Analyzing the Trade-Off between Power Consumption and Beamforming Algorithm Performance using a Hearing Aid ASIP

Lukas Gerlach, Guillermo Payá-Vayá, Shuang Liu,
Moritz Weißbrich, and Holger Blume
Institute of Microelectronic Systems
Cluster of Excellence Hearing4All
Leibniz Universität Hannover, Germany
Email: {gerlach, guipava, blume}@ims.uni-hannover.de

Daniel Marquardt and Simon Doclo
Department of Medical Physics and Acoustics
Cluster of Excellence Hearing4All
Carl von Ossietzky Universität Oldenburg, Germany
Email: {daniel.marquardt, simon.doclo}@uni-oldenburg.de

***Abstract*—In this paper, the effects of application-specific instruction-set processor (ASIP) hardware optimizations on the performance of beamforming algorithms and on the hardware requirements (i.e., silicon area and power consumption) are studied. For that, the performance of three beamforming algorithms with different fixed-point implementations are compared using objective instrumental measures, i.e., PESQ, STOI, and iSNR. The proposed application-specific hardware optimizations are implemented in a VLIW-SIMD hearing aid processor, modifying the processor's datapath width, using a co-processor for the division operation and applying register file power optimizations. In total 24 different optimized processor configurations are studied. The result of this evaluation is that the same processor, running one of the beamformers, can be optimized, decreasing up to 2 times the silicon area requirements or up to 11 times the power consumption, thereby only slightly decreasing the overall algorithm performance (e.g., -2dB iSNR for a fixed beamformer).**

I. Introduction

Hearing impaired persons often suffer from a loss of speech understanding resulting in difficulties when interacting in social groups. Especially in complex acoustic scenarios where several people are talking simultaneously, i.e., a so-called cocktail party scenario, speech intelligibility may be substantially degraded, such that communicating with other people becomes a serious challenge. Hence, in addition to standard hearing aid processing algorithms, such as frequency-dependent amplification and dynamic range compression, noise reduction algorithms in hearing aids are crucial to improve speech intelligibility [1], [2].

Modern digital hearing aids are equipped with two or more microphones [2], enabling multichannel digital signal processing [3], [4]. In order to increase speech intelligibility for the hearing impaired, beamforming algorithms are frequently used to suppress undesired components and to increase the signal-to-noise ratio (SNR) of the target speech signal. Fixed beamformers, also denoted as data-independent beamformers, are typically designed such that the signals arriving from a certain direction are passed through without any distortion. In contrast to fixed (data-independent) beamformers, adaptive (data-dependent) beamformers also exploit the signal statistics of the noise component in order to adapt to changing noise fields. The achieved algorithm performance in terms of improvement in SNR and in speech intelligibility for hearing aid users depends on the implementation of the algorithm and the acoustic scenario.

Besides the performance of the algorithm, the battery life of digital hearing aids is an important aspect for user acceptance. Battery life depends on power consumption, which is partially consumed by the hearing aid's digital processor. The power consumption of the processor depends on its architecture, its utilization, its operating clock frequency, and other factors. Specializing a processor architecture for a target appliction by customizing its instruction-set architecture is a common technique used to reduce power consumption while keeping the required processing performance. The resulting processor is called application-specific instruction-set (ASIP) processor. In this paper, application-specific optimizations are evaluated based on selected target hearing-aid algorithms and their performance. The main contributions are:

- Novel ASIP power consumption optimizations, including the modification of the processor's datapath width, register file power optimizations and a co-processor for the division operation;
- An evaluation method, which analyzes the hardware optimizations based on accurate power consumption and silicon area estimations, by comparing these to the performance of the hearing aid algorithms using objective instrumental measures.

This paper is organized as follows. In Section II, related work and the contribution of this paper are described. Section III gives a brief overview of the beamforming algorithms considered, which are then evaluated in Section IV using objective measures. The hearing aid processor and the ASIP optimizations are stated in Section V. The results of the power consumption, silicon area, and algorithm performance evaluation are presented in Section VI.

II. Related Work

Among the related works, which evaluate the power consumption of hearing aid architectures, are two dedicated implementations [5], [6] and six application-specific digital signal processors (ASIPs) [7]–[12]. In [5], an optimized ultra low-power delayed least mean squares (DLMS) adaptive filter in a parallel implementation for hearing aid applications is presented. This filter is implemented in pseudo nMOS and is operated in the subthreshold region. Simulations and chip tests show that the optimized filter architecture offers the same throughput with 90% less power consumption. The authors of [6] present an asynchronous low-power filter bank for digital hearing aids. Compared to the synchronous implementation, the asynchronous implementation with an equivalent functionality consumes a factor 5.5 less power. In [7], the TMS320C5000 DSP is evaluated using a hearing aid development platform. Different implementations of this DSP, using different technologies and supply voltages, are evaluated in terms of power consumption. The initial value of 0.8 $\frac{mA}{MIPS}$ is reduced to 0.18 $\frac{mA}{MIPS}$. A mixed-signal hearing aid SoC is presented in [12]. The total average power consumption of the fabricated chip is $1.1\,\mathrm{mW}$. The power consumption is given separately for the 24-bit ASIP ($0.44\,\mathrm{mW}$), the analog front end ($0.39\,\mathrm{mW}$) and other components ($0.27\,\mathrm{mW}$). In [8], the MACGIC DSP is proposed. The average power consumption of this chip increases from 150 $\mu W/MHz$ to 300 $\mu W/MHz$ when computing 'power-hungry' [8] algorithms like the Fast Fourier Transform (FFT). The same applies to the estimated values for the power consumption of the Freescale Starecore and the Philips CoolFlux DSPs, whose power consumption rises by a factor of 2.3 and 1.7, respectively.

This work is focused on the power consumption evaluation of hearing aid ASIP optimizations based on the performance of different beamforming algorithms. Compared to the previously mentioned related work and [9]–[12], which only mention the average or maximum power consumption, this work provides a detailed and application-specific power consumption evaluation. The power consumption evaluation includes the description of several ASIP optimizations and their impact on the power consumption during the processing of different exchangeable beamforming algorithms. Furthermore, this work evaluates the performance of different beamforming algorithms using objective measures. The performance of these algorithms is then compared to the power consumption, which enables a trade-off between the algorithm performance and power consumption. The proposed hardware optimizations are not only evaluated in terms of the average power consumption, silicon area overhead, and processing performance gain, but also in terms of the application-specific power consumption.

III. Evaluated Beamforming Algorithms

The following three dual-microphone monaural beamforming algorithms have been used for the evaluation:

- Fixed Beamformer [1], [13]
- Adaptive Gain Beamformer [1], [13], [14]

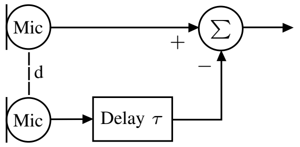

Fig. 1. Fixed differential beamformer according to [1]. One of the microphone signals is delayed by a constant time value τ and subtracted from the other microphone signal.

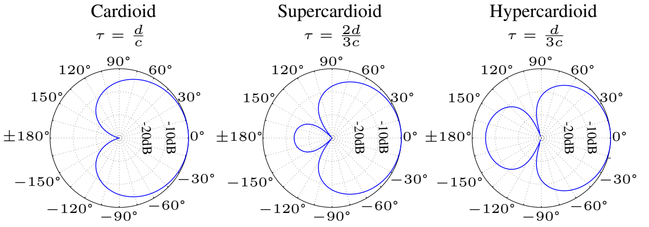

Fig. 2. Fixed beamformer patterns for different constant delays τ: Cardioid ($\tau = \frac{d}{c}$), supercardioid ($\tau = \frac{2d}{3c}$) and hypercardioid ($\tau = \frac{d}{3c}$), where d is the distance between the microphones and c is the speed of sound.

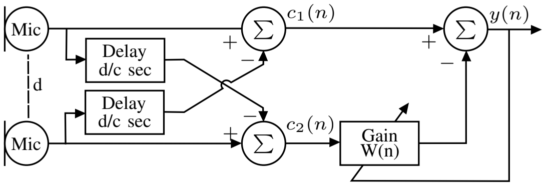

Fig. 3. Adaptive gain beamformer according to [13], [14]. The output is the subtraction of the front-facing cardioid and the adaptively weighted back-facing cardioid response.

- Adaptive Filter Beamformer [1], [13], [15]

The fixed differential beamformer, which is described in [1], is shown in Fig. 1. This beamformer uses two omnidirectional microphones, which are closely spaced at a distance d. Due to this distance, sound waves arriving from different angles reach the microphones at different times. The rear microphone signal of the hearing aid is delayed by a constant time value τ and subtracted from the front microphone signal, resulting in a directional pattern. A common assumption for the use of this directional pattern is that the desired speaker is located in front of the hearing aid user, whereas the interfering sounds are located behind the hearing aid user. The resulting directional patterns, which indicate the sensitivity for different sound source angles, are shown in Fig. 2 for three different constant delays. The resulting directional patterns are called cardioid, supercardioid, and hypercardioid.

Contrary to fixed beamformers, adaptive beamformers are able to adapt to the changing spatial characteristics of the interfering sounds. Two adaptive beamforming algorithms are shown in Fig. 3 and Fig. 4. The microphone inputs are first used to generate a front- and a back-facing cardioid response. For the adaptive gain beamformer (Fig. 3) the back-facing response $c_2(n)$ is weighted with a time-varying scalar $W(n)$ and subtracted from the front-facing response $c_1(n)$, such that the output of the adaptive gain beamformer is equal to:

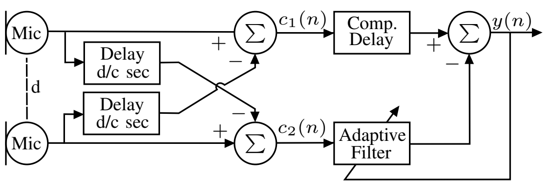

Fig. 4. Adaptive filter beamformer according to [1]. The output is the subtraction of the front-facing cardioid and the adaptively filtered back-facing cardioid response.

$$y(n) = c_1(n) - W(n) \cdot c_2(n) \tag{1}$$

Assuming a frame-by-frame processing with frame index m and the frame length $M = 64$, the following adaptation for the gain factor $W(m)$ is used [14]:

$$W(m) = \frac{\hat{R}_{c_1c_2}(m)}{\hat{R}_{c_2c_2}(m)} \tag{2}$$

$$\hat{R}_{c_1c_2}(m) = \frac{\alpha}{M}\sum_{n=1}^{M} c_1(m \cdot M + n)c_2(m \cdot M + n) + (1-\alpha)\hat{R}_{c_1c_2}(m-1) \tag{3}$$

$$\hat{R}_{c_2c_2}(m) = \frac{\alpha}{M}\sum_{n=1}^{M} c_2^2(m \cdot M + n) + (1-\alpha)\hat{R}_{c_2c_2}(m-1) \tag{4}$$

where $\alpha = 0.5$ is an adjustable recursive smoothing parameter.

The adaptive filter beamformer, presented in [1] and shown in Fig. 4, uses an adaptive FIR filter with a compensation delay in order to allow for acausal filter taps. The filter coefficients are updated using the following LMS adaptation:

$$\boldsymbol{w_k}(n+1) = \beta \cdot \boldsymbol{w_k}(n) + \mu \cdot \boldsymbol{c_2}(n) \cdot y(n) \tag{5}$$

with $\beta = 1.0$ and $\mu = 0.3$ adjustable parameters to control the adaptation speed.

IV. Objective Performance Evaluation of the Beamforming Algorithms

Objective instrumental measures have been used in this work to compare the performance of different beamforming algorithms. In Section VI, the outcomes of this evaluation are used to compare the algorithm performance obtained with hardware related requirements, such as static and dynamic power consumption for running these algorithms on a hearing aid processor.

The algorithm evaluation is based on the setup described in [16] for pre-processing strategies. An acoustic test scenario is created using a database of behind-the-ear impulse responses [17]. Each of the behind-the-ear hearing aid was equipped with 2 microphones at a distance of about 7.6 mm and was mounted on an artificial head. From this database, the anechoic impulse responses[1] have been used to generate the hearing aid microphone signals for an acoustic scenario comprising one target and one interfering sound source. The target signal is a male speaker (male.wav) and the interfering signal is babble noise (babble_olsa.wav), both recorded from the Oldenburg sentence test database (OLSA) at a sampling frequency of 16 kHz [18]. The target source is always at 0° in front of the head, while different interfering source angles ranging from 0° to 180° around the head are considered. To evaluate the performance of the fixed and the adaptive beamforming algorithms, the following instrumental measures are used:

- Perceptual evaluation of speech quality (PESQ) [19]
- Short time objective intelligibility index (STOI) [20]
- Intelligibility-weighted signal-to-noise ratio (iSNR) [21]

The results of this evaluation, i.e., the PESQ, STOI and iSNR scores for all considered angles of the interfering source, are shown in Fig. 5. The fixed beamformer is configured to generate either a cardiod, a supercardiod, or a hypercardiod response with a constant spatial null in its directional pattern (see Fig. 2). Based on the PESQ, STOI, and iSNR scores, it can be observed that the adaptive beamformers yield a better performance on average for interfering sound source angles larger than 90° compared to the fixed beamformers. The reason for this performance difference is the adaptation to the angle of the interfering sound source.

The data type used for the algorithm implementation and for the results shown in Fig. 5 is double precision floating-point. For power consumption reasons, fixed-point processors are used in hearing aids. Therefore, the influence of the quantization and rounding errors of the fixed-point beamformer algorithm implementations on the algorithm performance is studied. Four different implementations with floating-point and three fixed-point data types with different word lengths are compared. The average PESQ, STOI, and iSNR scores for interfering source angels larger than 90° are given in Table I. The 32-bit fixed-point implementation offers about the same algorithm performance compared to the double precision implementation (maximum of 1% deviation). The 24-bit and the 16-bit implementations have the same fixed-point formats as the 32-bit implementation with a reduced fraction length. The algorithm performance of the 24-bit adaptive filter beamformer is slightly decreased whereas both 16-bit adaptive beamformers become instable and do not converge due to reduced fixed-point word width [22], resulting in a not working adaptation.

V. Proposed Application-Specific KAVUAKA Hearing Aid Processor Hardware Optimizations

The beamforming algorithms are implemented on a low-power hearing aid processor, which is called KAVUAKA. This processor was presented in [23] and the architecture, extensions and co-processors are shown in Fig. 6 and listed in

[1] Although in practice obviously also reverberation is present and more robust versions of the discussed (adaptive) beamforming algorithms should be considered [3], [4], we believe that the considered anechoic scenario suffices for the trade-off analysis in this paper.

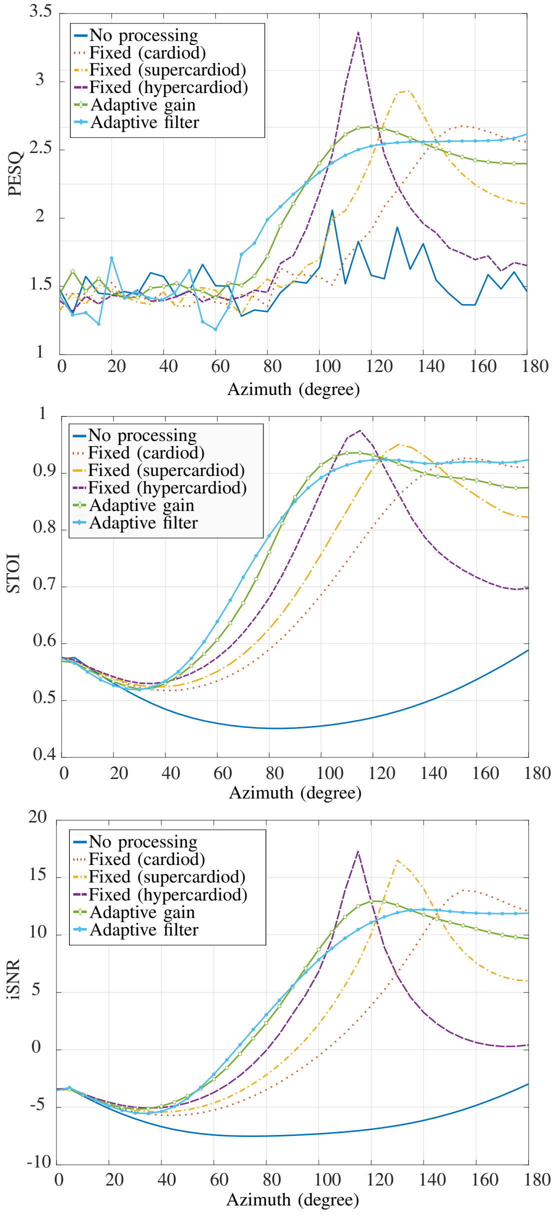

Fig. 5. PESQ, STOI, and iSNR scores for the different fixed and adaptive beamforming algorithms as a function of the azimuth angle of the interfering sound source. Double Precision Floating-Point is used in this case.

Table II. The processor executes very long instruction words (VLIW) in two issue-slots. Within the first pipeline stage the instructions are fetched and decoded and the registers are accessed. The second pipeline stage includes the SIMD functional units and the register file write back and forwarding paths. The processor is coupled to an audio and serial interface as well as external co-processors.

In order to optimize this hardware architecture for the beamforming algorithms described in Section III, relevant data processing characteristics of the software implementation are identified. The result of this are four proposed hardware optimizations, which are described and evaluated in a design space exploration in the subsequent sections.

TABLE I
Performance of Beamforming Algorithms: Average PESQ, STOI and iSNR Scores for Interfering Source Angles Larger than 90°. Deviations of Fixed-Point Implementations Compared to Double-Precision Floating-Point (FP) Implementations are Given in Percent.

Double Precision Floating-Point			
	PESQ	STOI	iSNR
No processing	1.60	0.50	-5.92
Fixed (cardiod)	2.19	0.83	7.29
Fixed (supercardiod)	2.26	0.85	8.03
Fixed (hypercardiod)	2.10	0.80	5.19
Adaptive gain	2.48	0.90	10.61
Adaptive filter	2.49	0.91	10.73

32-bit Fixed-Point			
	PESQ (% FP)	STOI (% FP)	iSNR (% FP)
No processing	1.60 (0%)	0.50 (0%)	-5.92 (0%)
Fixed (cardiod)	2.19 (0%)	0.83 (0%)	7.29 (0%)
Fixed (supercardiod)	2.26 (0%)	0.85 (0%)	8.03 (0%)
Fixed (hypercardiod)	2.10 (0%)	0.80 (0%)	5.19 (0%)
Adaptive gain	2.48 (0%)	0.90 (0%)	10.61 (-1%)
Adaptive filter	2.49 (-1%)	0.90 (-1%)	10.72 (-1%)

24-bit Fixed-Point			
	PESQ (% FP)	STOI (% FP)	iSNR (% FP)
No processing	1.60 (0%)	0.50 (0%)	-5.92 (0%)
Fixed (cardiod)	2.19 (-1%)	0.83 (0%)	7.29 (0%)
Fixed (supercardiod)	2.26 (0%)	0.85 (0%)	8.03 (0%)
Fixed (hypercardiod)	2.10 (0%)	0.80 (0%)	5.19 (0%)
Adaptive gain	2.47 (-1%)	0.90 (-1%)	10.61 (-1%)
Adaptive filter	2.38 (-5%)	0.89 (-2%)	10.07 (-7%)

16-bit Fixed-Point			
	PESQ (% FP)	STOI (% FP)	iSNR (% FP)
No processing	1.60 (0%)	0.50 (-1%)	-5.92 (-1%)
Fixed (cardiod)	2.18 (-1%)	0.83 (0%)	7.27 (-1%)
Fixed (supercardiod)	2.26 (-1%)	0.85 (0%)	8.01 (-1%)
Fixed (hypercardiod)	2.10 (0%)	0.80 (0%)	5.17 (-1%)
Adaptive gain	2.18 (-12%)	0.90 (-8%)	7.21 (-23%)
Adaptive filter	0.58 (-77%)	0.34 (-63%)	-12.4 (-216%)

TABLE II
Architectural Features of the KAVUAKA Processor

Architectural Feature	Quantity
Architecture	2 issue-slots VLIW + 2 virtual issue-slots (X2-MODE) [24]
Pipeline depth	2
Data Width	64-bit/48-bit/32-bit/24-bit
Register file	Partitioned, 2x 32 registers
Functional Units	8 including a complex-valued MAC unit [25]
SIMD capabilities	1, 2 or 4 subwords (see Table III)
Extensions	Co-processors, serial interfaces, low latency audio interface [26]

A. *Configurable Fixed-Point Word Width and Precision*

Fixed-point arithmetic is used in hearing aids due to the limited power consumption budget and restricted silicon area resources [7]–[11]. Additionally, the fixed-point word width determines the rounding error and the computational accuracy of the algorithm. Therefore, this accuracy may also influence the quality and benefit experienced by the hearing aid user.

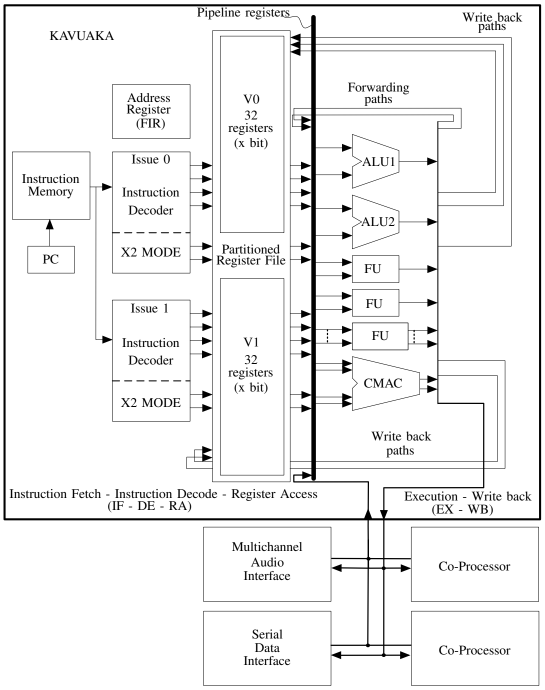

Fig. 6. Architecture of the KAVUAKA processor. The processor is divided into two pipeline stages. External co-processors and serial interfaces modules can be accessed by a bus interface.

TABLE III
DATAPATH CONFIGURATIONS WITH AND WITHOUT SIMD SUPPORT

Datapath Width	SIMD
64	2 and 4 subword modes (2x32-bit & 4x16-bit)
48	2 and 4 subword modes (2x24-bit & 4x12-bit)
32	no subwords
24	no subwords

In this work, the KAVUAKA processor is implemented with different datapath widths in order to evaluate the trade-off between power consumption, silicon area, and the performance of the beamforming algorithms.

The width of the datapath is varied between 24-bit, 32-bit, 48-bit, and 64-bit. The parallel processing mechanism SIMD (Single Instruction Multiple Data) is implemented depending on the datapath width. SIMD instructions with two and four subwords are enabled for the 48-bit and 64-bit configurations and are disabled for the 24-bit and 32-bit configurations. The width of the subwords of the SIMD modes is half and quarter as long as the width of the whole datapath. In total, four different datapath configurations are implemented within the KAVUAKA hearing aid processor. All configurations are listed in Table III.

B. Dummy Register Mechanism and Address Isolation

The beamforming algorithms consist of filter structures, which are updated on every new sampled audio data. Therefore, the number of temporary values, which have to be held within the register file, is high. Accesses to the register file may cause a lot of switching power consumption in those parts of the processor. This is especially the case for digital signal processors. These are equipped with many registers, which are accessed by multiple data buses and large address decoder networks. In order to decrease the power consumption within the register file, which accounts on average for about 45% of the KAVUAKA processor's power consumption, a register file address isolation and dummy register mechanism are implemented.

The architecture of the dummy register mechanism is shown in Fig. 7 and enhances the approach proposed in [27]. In this work, the approach is extended by making the number of dummy registers configurable during runtime as an application-specific optimization. Dummy registers are registers in the register file, which are implemented but their read and write access is enabled by a programmable dummy control register. Dummy registers can be exploited to reduce the power consumption, taking into account those temporal variables of the application program obtained during the execution from the forwarding paths. If these temporal variables are not needed any more, they are called short-lived variables and can be stored in dummy registers, not actually requiring to be written back to the register file. This mechanism decreases power consumption in the register file, since registers are not read or written. The compiler determines the optimal number of dummy registers by detecting the use of short-lived variables in the application and optimizes the allocation of dummy registers instead of real registers to decrease power consumption.

The second approach proposed to decrease the power consumption within the register file is the address isolation of the register file. Like other register files of current digital signal processors, the register file of the KAVUAKA processor has many read and write ports to feed all parallel working execution units. In this case, 4 read and 2 write ports are used to address 32 registers in each of the two partitioned register banks. These ports account for around 14% of the register file silicon area. To avoid switching activity and therefore reduce the power consumption, the last read/write address of each port is held by an additional register, if this port is temporarily not used. The address isolation mechanism is shown in Fig. 7.

C. Instruction- and Data-Level Parallelism

The parallelizability of the application is important for an efficient implementation on a digital signal processor. Digital signal processors offer data- and instruction-level parallelism on the hardware level. These processors are equipped with multiple issue-slots for processing multiple instructions at the same time or single instruction multiple data (SIMD) mechanisms to process multiple data points with the same instruction. The implementation of an algorithm on these processors is not efficient, if the available hardware mechanisms can not be used due to data dependencies. Although switching activity can be avoided for unused hardware, static power is always consumed. In this work, the parallelizability of the beamforming algorithms is evaluated.

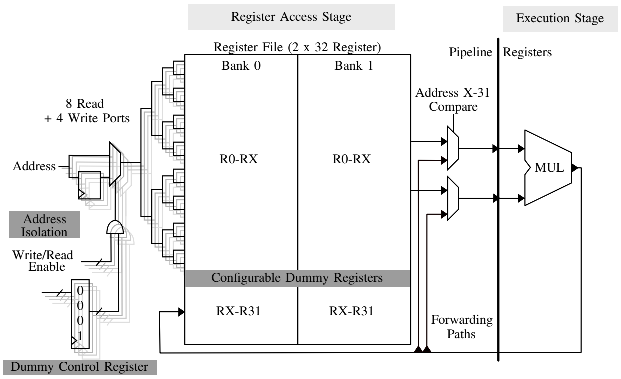

Fig. 7. Register file implementation with address isolation and dummy registers.

D. Complex Arithmetic Operations

Complex arithmetic operations require considerable computation time, if there is no hardware acceleration. A division operation is required for the adaptive gain beamformer. The division operation can be calculated using software libraries, which is comparatively slow, or it can be calculated using a hardware accelerator. This hardware acceleration offers faster computation speeds. However, the silicon area and power consumption overhead have to be considered. This trade-off analysis is included in this work for the beamforming algorithms by extending the processor with a co-processor accelerator. A software (SW) and a hardware implementation (HW) of a CORDIC (Coordinate Rotation Digital Computer) algorithm is used to calculate the division operations. The CORDIC algorithm is suitable for hearing aid applications, since beside the division operation, other hyperbolic and trigonometric mathematic operations can be computed with this software or hardware solution. A fast and accurate computation compared to other approximation algorithms is possible with reduced memory requirement compared to look-up-table interpolation algorithms.

In this work, the hardware implementation of the CORDIC algorithm is a co-processor, which is based on the architecture described in [23]. The co-processor consists of a CORDIC core, an iteration controller, an output register and an angle table as shown in Fig. 8. The CORDIC core computes the matrix multiplications using additions and shifts. This co-processor is coupled to the KAVUAKA processor as shown in Fig. 6.

VI. Evaluation Results

The silicon area of the KAVUAKA hearing aid processor architecture is determined by performing ASIC synthesis, using a TSMC $40\,\mathrm{nm}$ bulk CMOS technology excluding instruction and data memories. The static and dynamic power consumption of KAVUAKA running the beamforming algorithms is estimated using gate-level switching activity and sign-off power analysis. The minimal required operating frequency is determined and applied separately for each of the applications and processor configurations. The switching activity is recorded after all internal registers are filled with realistic data for the time taken to process 700 audio samples.

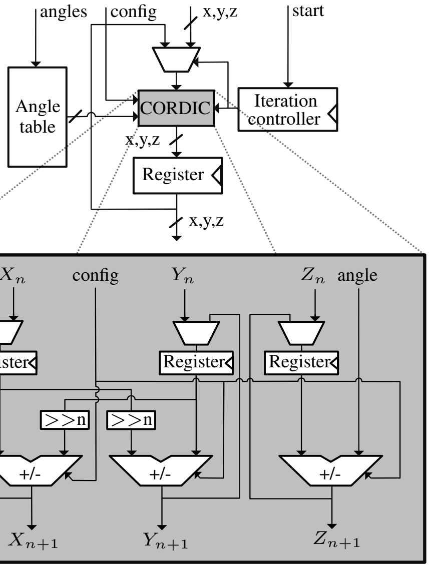

Fig. 8. Architecture of the CORDIC Co-Processor and the CORDIC core.

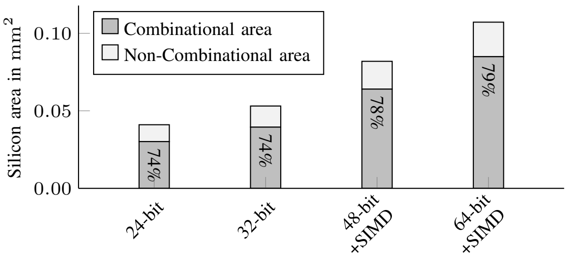

Fig. 9. Total silicon area for different datapath width and SIMD modes

A. Configurable Word Width, Precision, and Instruction- & Data-Level Parallelism

The resulting silicon area for the four datapath configurations from Table III is shown in Fig. 9. The silicon area scales with the datapath width. The smallest configuration is the 24-bit configuration without SIMD support. The 32-bit configuration is 31% larger. The 48-bit and 64-bit configurations with SIMD support are 212% and 215% larger than the 24-bit and 32-bit configurations without SIMD. Based on these values the SIMD overhead in silicon area is between 12% and 15%. The overhead primarily increases the combinational area proportion, since it is around 78-79% for the SIMD processors as compared to 74% for the non-SIMD processors.

The static and dynamic power consumption of these four configurations running the beamforming algorithms are shown in Fig. 10. The minimal required operating frequencies for processing the beamforming algorithms, which are listed in Table IV, are used. Due to its comparatively low computing complexity, the fixed beamformer offers the lowest power consumption with a minimal value of $0.014\,\mathrm{mW}$ for the 24-bit configuration. The adaptive gain beamformer with a division computed by software library requires up to $0.624\,\mathrm{mW}$, run-

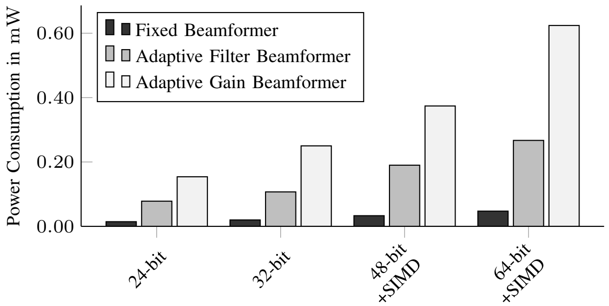

Fig. 10. Total power consumption for different datapath width and SIMD modes

TABLE IV
DYNAMIC INSTRUCTIONS PER CYCLE AND MINIMAL REQUIRED OPERATING FREQUENCIES

Algorithm	Fixed		Adaptive Filter		Adaptive Gain	
	non-SIMD	SIMD	non-SIMD	SIMD	non-SIMD	SIMD
IPC	1.86	1.80	1.93	1.94	1.87	1.95
min. MHz	0.41	0.42	2.55	2.91	7.89	10.09

ning on the 64-bit SIMD processor configuration. The power consumption of all beamforming algorithms scales linearly with the silicon area.

The parallelizability of the beamforming algorithms is evaluated using the dynamic count of the instructions per cycle (IPC) and the minimal required operating frequencies, which are listed in Table IV. The instruction level parallelism is high, since the IPC values reach almost the maximum achievable value of 2 on the two issue-slot processor. The data level parallelism provided by the SIMD mechanism is not beneficial. The filter structures of the beamformers can not be packed efficiently into the SIMD subwords due to data dependencies. This leads to higher minimal operating frequencies, due to the extra required operations to repack the subwords.

B. Dummy Register Mechanism and Address Isolation

The silicon area overhead for implementing the dummy register and address isolation mechanism within the register file are depicted in Fig. 11. Compared to the unoptimized reference register file, the silicon area increases by 2.3-3.6% for the isolation hardware. If the dummy registers are additionally added, this overhead rises to 2.4-4.4%. The power consumption for these three register file configurations, including the optimizations, is evaluated with the beamforming algorithms. The static and dynamic power consumption comparison to the unoptimized reference register file is shown in Fig. 12. The power consumption decreases by 4-10% for the address-isolated register file. With dummy registers, the power consumption in the register file can be decreased by 6-17%. The power consumption saving by the dummy register optimization depends on the utilization of dummy registers of the application. The dynamic utilization is given in Table V. These values are based on the maximum number of addressable register read and write ports per cycle, which is 12 in case of the 8 read and 4 write ports. If neither a register or a dummy register is read or written by one port, this port stays unused. If the dummy usage is high compared to the standard register usage, which is noticeably the case for the 24-bit and 32-bit adaptive filter beamformer, the reduction in power consumption increases.

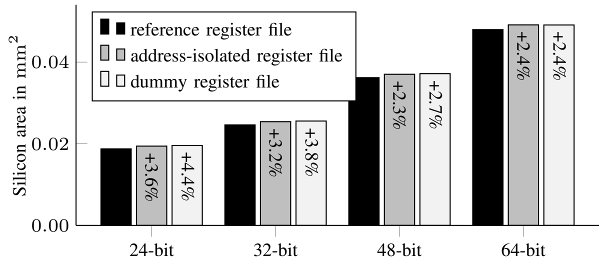

Fig. 11. Register file silicon area for the unoptimized reference register file, the address-isolated register file and the dummy register file.

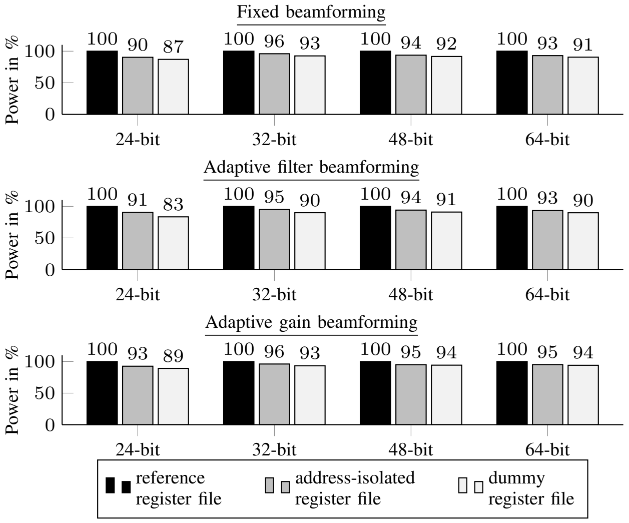

Fig. 12. Power consumption of the address-isolated register file and dummy register file compared to unoptimized register file for the fixed beamformer, the adaptive filter beamformer and the adaptive gain beamformer.

TABLE V
REGISTER AND DUMMY USAGE FOR BEAMFORMING ALGORITHMS

Fixed beamformer				
Bit width	24-bit	32-bit	48-bit	64-bit
Register usage	24.19 %	21.22 %	23.99 %	27.03 %
Dummy usage	5.24 %	8.22 %	7.74 %	4.71 %
No usage	70.57 %	70.56 %	68.27 %	68.26 %

Adaptive filter beamformer				
Bit width	24-bit	32-bit	48-bit	64-bit
Register usage	20.91 %	20.74 %	25.04%	25.97 %
Dummy usage	17.04 %	17.37 %	11.82 %	11.52 %
No usage	62.05 %	61.89 %	63.14 %	62.51 %

Adaptive gain beamformer				
Bit width	24-bit	32-bit	48-bit	64-bit
Register usage	21.31 %	20.95 %	17.58 %	16.49 %
Dummy usage	7.12 %	7.22 %	8.76 %	9.01 %
No usage	71.52 %	71.83 %	73.66 %	74.50 %

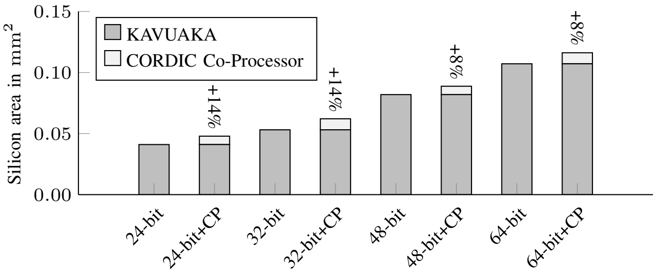

Fig. 13. CORDIC co-processor silicon area compared to total silicon area.

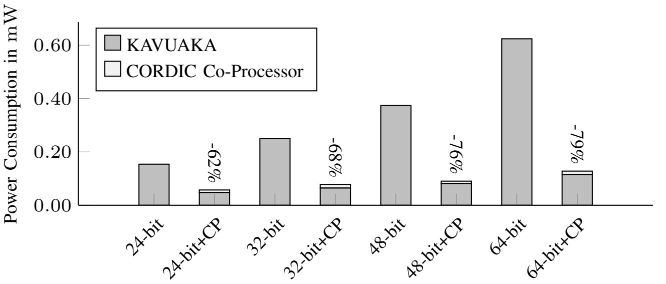

Fig. 14. Total power consumption with and without co-processor (CP).

TABLE VI
MINIMAL OPERATING CLOCK FREQUENCIES IN MHZ FOR CONFIGURATIONS WITH AND WITHOUT A CO-PROCESSOR (CP)

24-bit	24-bit +CP	32-bit	32-bit +CP	48-bit	48-bit +CP	64-bit	64-bit +CP
6.2	1.9 -69%	7.9	2.0 -74%	7.9	1.7 -78%	10.1	1.9 -81%

C. Complex Arithmetic Operations

The silicon area of the CORDIC co-processor implemented using the TSMC $40\,\mathrm{nm}$ technology is $0.0068\,\mathrm{mm}^2$ for a 24-bit datapath width/resolution and $0.0090\,\mathrm{mm}^2$ for a 32-bit datapath width/resolution. The silicon area overhead caused by attaching this co-processor to the KAVUAKA processor with different datapath width is shown in Fig. 13. The overhead is smaller for the 48-bit and 64-bit, since the co-processor is implemented without SIMD support in this case.

Due to the hardware acceleration using the CORDIC co-processor, the minimal operating frequency required for the adaptive gain beamformer can be decreased by up to 81%, as shown in Table VI. The resulting static and dynamic power consumption with and without the co-processor is given in Fig. 14. Despite the increased silicon area of the co-processor, the total power consumption drops by 62-79%, when using the hardware acceleration instead of the software computation of the division operation for the adaptive gain beamformer.

D. Overall Evaluation Including Algorithm Performance

The performance of the beamforming algorithms is compared with the hardware-related requirements, which are the silicon area and static & dynamic power consumption, in a design space exploration. All application-specific hardware optimizations, which are described in Section V, are implemented in 24 different configurations of the KAVUAKA processor. The total power consumption of these configurations, running the beamforming algorithms from Section III, is compared with the silicon area requirements shown in Fig. 15. The fixed beamformer (black star marks) consumes the lowest power, even when implemented on the 64-bit configuration. The hardware accelerated adaptive gain beamformer (blue + marks) consumes less power than the adaptive filter beamformer (red x marks) and the software-based adaptive gain beamformer. The silicon area scales with the datapath width and increases with the configurations equipped with the hardware division co-processor. The most efficient hardware and algorithm combination with the minimal area-power product is the 24-bit processor without a co-processor and without SIMD support running the fixed beamformer.

In order to evaluate the algorithm performance, the PESQ, STOI, and iSNR scores are plotted over the total power consumption in Fig. 16. The fixed beamformer offers the lowest power consumption, but the algorithm performance is lower compared to the adaptive beamformers. Besides the 24-bit adaptive filter beamformer, both adaptive beamformers offer almost identical performance. When prioritizing the performance against the silicon area, the best combination is the 24-bit adaptive gain beamformer with the hardware co-processor and the dummy register file.

VII. CONCLUSION

In this paper, a power consumption evaluation of hearing aid ASIP optimizations based on the performance of different fixed and adaptive beamforming algorithms was presented. Twenty four different optimized processor configurations and three beamforming algorithm combinations are evaluated. The results show that one of the best combinations in terms of the algorithm performance is the adaptive gain beamformer running on a 24-bit processor without SIMD, with a division co-processor and optimized register file, including address isolation and dummy registers. This combination offers the highest possible algorithm performance, while the power consumption is by a factor of 11x smaller than using the unoptimized 64-bit processor with SIMD and the same algorithm. The smallest possible combination, with reduced performance (i.e., fixed beamformer with -2dB iSNR), requires 2.2x times less silicon area than the largest combination.

REFERENCES

[1] J. M. Kates, *Digital Hearing Aids*. Plural publishing, 2008.

[2] V. Hamacher, J. Chalupper *et al.*, "Signal Processing in High-End Hearing Aids: State of the Art, Challenges, and Future Trends," *EURASIP Journal on Applied Signal Processing*, vol. 2005, pp. 2915–2929, 2005.

[3] S. Doclo, W. Kellermann *et al.*, "Multichannel signal enhancement algorithms for assisted listening devices: Exploiting spatial diversity using multiple microphones," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 18–30, 2015.

[4] S. Doclo, S. Gannot *et al.*, "Acoustic beamforming for hearing aid applications," *Handbook on Array Processing and Sensor Networks, Wiley IEEE Press*, 2010.

[5] C.-I. Kim, H. Soeleman, and K. Roy, "Ultra-Low-Power DLMS Adaptive Filter for Hearing Aid Applications," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 6, pp. 1058–1067, 2003.

[6] L. S. Nielsen and J. Sparsø, "Designing Asynchronous Circuits for Low Power: An IFIR Filter Bank for a Digital Hearing Aid," *Proceedings of the IEEE*, vol. 87, no. 2, pp. 268–281, 1999.

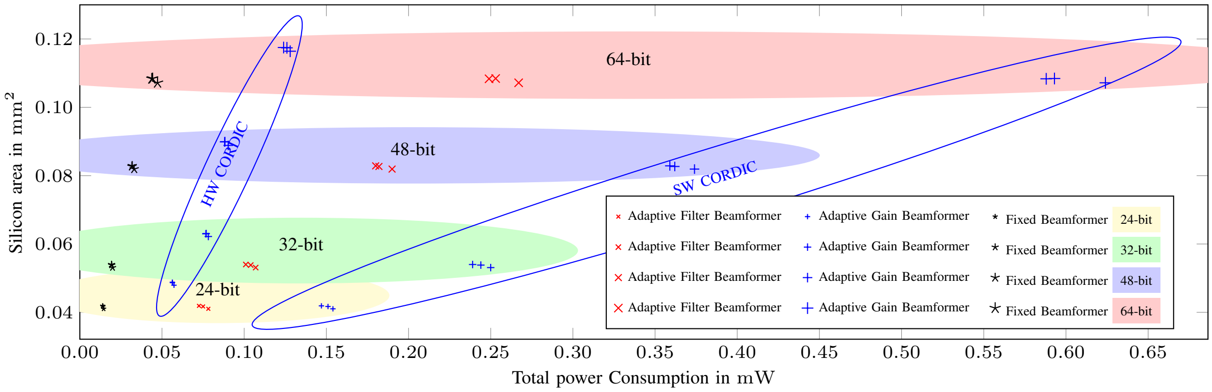

Fig. 15. Total power consumption compared to silicon area requirement. The colored ellipses hold all processor configurations with 24-bit, 32-bit, 48-bit and 64-bit with the reference, dummy and isolation register file implementation.

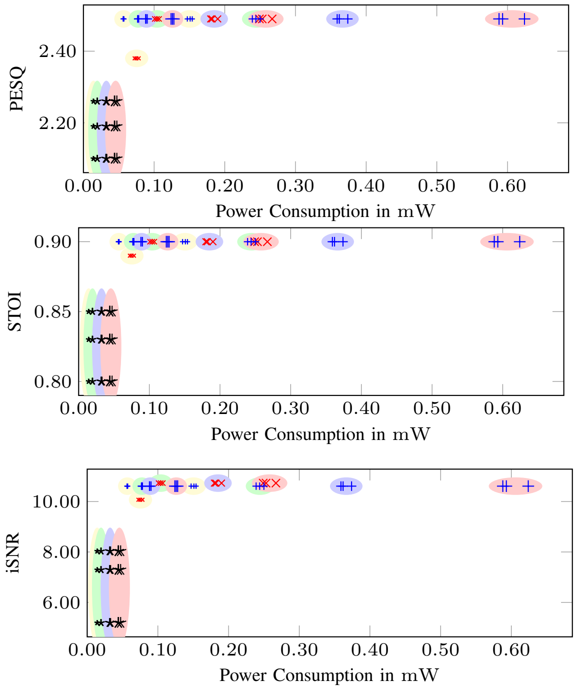

Fig. 16. Total power consumption compared to the average values of perceptual evaluation of speech quality (PESQ), short time objective intelligibility index (STOI) and intelligibility-weighted signal-to-noise ratio (iSNR) for interfering source angles larger than 90°. The legend and color schemes are the same as in Fig. 15.

[7] T. Stetzler, N. Magotra *et al.*, “Low Power Real-Time Programmable DSP Development Platform for Digital Hearing Aids,” in *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, vol. 4. IEEE, 1999, pp. 2339–2342.

[8] C. Arm, J.-M. Masgonty *et al.*, “Low-power quad-mac 170 uw/mhz 1.0 v macgic dsp core,” in *Solid-State Circuits Conference, 2006. ESSCIRC 2006. Proc. of the 32nd European.* IEEE, 2006, pp. 223–226.

[9] H. Neuteboom, B. M. Kup, and M. Janssens, “A DSP-Based Hearing Instrument IC,” *IEEE Journal of Solid-State Circuits*, vol. 32, no. 11, pp. 1790–1806, 1997.

[10] O. Semiconductor, “Solving the Hearing Aid Platform Puzzle,” Tech. Rep., 2014.

[11] H. Roeven, J. Coninx, and M. Ade, “CoolFlux DSP-The embedded ultra low power C-programmable DSP core,” in *Proc. Intl. Signal Proc. Conf. (GSPx).* Citeseer, 2004.

[12] C. Chen, L. Chen *et al.*, “A 1V, 1.1 mW mixed-signal hearing aid SoC in 0.13 μm CMOS process,” in *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on.* IEEE, 2016, pp. 225–228.

[13] G. W. Elko, *Differential Microphone Arrays.* Boston, MA: Springer US, 2004, pp. 11–65.

[14] F.-L. Luo, J. Yang *et al.*, “Adaptive Null-Forming Scheme in Digital Hearing Aids,” *Signal Processing, IEEE Transactions on*, vol. 50, no. 7, pp. 1583–1590, 2002.

[15] S. Haykin, *Adaptive Filter Theory*, ser. Always learning. Pearson, 2014.

[16] R. M. Baumgärtel, H. Hu *et al.*, “Comparing Binaural Pre-processing Strategies II: Speech Intelligibility of Bilateral Cochlear Implant Users,” *Trends in hearing*, vol. 19, 2015.

[17] H. Kayser, S. D. Ewert *et al.*, “Database of multichannel in-ear and behind-the-ear head-related and binaural room impulse responses,” *EURASIP J. on Advances in Signal Processing*, vol. 2009, p. 6, 2009.

[18] K. Wagener, T. Brand, and B. Kollmeier, “Entwicklung und Evaluation eines Satztests für die deutsche Sprache III: Evaluation des Oldenburger Satztests [Development and Evaluation of a Sentence Test for the German Language III: Evaluation of the Oldenburg Sentence Test],” *Zeitschrift für Audiologie 1999c; 38: 86*, vol. 95, 1999.

[19] R. P. ITU-T, “862-perceptual evaluation of speech quality (PESQ): an objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs,” *International Telecommunication Union-Telecommunication Standardisation Sector*, 2001.

[20] C. H. Taal, R. C. Hendriks *et al.*, “An evaluation of objective measures for intelligibility prediction of time-frequency weighted noisy speech,” *The J. of the Acoustical Society of America*, vol. 130, no. 5, pp. 3013–3027, 2011.

[21] J. Greenberg, P. Peterson, and P. Zurek, “Intelligibility-weighted measures of speech-to-interference ratio and speech system performance,” *The J. of the Acoustical Society of America*, vol. 94, no. 5, pp. 3009–3010, 1993.

[22] A. Frias Velazquez, R. d. J. Romero-Troncoso *et al.*, “Exact LMS learning curve analysis under finite word length effects,” in *20th Annual Workshop on Circuits, Systems and Signal Processing (ProRISC 2009).* STW Technology Foundation, 2009, pp. 218–222.

[23] J. Hartig, L. Gerlach *et al.*, “Customizing a VLIW-SIMD Application-Specific Instruction-Set Processor for Hearing Aid Devices,” in *Signal Process. Syst. (SiPS), 2014 IEEE Workshop on.* IEEE, 2014, pp. 1–6.

[24] G. Payá-Vayá, J. Martín-Langerwerf *et al.*, “Instruction Merging to Increase Parallelism in VLIW Architectures,” in *System-on-Chip, 2009. SOC 2009. International Symposium on.* IEEE, 2009, pp. 143–146.

[25] L. Gerlach, G. Payá-Vayá, and H. Blume, “An Area Efficient Real- and Complex-Valued Multiply-Accumulate SIMD Unit for Digital Signal Processors,” in *Signal Process. Systems (SiPS), 2015 IEEE Workshop on.* IEEE, 2015, pp. 1–6.

[26] ——, “A Low Latency Multichannel Audio Interface for Low Power SIMD Digital Signal Processors,” in *ICT.OPEN2016, ISBN: 978-90-73461-932*, 2016.

[27] G. Payá-Vayá, J. Martin-Langerwerf *et al.*, “A Forwarding-sensitive Instruction Scheduling Approach to Reduce Register File Constraints in VLIW Architectures,” in *Application-specific Systems Architectures and Processors (ASAP), 2010 21st IEEE International Conference on.* IEEE, 2010, pp. 151–158.