Low-Power Architectural Trade-Offs in a VLSI Implementation of an Adaptive Hearing Aid Algorithm

Felix Buergin
Integrated Systems
Laboratory (IIS), ETH Zurich
8092 Zurich, Switzerland
buergin@iis.ee.ethz.ch

Flavio Carbognani
Integrated Systems
Laboratory (IIS), ETH Zurich
8092 Zurich, Switzerland
carbo@iis.ee.ethz.ch

Martin Hediger,
Hektor Meier,
Robert Meyer-Piening,
Rafael Santschi
Students at IIS

ABSTRACT

This paper analyzes the power-area trade-off of functionally equivalent architectural implementations of a speech enhancement algorithm for hearing aids. Gate-level simulations and measurements show that an optimum degree of resource sharing (0.60 mW in a 0.25 µm CMOS process) is more energy-efficient than both the fully time-multiplexed (1.42 mW) and the isomorphic architecture (1.54 mW), without overly large area overhead (0.77 mm^2 against 0.43 mm^2 and 4.31 mm^2, respectively).

Categories and Subject Descriptors: B.7.1 [Integrated Circuits]: Types and Design Styles — Algorithms implemented in hardware, VLSI (very large scale integration)

General Terms: Algorithms, Design, Measurement

Keywords: Hearing aids, low-power architecture, speech enhancement

1. INTRODUCTION

Hearing aids are a typical example of a portable device. They include digital signal processing algorithms, which demand considerable computing power. Yet, miniature pill-sized batteries store a small amount of energy, limiting their lifetime. On the other hand, chip size is another important issue. Consequently, it is mandatory to employ low-power design and circuit techniques without neglecting their impact on area occupation.

Hearing impairment is often accompanied with reduced frequency selectivity which leads to a decreased speech intelligibility in noisy environments. One possibility to alleviate this deficiency is the spectral sharpening for speech enhancement proposed in [6]: based on adaptive filtering, the important frequency contributions for intelligibility (formants) in the speech are identified and accentuated.

Due to area constraints, such algorithms are usually implemented in totally time-multiplexed architectures, in which multiple operations are scheduled to run on a few processing units [4]. The opposite pole is represented by the fully-parallel or isomorphic architectures (Par. 7.2.4 in [1], [3]).

DAC 2006, July 24–28, 2006, San Francisco, California, USA.

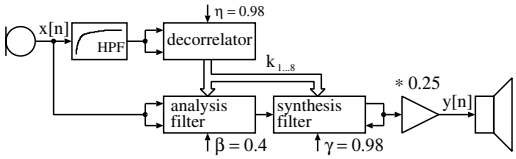

Figure 1: The spectral sharpening filter.

This work discusses the trade-off between chip size and power consumption in an ASIC implementation of the speech enhancement algorithm. It points out that while time-multiplexed architectures introduce a large control overhead and spoil signal correlations [1], isomorphic architectures tend to suffer from glitch propagation. The optimum efficiency is therefore expected somewhere between these two extremes. Several implementations of the algorithm, differing only in the degree of resource sharing (iterative decomposition) are investigated aiming at power-efficiency maximization. Two of them have been integrated on silicon, while the others have been simulated at gate-level.

This paper is organized as follows: Sec. 2 gives an overview of the algorithm. In Sec. 3 the realized architectures are presented. Additionally, different number representations are discussed regarding power efficiency. The synthesis and power simulation flow is explained in Sec. 4. Sec. 5 discusses gate-level simulation results, which are compared with measurements in Sec. 6. The paper is concluded with Sec. 7.

2. SPEECH ENHANCEMENT

In Fig. 1 the block diagram of the algorithm [6] is depicted. The input signal $x[n]$ is split in two paths. The lower is the actual signal path. By means of the analysis $(1 - A(z/\beta)$, FIR) and the synthesis $([1 - A(z/\gamma)]^{-1}$, IIR) filter, the formants are amplified. In the upper path the coefficients $k_1, ..., k_8$ are calculated in the adaptive decorrelator, whose actual filter output is not used. A first-order high-pass filter preceeds this block in order to compensate for the spectral roll-off in speech. As suggested in [6], a filter order of eight has been chosen for the three main building blocks (decorrelator, analysis and synthesis filter).

The realization of the latter two stages including details of saturation and truncation is shown in Fig. 2. The feedback structure of the synthesis is given by the required recursion. All data are represented in two's complement fixed point format. Bit widths are given by `NI.NF`, where `NI` and `NF` denote

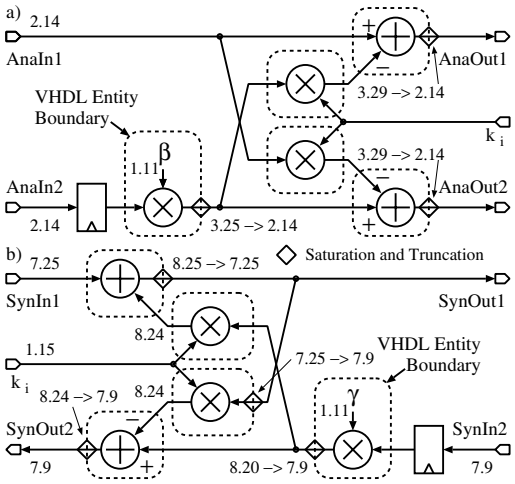

Figure 2: Structure of the analysis (a) and synthesis (b) filter stages.

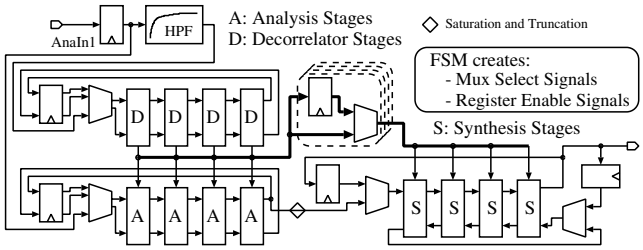

Figure 3: ID2 structure.

	Isom.	ID2	ID4	ID8	ID41
f_{clk} [kHz]	22.05	44.1	88.2	176.4	904.05
No. of adders 16b	$8 \cdot 1$	$4 \cdot 1$	$2 \cdot 1$	1	0
No. of adders 32b	$8 \cdot 9$	$4 \cdot 9$	$2 \cdot 9$	9	2
No. of mult. 16b	$8 \cdot 12$	$4 \cdot 12$	$2 \cdot 12$	12	2
No. of mult. 32b	$8 \cdot 1$	$4 \cdot 1$	$2 \cdot 1$	1	1
No. of shifters	8	4	2	1	1

Table 1: Overview of the different architectures.

the number of integer and fractional bits, respectively [2]. To ease power simulation and analysis, each operation is located in its own VHDL entity.

Four decorrelator stages are lined up in Fig. 6. One coefficient is calculated per stage. The division has been replaced by a barrel shifter, whose introduced error is barely audible.

3. ARCHITECTURAL IMPLEMENTATIONS

3.1 The Isomorphic Architecture

Generally, an isomorphic architecture is characterized as follows [3]:

1. each operation is executed by its own hardware unit,
2. no control is needed, since the data dependency graph and the block diagram are isomorphic, and
3. clock and sample rate (here 22.05 kHz) are the same.

For the considered algorithm, each main block requires to implement eight lattice stages. The blocks are connected according to Fig. 1. The number of registers is minimal and corresponds to the lower bound of memory requirement for any implementation of this algorithm.

3.2 Iterative Decomposition

Iteratively decomposing a function means to break it up into subtasks which are then executed in a step-by-step fashion on fewer computational units (resource sharing).

In order to compare the different implementations, we define the "ID-index" x of a design as a measure of the level of iterative decomposition: x is the ratio between the clock frequencies of the considered design and the isomorphic design. Each design is named as IDx.

3.3 Implemented Architectural Versions

The following architectures with identical functionality have been devised:

ID1: By definition this implementation corresponds to the isomorphic architecture (Sec. 3.1). Certainly, this architecture occupies the largest area.

ID2: Operating at doubled frequency, this design does with only four lattice stages, through which the samples are looped twice. Fig. 3 shows the implementation.

ID4: The working frequency is doubled again and the hardware effort is halved, excluding small overhead.

ID8: It represents the highest degree of natural iterative decomposition. This means that each block consists of one stage, through which data are fed eight times. The overhead is still small.

ID41: This architecture goes further in that the idea of iterative decomposition is applied to the one stage of ID8. Only three multipliers, two adders and one barrel shifter are allocated to perform all necessary operations. As a consequence, it takes 41 clock cycles to compute one output sample from the spectral sharpening filter.

Designs ID2 and ID41 have been integrated on silicon, ID4 and ID8 can be implemented similarly as ID2. Main characteristics of the designs are given in Tab. 1.

3.4 Number Representation

Additions can be implemented more easily in the two's complement (2sC) format, while multiplications are simpler in sign & maginitude (S&M). Conversion in both directions can be realized by XOR gates and an incrementer. Sometimes it is possible to postpone this increment to later adders by making use of a full-adder in the LSB. This hybrid format (2sC and S&M) is more power-efficient in a regular FIR filter implemented with one MAC unit, especially if most of the data is small in magnitude [7]. This is often the case in hearing aids.

To evaluate this approach, one analysis stage has been re-implemented accordingly. The lower right subtractor in Fig. 2a needs one additional incrementer, because both inputs require conversion. Simulations (see Sec. 4) of this stage have shown an unexpected result: the hybrid exhibits around 17 % more power consumption than the pure 2sC implementation. Investigations have shown that this incrementer destroys the potential of the hybrid. Moreover, glitches emanating from the converter, which also contains logic to catch exceptions (± 0, e. g.), cause an increased activity in downstream logic, which results in a larger dissipation of the hybrid stage.

Additional simulations of the multiplier in the path from top-left to bottom-right in the analysis stage (Fig. 2a) have

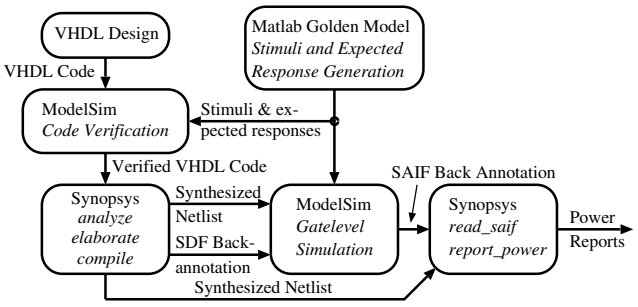

Figure 4: Power estimation design flow using Synopsys Power Compiler (simplified).

been performed. They have shown that the S&M multiplier (without conversion), which is fed with glitch-free stimuli, consumes about 8.8 µW. Simulating the same S&M multiplier and stimuli (in 2sC) together with the conversion before, a total power consumption of around 16.3 µW results. Finally, the corresponding 2sC multiplier has been simulated, it shows a consumption of 10.7 µW. Assuming that the dissipation of the conversion block itself is negligible, one can state that the effect of the glitches on the subsequent multiplier nullifies the potential of a hybrid implementation. For this reason, all subsequent designs have been implemented in two's complement format. The focus for the remaining part of this work will be on the trade-off between area and power consumption.

4. SYNTHESIS AND SIMULATIONS

All designs have been coded in VHDL. As functional verification and simulation tool, ModelSim has been employed. A synthesis with Synopsys Design Compiler, targeting a 0.25 µm CMOS process (VDD = 2.5 V), resulted in a Verilog netlist and a SDF (Standard Delay Format) file. This file contains the cell delays and an estimation of interconnect delays. This netlist, together with the SDF file, is used to perform a post-synthesis simulation. This design and power-estimation flow is shown in Fig. 4. The result is a SAIF (Switching Activity Interchange Format) back-annotation file containing the activity factor α to calculate the switching power (Par. 3.1.3 in [1]):

$$P_{sw,\ tot.} = \sum_{i=1}^{\#\ nodes} \alpha_i C_i V_{dd}^2 f_{clk}$$

Again, node capacitances C_i are estimated by Synopsys.

5. AREA AND POWER COMPARISON

The resulting power and area figures based on post-synthesis simulations are given in Fig. 5. For designs ID1 to ID8 the scaling of area with each step of decomposition can be seen. As the number of memory cells does not differ significantly in every decomposition step and the additional control area overhead is small, the total area almost halves with each degree of decomposition. It is impossible, however, to extrapolate the area and power figures from ID4 to ID8 in order to get an estimation of ID41, because the modular structure of the algorithm has to be broken up completely.

The power dissipation of the isomorphic structure is relatively high compared to the other designs. Although ID1 does not need any control unit, it consumes more power than all other considered designs. Regarding power consumption,

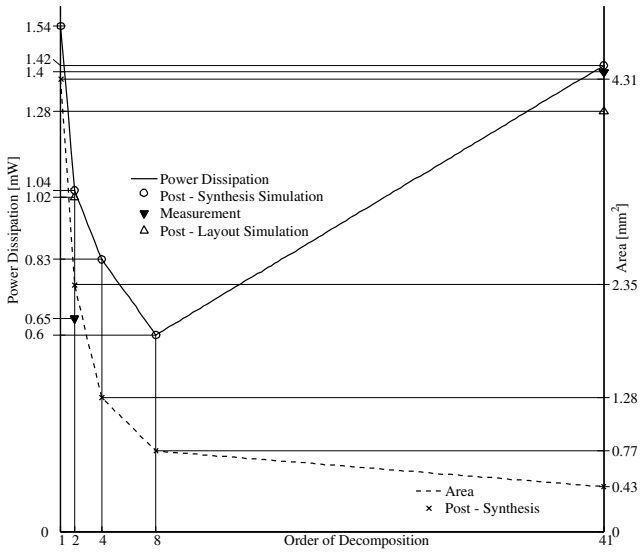

Figure 5: Power dissipation and area used by designs ID1, 2, 4, 8 and ID 41.

ID8 represents the minimum. It occupies a larger area than the best case, ID41. Note that no reasonable design could be devised between ID8 and ID41.

To understand the large power consumption of the isomorphic architecture, toggle activities in four subsequent decorrelator stages have been analyzed (Fig. 6). The annotated activities have been read out of a SAIF file produced by a gate-level simulation (Fig 4). The displayed activities are the averaged values of each input port of the corresponding arithmetic unit. The following observations made for Fig. 6 hold accordingly for ID1 (eight stages):

- Arithmetic units placed behind at least one other unit without any register in between suffer from glitches. The longer the chain, the more glitches may occur.
- Downstream stages suffer more from glitches than (e. g.) the very first stage.
- A unit placed directly after a register exhibits a low switching activity. This can be seen (e. g.) in the multiplier in all pathes from bottom-left to top-right.

By decomposing the algorithm, registers are introduced after each iteration step. These additional registers catch glitches. Since the toggle activity α is normalized to the clock frequency, different levels of iterative decomposition can be compared by means of Fig. 6. This means that the approximate activity of ID4 corresponds to the activity of the first two stages in the figure. ID8 corresponds to the first stage only. One can state that the more stages are put in sequence the higher the average toggle activity gets. In rough approximation, if the switching activity is dominant and the internal loads are more or less equal, the dissipation is proportional to the average activity α.

In designs ID1 to ID8, the frequency doubles while the number of computational units halves at each degree of decomposition. It follows that, as the control dissipation is negligible, the ratio between the consumption of any two designs approximately equals the ratio between the corresponding switching activities of the computational units. These results have been substantiated by verifying the linear relationship between toggle activity and power dissipation in designs ID1 to ID8.

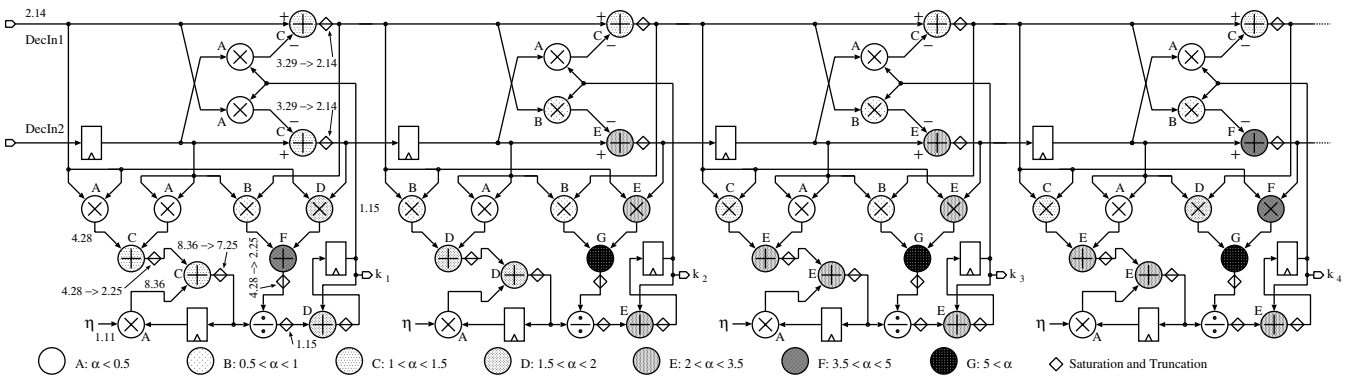

Figure 6: Activity of the decorrelator block in ID2. Reduced to the first two stages, the approximate activity of ID4 can be read from the activity patterns. For ID8, only stage one is representative.

Design ID41 does not profit from its high degree of decomposition. It suffers from a dominating overhead due to more complex control and many large multiplexers in front of the arithmetic units and registers. Additionally, because of its high degree of resource sharing, switching activity in the few arithmetic units is increased since signal correlations are destroyed completely (Par. 7.2.4 in [1]).

To prevent glitches from propagating along arithmetic chains in less decomposed designs, one might be tempted to add pipeline registers. Yet, this is no option as this algorithm contains a feedback structure (see Fig. 2b).

6. MEASUREMENTS ON SILICON

Placement & Routing has been performed only for designs ID2 and ID41. They have been integrated on the same chip. ID1 could not be integrated due to area limitations. Both designs have separate power nets to facilitate accurate power measurements. A separately powered multiplexer has been included because both designs share the same IO pads. ID2 occupies 7.72 mm^2 on silicon, whereas ID41 takes 0.78 mm^2.

Measurement results at 2.5 V can be found in Fig. 5. ID2 dissipates 0.65 mW and ID41 1.4 mW. Note that these figures are not directly comparable with the simulation results in the same figure, since those are based on post-synthesis netlists (no clock tree, no back-annotation of parasitics etc.). For these two designs, the flow presented in Fig. 4 has been enhanced. After place and route, parasitics have been extracted. Together with this information the netlists and SDFs after placement & routing have been simulated (post-layout gate-level). This resulted in an estimated power dissipation of 1.02 mW and 1.28 mW, respectively (see Fig. 5).

It can be observed that the agreement of measurement and post-layout simulation in design ID41 is much better than in ID2. A possible explanation is the overestimation of glitches in such simulations. Due to the high degree of decomposition in ID41, arithmetic units therein are mostly followed by registers, which catch glitches. In ID2, chains of arithmetic units are more frequent, hence, glitches may occur more often and are overestimated frequently. However, as predicted by simulations, ID2 is still more power-efficient than ID41.

Experience has shown that glitches are not modelled accurately enough to get reliable power estimations. In particular, in gate-level simulations each glitch is assumed to show a full swing from ground to VDD and back. Sometimes, on silicon, a hazard does not materialize as a glitch or has a little swing only. This leads to overestimation of glitches and power consumption. Future simulators should incorporate more precise techniques to estimate spurious switching activity. The work by Ruiz de Clavijo et al. [5] points into this direction.

7. CONCLUSIONS

Different levels of iterative decomposition of a speech enhancement algorithm for hearing aids have been implemented and analyzed in terms of power consumption and area. The design with the highest possible degree of decomposition without destroying the regularity of the algorithm is the most power-efficient solution. Yet, an area penalty has to be paid compared to the smallest design (fully decomposed). Measurements on real silicon have approximately confirmed the simulation results, although glitches seem to be overestimated. Additionally, it has been shown that for this type of algorithm, the two's complement number representation is more efficient than a hybrid format.

8. ADDITIONAL AUTHORS

Additional authors: Hubert Kaeslin (Design Center at ETH Zurich, email: `kaeslin@ee.ethz.ch`), Norbert Felber (IIS, email: `felber@iis.ee.ethz.ch`), and Wolfgang Fichtner (IIS, email: `fw@iis.ee.ethz.ch`)

9. REFERENCES

[1] A. P. Chandrakasan and R. W. Brodersen. *Low Power Digital CMOS Design*. Kluwer Academic Publishers, Boston, Massachusetts, 1995.

[2] S. Grassi. *Optimized Implementation of Speech Processing Algorithms*. University of Neuchâtel, Switzerland, 1998.

[3] H. Kaeslin. *VLSI I Lecture Notes on Architectures of Very Large Scale Integration Circuits*. Microelectronics Design Center, Swiss Federal Institute of Technology, Zurich, Switzerland, 2005.

[4] P. Mosch et al. A 660-μW 50-Mops 1-V DSP for a hearing aid chip set. *IEEE Journal on Solid-State Circuits*, 35(11):1705–1712, November 2000.

[5] P. Ruiz-de Clavijo et al. Logic-level fast current simulation for digital cmos circuits. In *Proceedings 15th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS 2005)*, pages 425–435, September 2005.

[6] A. Schaub and P. Straub. Spectral sharpening for speech enhancement/noise reduction. In *IEEE Acoustics, Speech and Signal Processing (ICASSP-91)*, pages 993–996, April 1991.

[7] J. Wassner, H. Kaeslin, N. Felber, and W. Fichtner. Waveform coding for low-power digital filtering of speech data. *IEEE Transactions on Signal Processing*, 51(6):1656–1661, June 2003.