is set at 24 bits.

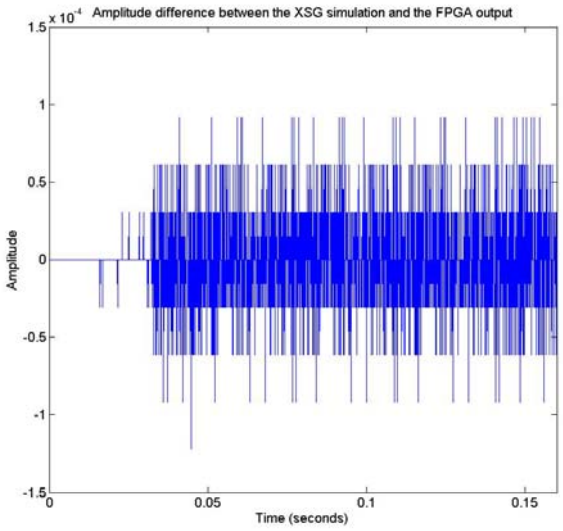

Fig. 8. Amplitude difference between the XSG simulation and FPGA output using ramp signals as input.

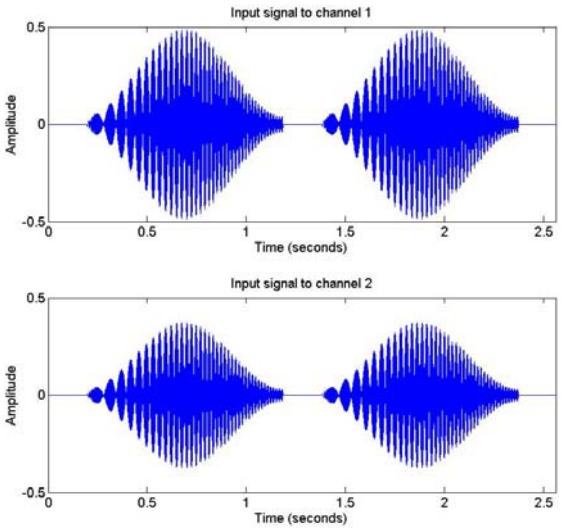

Fig. 9. Modulated chirp signals used to verify the designed system.

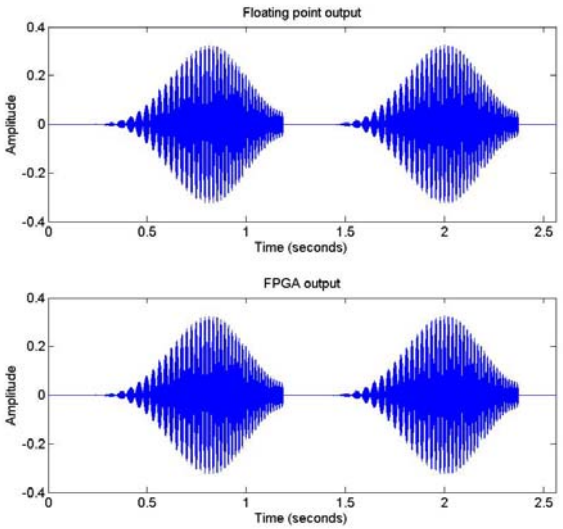

Fig. 10. The output of floating-point MATLAB and fixed-point FPGA using modulated chirp as input.

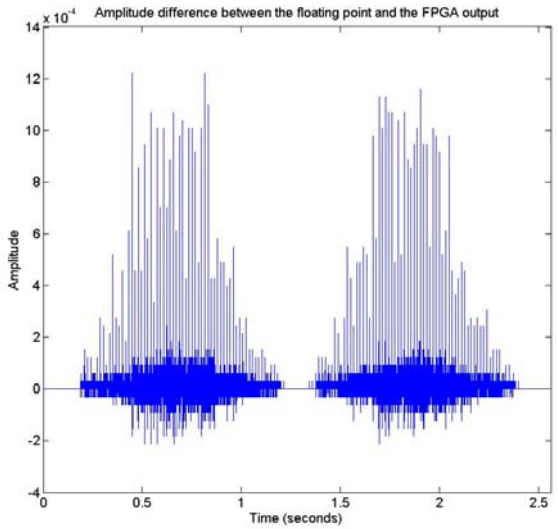

Fig. 11. Amplitude difference between the floating-point MATLAB and fixed-point FPGA outputs using modulated chirp signals as input.

B. Test Results on Real Speech Data

The test inputs involve real speech recordings from microphone arrays in a car environment. For the symmetrical case, test files are obtained from the AVICAR database (Lee *et al.* 2004). Microphone 2 is chosen as Channel 1 and that from Microphone 6 as Channel 2. Fig. 12 shows an example of two input signals. The output from the floating-point MATLAB version and that from FPGA are shown in Fig. 13. Fig. 14 shows the difference between the two outputs.

For the asymmetrical case, the test files were obtained from data collected in conjunction with this work (Kleinschmidt *et al.* 2009). The signal from Microphone 0 as Channel 1 and that from Microphone 3 as Channel 2. Fig. 15 shows the input signals. The output of the floating-point MATLAB version of the system and that of the FPGA is shown in Fig. 16. Fig. 17 shows the difference between the two outputs.

The amplitude of the difference plots again validates that the FPGA implementation generates outputs that are close to that generated by the floating-point model.

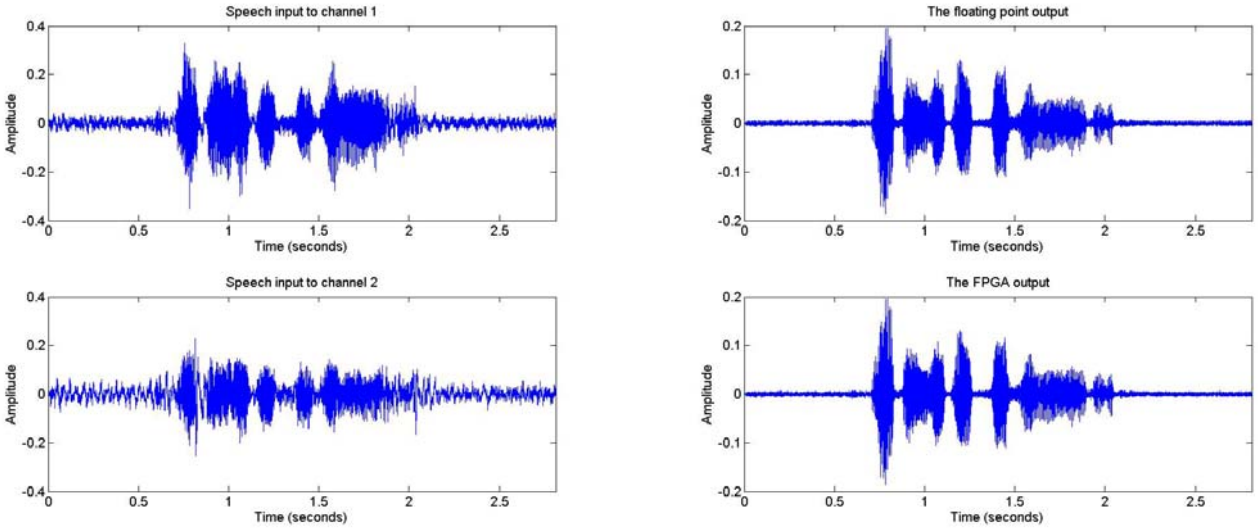

Fig. 13. Output of the floating-point MATLAB and FPGA delay-and-sum beamformers using symmetrical delay filters.

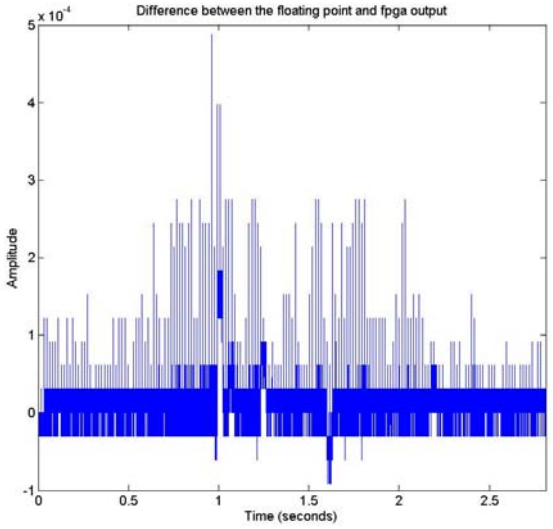

Fig. 14. Difference between the output of the floating-point MATLAB version and that of the Virtex-4 FPGA DASB.

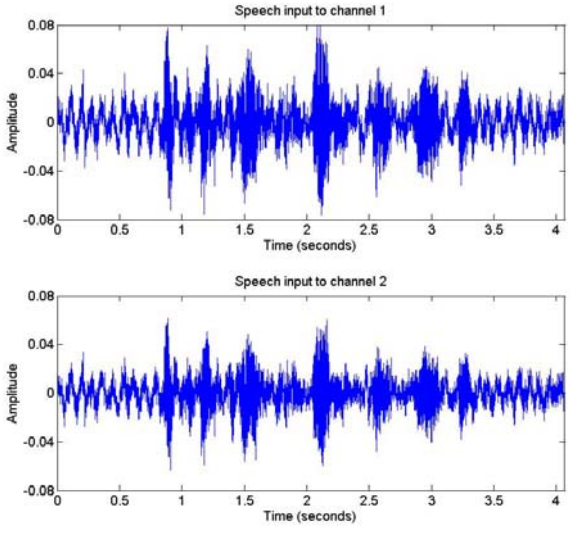

Fig. 15. Input speech signals for asymmetrical delay filters.

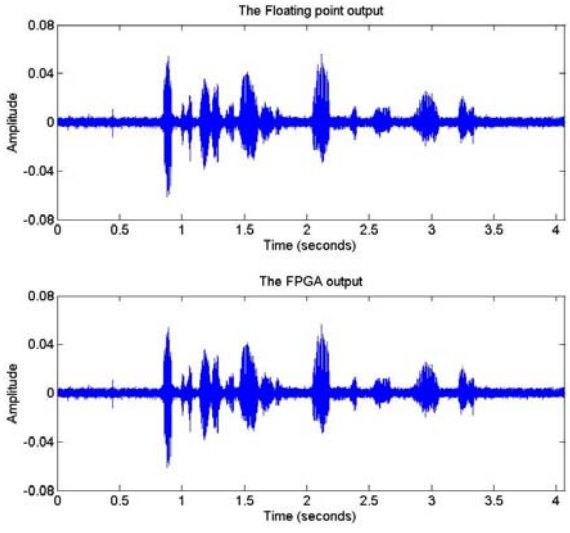

Fig. 16. Outputs of the floating-point MATLAB and FPGA delay-and-sum beamformers using asymmetrical delay filters.

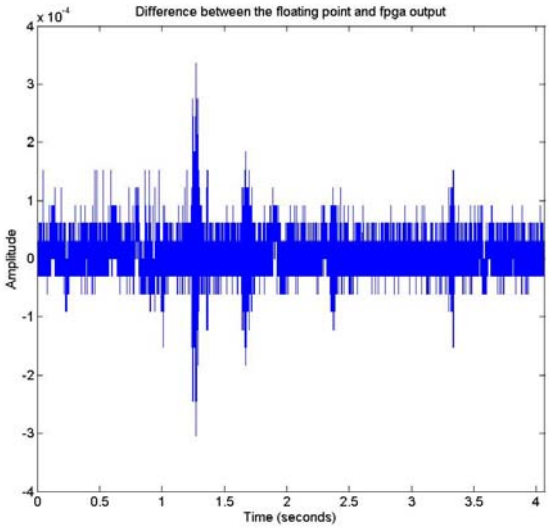

Fig. 17. Difference between the output of the floating-point MATLAB version and that of the designed Virtex-4 FPGA DASB.

C. Speech Recognition Experiments

To evaluate the impact of the quantisation error on recognition performance, the floating-point and FPGA implementations of dual microphone delay-and-sum beamforming were applied to the AVICAR database (Lee *et al.* 2004), (Kleinschmidt *et al.* 2007) and the Australian English In-Car Speech Database collected in conjunction with this work (Kleinschmidt *et al.* 2009). The microphone pairs used are those described in Section IV. Utterance decoding was performed using the Hidden Markov Model Toolkit (Young *et al.* 2006). Results are provided in Tables I and II.

For both the AVICAR database and the Australian English In-Car Speech Database, the difference between the recognition rate using the floating-point model and that using the FPGA implementation is typically below 0.1% for all noise conditions. This confirms that the quantisation error of the FPGA implementation has very little impact on the speech recognition performance.

Noise condition	*Baseline*	*Floating point*	*FPGA output*
Idle	28.4	19.6	19.7
35U	50.4	36.2	36.2
35D	62.8	47.0	47.1
55U	57.1	43.2	43.1
55D	75.3	62.8	62.8
Overall	**54.8**	**41.7**	**41.8**

Table I: Speech Recognition Performance Using Phone Numbers Task of the AVICAR Database. All Figures in % Word Error Rate.

Noise condition	*Baseline*	*Floating point*	*FPGA output*
C0	15.1	15.3	15.4
C1	29.8	30.8	30.5
C2	65.2	59.3	59.3
C3	46.9	56.4	56.0
C4	45.9	46.4	46.5
C5	68.4	62.4	62.3
C6	58.3	52.4	52.4
Overall	**47.5**	**46.6**	**46.5**

Table II: Speech Recognition Performance Using the Australian English In-Car Speech Database. All Figures in % Word Error Rate.

D. FPGA resource utilisation

Table III summaries the resource usage of the current design. Slices, which represent the general FPGA logic fabric, only utilise 31% of the total resources available. While the current design fits comfortably into a Virtex-4 SX device, the aim is to implement the DASB system on a low-cost FPGA such as the Xilinx Spartan-3A DSP device. The resource utilisation for the Virtex-4 SX device indicates that it is highly probable that this design can be directly implemented on a low-cost FPGA device, such as Xilinx Spartan-3A DSP.

Resource Type	*Available*	*Usage*
Slices	15360	4829 (31%)
Flip Flops	30720	2283 (7%)
4-Input LUTs	30720	8078 (26%)
FIFO16/RAMB16s	192	9 (4%)
DSP48s	192	22 (11%)

Table III: Virtex-4 FPGA Resource Usage Summary.

V. Discussion & Conclusion

In this paper, a successful FPGA implementation of a speech enhancement technique – dual microphone DASB for application to in-car speech recognition – has been presented. As an initial step, the implementation was carried out on a Xilinx Virtex-4 SX device. In the design, a 1-FFT strategy which saves on the DSP48 resources, which are limited on low-cost FPGA devices, has been adopted. Experimental results show that the hardware implementation generates outputs that are very close to theoretical (floating point) results and therefore confirm the correctness of the design. The small quantisation error has little effect on the speech recognition accuracy of two speech databases with differing microphone configurations. The actual resource usage of the design shows promising potential of implementing the current system on a low-cost FPGA device.

VI. References

[1] Lim, JS & Oppenheim AV 1979, "Enhancement and bandwidth com- pression of noisy speech", *Proc. IEEE*, pp. 1586–1604.

[2] Boll, S 1979, "Suppression of acoustic noise in speech using spectral subtraction", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 27, no. 2, pp. 113–120.

[3] Berouti, M, Schwartz, R & Makhoul, J 1979, "Enhancement of speech corrupted by acoustic noise", *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, pp. 208–211.

[4] Ephraim, Y & Malah, D 1984, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 6, pp. 1109–1121.

[5] Johnson, DH & Dudgeon, DE 1993, "*Array Signal Processing: Concepts and Techniques"*, Prentice Hall, New Jersey.

[6] Aarabi, P & Shi, G 2004, "Phase-based dual-microphone robust speech enhancement", *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 34, no. 4, pp. 1763–1773.
[7] Ahn, S & Ko, H 2005, "Background noise reduction via dual-channel scheme for speech recognition in vehicular environment", *IEEE Transaction on Consumer Electronics*, vol. 51, no. 1, pp. 22–27.
[8] Beh, J, Baran, RH & Ko, H 2006, "Dual channel based speech enhancement using novelty filter for robust speech recognition in automobile environment", *IEEE Transaction on Consumer Electronics*, vol. 52, no. 2, pp. 583–589.
[9] Halupka, D, Rabi, AS, Aarabi, P & Sheikholeslami, A 2007, "Low-power dual-microphone speech enhancement using field programmable gate arrays", *IEEE Transactions on Signal Processing*, vol. 55, no. 7, pp. 3526–3535.
[10] Bagni, D & Zoratti, P 2007, "Block matching for automotive applications on Spartan-3A DSP devices", *XCell Journal*, no. 63, pp. 16–19.
[11] Whittington, J, Deo, K, Kleinschmidt, T & Mason, M 2008, "FPGA implementation of spectral subtraction for in-car speech enhancement and recognition", *2nd International Conference on Signal Processing and Communication Systems*.
[12] Kitagawa, K 2007, "At the heart of consumer and automotive innovation", XCell Journal, no. 63, pp. 12-13.
[13] Xilinx Inc 2007, "*Xilinx Automotive - flexible solutions beyond silicon"*, Xilinx Inc.
[14] Xilinx Inc 2008, "*XtremeDSP Solutions Selection Guide"*, Xilinx Inc.
[15] Bitzer, J & Simmer, KU 2001, "Superdirective microphone arrays", *Microphone Arrays*, M. S. Brandstein and D. B. Ward, Eds. Springer, ch. 2, pp. 19–38.
[16] Lee, B, Hasegawa-Johnson, M, Goudeseune, C, Kamdar, S, Borys, S, Liu, M & Huang, T 2004, "AVICAR: Audio-visual speech corpus in a car environment", *Proc. of Interspeech*, pp. 2489–2492.
[17] Kleinschmidt, T, Mason, M, Wong, E & Sridharan, S 2009, "The Australian English corpus for in-car speech processing", to be presented at IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP), Taiwan.
[18] Kleinschmidt, T, Dean, D, Sridharan, S & Mason, M 2007, "A continuous speech recognition protocol for the AVICAR database", *1st International Conference on Signal Processing and Communication Systems*, Gold Coast, Australia, pp. 339–344.
[19] Young, S, Evermann, G, Gales, M, Hain, T, Kershaw, D, Liu, X, Moore, G, Odell, J, Ollason, D, Povey, D, Valtchev, V & Woodland, P 2006, "*The HTK Book*, 3rd ed.", Cambridge University Engineering Department.