PSEUDEC: IMPLEMENTATION OF THE COMPUTATION-INTENSIVE PARTRAN FUNCTIONALITY USING A DEDICATED ON-LINE CORDIC CO-PROCESSOR

Finn T. Møller, Jack B. Andersen+, Hans R. Jensen, Ole Olsen and Flemming K. Fink

Institute for Electronic Systems
Aalborg University
9220 Aalborg Øst
Denmark
e-mail: {ftm, hrje90, oo, fkf}@kom.auc.dk

+Grundfos Electronics
8850 Bjerringbro
Denmark
e-mail: dkgrfh3x@ibmmail.com

ABSTRACT

This paper describes PSEUDEC, a dedicated co-processor and the rationale behind it's design.

The final goal of our work is to present a single chip solution with low power consumption for an advanced digital hearing aid based on a parameterized transformation of speech (PARTRAN).

Characterization of the constituent parts of the PARTRAN algorithm shows, that it is well suited for implementation on a heterogeneous architecture. The design strategy used identifies a subset suited for implementation on dedicated hardware, with computational complexity roughly equivalent to the performance of a standard 10 MIPS DSP.

The subset of PARTRAN implemented by PSEUDEC performs PSEUdo DEComposition of a 12th order LPC polynomial. An adapted algorithm displays improved dynamic range compared to a conventional solution suited for DSP's, calculating the amplitude spectrum rather than the power spectrum.

Highly pipelined CORDIC-units optimized for the application replaces complex multiplication, trigonometric operations (for e^{jw}) and square root (for $|a| = \sqrt{a_r^2 + a_i^2}$), exploiting the power of CORDIC operations in advanced DSP algorithms.

PSEUDEC uses redundant data representation and bit-serial arithmetic, most significant digit first (ON-LINE arithmetic) for efficient implementation of operators and for efficient (inter-operator) communication. The inherent nature of ON-LINE arithmetic and the operators used allows for fast and efficient implementation even when using ordinary standard cells.

1. INTRODUCTION

Demanding DSP applications has requested solutions of constantly increasing complexity, an increase in computational capabilities in terms of basic operations available and in the number of operations performed. Further, modern DSP systems involve a wide spectrum of DSP algorithms, each of which is most efficiently implemented on different architectures. This has driven the evolution of implementation tools and strategies from soft-coded single processor solutions to heterogeneous architectural solutions, where a complete DSP system is implemented on a mixture of programmable signal processors as well as on application specific data paths and array architectures.

In the design strategy exemplified in this article by the design of the PSEUDEC co-processor, the overall research effort comprises:

1. Application and algorithm development and characterization. This includes identifying and describing the constituent part(s) of the algorithm.
2. Selecting relevant parts of the algorithm for hardware implementation based on their characteristics regarding computational complexity (isolating bottlenecks), regularity and operator constraints.
3. Selecting a suitable data representation. Bit serial and redundant arithmetic are interesting candidates for many algorithm parts, due to their virtues with respect to high throughput and local communication.
4. Implementing the algorithm(s). For many regular algorithms, a direct mapping into an architecture leaves a high degree of control to the designer, often yielding better results than high level synthesis. Using structural hierarchical net-list descriptions in for instance VHDL or C++ as in the implementation of PSEUDEC, enables compact description of parameterized basic blocks. Reusing basic blocks and employing standard cell place and route tools results in a fast - and to a large degree process independent - implementation, leaving more time for refinements of the algorithm, architecture and arithmetic.

2. APPLICATION

In the PARTRAN (PARametric speech TRANsformation [1] [2]) application, a speech signal $X(z)$ is split into an LPC

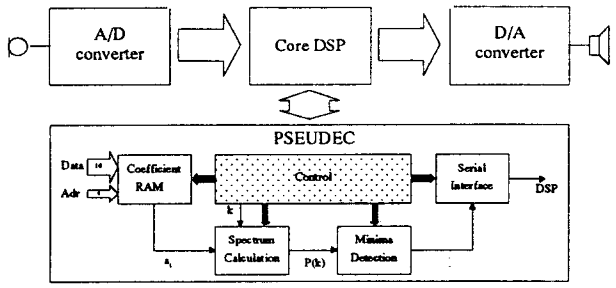

Figure 1 : *The heterogeneous PARTRAN architecture consists of a core-DSP and the PSEUDEC co-processor.*

model $1/A(z)$ of the speech organs and a residual signal $E(z)$:

$$\frac{1}{A(z)}E(z) = X(z)$$

The 12th order LPC model $1/A(z)$ is decomposed into a set of parallel second order bandpass filter sections. Each second order section can then be subjected to arbitrary transformations based on its center frequency, bandwidth and power - resulting in a transformed model $A'(z)$. Finally a transformed speech signal $X'(z)$ is re-synthesized:

$$X'(z) = \frac{1}{A'(z)}E(z)$$

The decomposition into second order sections involves finding the roots of the transfer function $1/A(z)$, corresponding to the formant frequencies of the speech signal $X(z)$. Due to numerical problems of traditional root-finding algorithms, a complete inverse spectrum $A(e^{j\omega})$ is calculated instead, and the complex root pairs of the polynomial are identified as the minima. This approach is not only numerically stable in fixed point arithmetic, but also sorts out "insignificant" roots [1]. The major drawback is however an increase in computational complexity, making the decomposition account for half the complexity of the total PARTRAN algorithm [3]. The pseudo decomposition is very regular, which suggests an implementation as a separate dedicated co-processor, and the remaining parts of the PARTRAN algorithm on a standard/core DSP, thus creating the heterogeneous architecture outlined in figure 1.

3. ALGORITHM

The pseudo decomposition (PSEUDEC) algorithm is subdivided into two specific tasks: Calculating the inverse LPC-spectrum $A(e^{j\omega})$, and detecting the minima.

Taking the LPC coefficients $\{a_0, \cdots, a_{12}\}$ as input, the polynomium $A(e^{j\omega})$ is calculated directly using Horner's scheme:

$$\begin{aligned} A(e^{j\omega}) &= \sum_{i=0}^{N-1} a_i \cdot e^{-j\omega i} \\ &= ((a_N e^{-j\omega} + a_{N-1})e^{-j\omega} + \cdots + a_1)e^{-j\omega} + a_0 \end{aligned}$$

This results in a better dynamic range compared to the original programmed version, which calculated the power spectrum $A(e^{j\omega}) \times A^*(e^{j\omega})$ for better computational efficiency on a DSP [1].

```
# Calculate Spectrum $P(k)$ in 200 points
For $k = 0$ to 199
    # Complex spectrum $A(k) = \sum_{i=0}^{12} a_i\, e^{j\omega_k i}$
    $A(k, 13) = 0$
    For $i = 12$ downto 0
        $A(k,i) = A(k,i+1)e^{j\pi k/200} + a_i$
    # Calculate magnitude
    $P(k) = |A(k,0)|$

    # Detect minima
    $P'(k) = P(k) - P(k-1)$
    $P'(k-1) = P(k-1) - P(k-2)$
    If $P'(k-1) < 0$ and $P'(k) > 0$
        Output($P(k), P(k-2), k$)
```

Algorithm 1 : *The PSEUDEC algorithm calculates a spectrum and detects minima.*

The minima are detected by numerical differentiation of the amplitude spectrum $P(e^{j\omega}) = |A(e^{j\omega})|$. A minima is reached, where the slope $P'(e^{j\omega})$ changes sign from negative to positive. The minima and the corresponding spectrum values are then output from PSEUDEC.

4. ARITHMETIC

The design rationale for algorithm rewriting was the use of ON-LINE arithmetic, which is the term used for digit serial calculations performed most significant digit first (MSDF) [4] [5] [6].

ON-LINE arithmetic requires a redundant data representation [7] [8]. PSEUDEC uses the signed binary representation [9] - radix 2, digit set $x_n \in \{-1, 0, 1\}$, $x_n = (xp_n - xn_n)$, where $xp_n, xn_n \in \{0, 1\}$. Using two bits per digit leaves a "spare" digit representation, which is used as a "space-digit" to separate consecutive digit serial words. This enables an efficient distributed control scheme to be used (time is running from left to right):

$$s\ s\ s\ x_0\ x_1\ x_2\ .\ .\ .\ .\ x_{N-2}\ x_{N-1}\ s\ s\ s$$

A data-word $X = \sum_{n=0}^{N-1} x_n 2^{-n}$ is separated from other data-words by inserting "space-digits" ($sp = sn = 1$) in between, thus disallowing two one's to represent arithmetical zero. This makes the interfacing of arithmetic operators simple and efficient.

Ongoing research show, that ON-LINE arithmetic can implement a larger set of the operations used in advanced DSP algorithms than found in commercial DSP's or feasible in traditional digit serial LSDF arithmetic [10]. CORDIC operations [11] [12] [13] are especially interesting for PSEUDEC, since complex multiplication by $e^{j\omega}$ and magnitude $|A(e^{j\omega})|$ can be calculated efficiently using ON-LINE CORDIC rotation and vectoring respectively.

CORDIC operations are performed as a series of "micro-

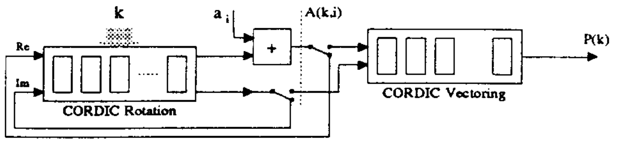

Figure 2 : *Architecture of the spectrum calculator part of PSEUDEC*

rotations" of a vector (x, y) by an angle $\xi_n(k) \cdot atan(2^{-(n-1)})$, $\xi_n(k) \in \{-1, 1\}$ each consisting of shift/add operations only. In digit serial mode, the number of clock cycles (ie. *time*) decides the precision of the vector (x, y), while the precision of the angle $\omega(k) = \sum_n \xi_n(k) \cdot atan(2^{-(n-1)})$ is decided by the number of rotations performed (ie. *area*).

5. ARCHITECTURE

Figure 2 shows the architecture for the spectrum calculation derived from the PSEUDEC algorithm. All the necessary rotations are mapped onto a single CORDIC rotation unit, which is then optimized to reduce area and latency [3]. While latency at the digit level is not an important issue for a PSEUDEC application, reducing latency to allow only a single word to circulate in the CORDIC rotation, will result in the simplest control scheme.

"Truncating" the rotation sacrifices some precision on the rotation angle, but reduces the number of micro-rotations from 24 at full precision to 8 necessary for 200 point resolution. This results in a major reduction in area, while maintaining full functionality and precision of the original algorithm.

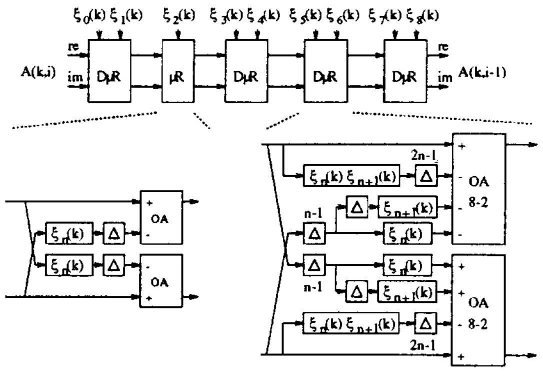

Figure 3 : *The CORDIC rotation* $A(k, i+1) = A(k, i) \cdot K\, e^{-j\omega(k)}$ *, is performed by single* μ*-rotation (left) and double* μ*-rotation (right)*

Figure 3 shows the implementation of the CORDIC Rotation algorithm:

$$A_{re(n+1)} = A_{re(n)} + \xi_n \cdot 2^{-(n-1)} \cdot A_{im(n)}$$

$$A_{im(n+1)} = A_{im(n)} - \xi_n \cdot 2^{-(n-1)} \cdot A_{re(n)}$$

To minimize latency, the micro-rotations are look-ahead transformed two-by-two so that iteration 3-8 consists of

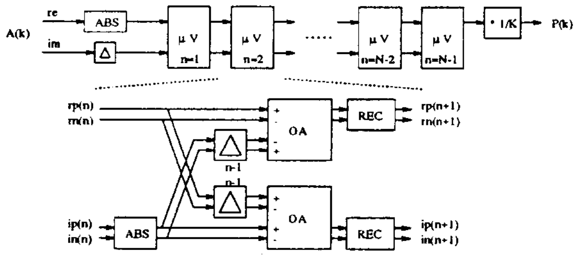

Figure 4 : *The CORDIC Vectoring algorithm implementation computing the 2-norm* $P(k) = (A_{re}(k)^2 + A_{im}(k)^2)^{\frac{1}{2}}$

"double micro-rotations". Further, latency is reduced by using some of the standard On-Line latency (δ) in the 8-2 addition instead of delay elements in the delay lines to right-shift the operands. Or in other terms using ON-LINE delays instead of physical ones.

Finally, multiplication by the inverse scaling factor K [11] [14] which normally is performed after every CORDIC operation is "retimed" outside the feedback-loop, so that the a_i's are pre-multiplied by K^{-i} before entering the calculation loop.

These optimization steps has reduced latency from more than 150 to 29 clock cycles, allowing one 24 digit word separated by 5 "space-digits" to circulate in the rotation unit. The specifications concerning clock-frequency shown in table 1 is satisfied by:

$$f_{\text{clock}} = \frac{S \times R \times D}{T} = 75.4\ MHz \quad \text{where:}$$

$$S = 200 \text{ points/spectrum}$$

$$R = 13 \text{ rotations/point}$$

$$D = 29 \text{ digits/rotation}$$

$$T = 1 \text{ ms/spectrum}$$

The critical path of the spectrum unit consists of two full-adders and a register, making 75 MHz operation feasible.

Figure 4 shows the implementation of CORDIC vectoring algorithm calculating the 2-norm (length of vector):

$$A_{re(n+1)} = A_{re(n)} + 2^{-(n-1)} \cdot |A_{im(n)}|$$

$$A_{im(n+1)} = |A_{im(n)}| - 2^{-(n-1)} \cdot A_{re(n)}$$

This block is not fully utilized, as it performs only one calculation for each 13 rotations. This could have been multiplexed onto a single μ-vectoring unit, reducing the area of the vectoring unit from 5.1 mm^2 to approximately 2 mm^2. However, as the real part of the result contains the 2-norm output, the result is not changed after the $\frac{N}{2}$ 'th μ-vectoring iteration, meaning that the CORDIC vectoring can be terminated ("truncated") after 12 iterations as well.

Figure 5 shows the architecture for the minima detection. It should be noted, that at 1.2 mm^2 this circuit uses only 10 % of the hardware resources of the spectrum calculation. The corresponding software implementations show roughly equal computational complexity, illustrating how poorly decision making utilize a standard DSP.

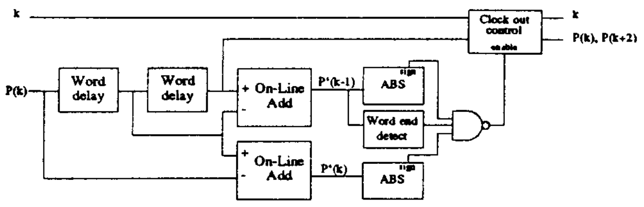

Figure 5 : *Architecture of minima detection part of PSEUDEC*

6. CONCLUSION

This article has described the implementation of a computation intensive DSP functionality targeted for a single chip solution in a heterogeneous architecture. PSEUDEC constitutes a practical implementation of a DSP algorithm using ON-LINE arithmetic throughout, delivering computational complexity equal to a state of the art DSP using a core area of 20 mm^2 1 μ CMOS. Exploiting multiplexing of the CORDIC vectoring unit and replacing large static flip-flops with smaller dynamic ones has shown to reduce the area to 12 mm^2.

The chip is implemented based on a fully synchronous design philosophy, using C++ as a hardware description language. A full functional simulation is performed in C++, before a netlist for final place and route is extracted. The chip was implemented using the Mentor GDT tools during one 5 month project [3]. The requirements of the PARTRAN application and primary specifications are shown in table 1.

The key to efficient implementation of a DSP application is a combination of algorithm rewriting and architecture transformations. ON-LINE makes the computation word-length independent of the computation precision. Precision can be optimized in individual parts of an architecture, e.g "8 bit" rotation angle precision has been combined with 12 vectoring iterations at 24 bit dynamic range.

PSEUDEC has been fabricated through EUROCHIP in a 1 μ CMOS process. The core area is 20 mm^2, reducible to 12 mm^2 by fairly straightforward optimizations. The chosen critical path simulates to a clock frequency in excess of the 75 MHz required by the specifications.

Detailed information [3] on PSEUDEC and additional papers [6] [10] describing applications of ON-LINE is available at WWW, at `http://www-i8.auc.dk/KOM/DSP/`.

External precision	16	bits
Internal precision	24	bits
Spectrum resolution (P)	200	points
Calculation time (T)	1	ms
Required clock frequency	75	MHz
Core area $4.51 \times 4.35 =$	19.6	mm^2
Chip area $5.40 \times 5.40 =$	29.1	mm^2
# of Transistors	70830	

Table 1 : *Main specifications of the PSEUDEC Co-processor*

7. REFERENCES

[1] Kjeld Hermansen, Flemming K. Fink, and Uwe Hartmann. Parametric Transformation of Speech Signals. In *Proceedings of the ESCA Workshop, KTH, Stockholm*, 1993.

[2] Kjeld Hermansen, Flemming K. Fink, Uwe Hartmann, and V. Moss-Hansen. Hearing Aids for Profoundly Deaf People Based on a New Parametric Consept. In *Proceedings of the 1993 Workshop on Applications of Signal Processing to Audio and Acoustics, New Palz, USA*, 1993.

[3] Hans Roelsgaard Jensen and Finn Thøger Møller. PSEUDEC - The Formant Frequency Co-Processor. 9. semester thesis, Aalborg University, January 1994.

[4] Miloš D. Ercegovac and Thomas Lang. On-Line Arithmetic, A Design Methodology and Applications in Digital Signal Processing. VLSI Signal Processing, III, 1988.

[5] Mary Jane Irwin and Robert Michael Owens. A Case for Digit Serial VLSI Signal Processors. *Journal of VLSI Signal Processing*, (1), 1990.

[6] Jack Andersen, Finn Møller, and Ole Olsen. Pratical and Educational Aspects of On-Line Arithmetic. In *5th EUROCHIP Workshop on VLSI Design Training, Dresden, Germany*, October 1994.

[7] Algirdas Avizienis. Signed-digit Number Representations for Fast Parallel Arithmetic. *IRE Transactions on Electronic Computers*, C-10, 1961.

[8] Alain Guyot, Yvan Herreros, and Jean-Michel Muller. JANUS, an ON-LINE Multiplier/divider for manipulating large numbers. In *Proceedings of 9th Symposium on Computer Arithmetic*, pages 106–111, 1989.

[9] Behrooz Parhami. Generalized Signed-Digit Number Systems: A Unifying Framework for Redundant Number Representations. *IEEE Transactions on Computers*, January 1990.

[10] Jack Andersen, Anders Færgemand Nielsen, and Ole Olsen. A Systolic ON-LINE Non-restoring Division Scheme. In *Proceedings of 27th Hawaii International Conference on System Sciences*, January 1994.

[11] Jack E. Volder. The CORDIC Trigonometric Computing Technique. *IRE Transactions on Electronic Computing*, EC-8:330–334, 1959.

[12] J. S. Walther. A unified algorithm for elementary functions. In *Proceedings of the Spring Joint Computer Conference*, pages 379–385, 1971.

[13] Yu Hen Hu. CORDIC-Based VLSI Architectures for Digital Signal Processing. *IEEE Signal Processing Magazine*, July 1992.

[14] Haixiang Lin and Henk J. Sips. ON-LINE CORDIC Algorithms. In *Proceedings of 9th Symposium on Computer Arithmetic*, pages 26–33, 1989.