A 100 MHz PIPELINED RLS ADAPTIVE FILTER *

Kalavai J. Raghunath † and Keshab K. Parhi
Department of Electrical Engineering
University of Minnesota, Minneapolis, MN-55455

ABSTRACT

Recently, a new pipelinable PSTAR-RLS algorithm was developed. It was shown to be an effective alternative to the QRD-RLS algorithm when high-speeds are required. Using folding technique, a 4-tap PSTAR-RLS algorithm was implemented on a single VLSI chip. All the operations in the chip are bit-level pipelined. With a 1.2μ CMOS technology this chip is expected to run at 100 MHz. Redundant number system based arithmetic operators were used for performance advantage. Apart from a wafer scale implementation, this is the first ever single chip ASIC implementation of a RLS adaptive filter.

I INTRODUCTION

Recursive least squares (RLS) based adaptive filters are used in applications such as channel equalization, voiceband modems, digital mobile radio, beamforming, speech and image processing. The QRD-RLS algorithm [1],[2] is the most promising algorithm since it is known to have very good numerical properties and can be mapped on to a systolic array. The speed (or sample rate) of the QRD-RLS algorithm is, however, limited by the recursive equations in its cells. A new STAR-RLS algorithm [3],[4] was recently developed which can be used for high-speed applications, for example, in communications, magnetic recording, image processing etc. This algorithm uses *scaled tangent rotations (STAR)* instead of the Givens rotations which are normally used. These rotations are so designed that look-ahead, [5], can be applied with little increase in hardware complexity. The pipelined version of the algorithm is referred to as PSTAR-RLS [6],[4]. The PSTAR-RLS algorithm can be pipelined to operate at very high speeds. The STAR-RLS algorithm [3] and PSTAR-RLS have lower complexity and about half the inter-cell communication as compared with the QRD-RLS algorithm. The performance of the STAR-RLS algorithm is similar to that of the QRD-RLS [3],[6].

In adaptive filtering applications, the LMS algorithm, [2], is more commonly used than the RLS algorithm, in spite of RLS having a faster convergence. The large computational requirement of the RLS algorithm is its main drawback. It is possible to implement the LMS algorithm on a single CMOS chip with the state of art technology [7]. This makes the LMS algorithm very attractive. It is difficult to implement a RLS adaptive filter on a single chip. There was only one attempt made to design an RLS type of filter chip to the best of out knowledge [8]. But, here a wafer scale integration was needed to implement an RLS type of filter for adaptive nulling. This design used 96 cordic cells which were implemented in a wafer 16 square inches in area. This wafer was implemented in 2μm technology and the maximum clock speed was 12 MHz.

The aim of our paper is to demonstrate practicality of the PSTAR-RLS architecture and the feasibility of implementing RLS adaptive filters on a single chip. The design of a 4-tap PSTAR-RLS CMOS chip is presented. Using folding techniques [9], the area required to implement the chip is reduced to fit in a single chip. The architecture is bit-level pipelined and is expected to operate at 100MHz. Redundant number system based arithmetic modules are used in this design.

II CHIP ARCHITECTURE

In this section we describe the design of the architecture and the timing aspects.

2.1 Folded Architecture

A 4-tap PSTAR-RLS systolic array would have 4 boundary cells and 10 internal cells (see [3],[6],[4]). We design a *composite* cell which can act as a boundary cell as well as an internal cell. The mode bit is β (0 for boundary cell, 1 for internal cell). The composite cell requires 4 multiplier/divider operators while the original internal cell and boundary cell required only 3. This small hardware overhead, however, simplifies the design to a great extent. The PSTAR-RLS systolic array using such composite cells is shown in Figure 1. Using the transformation technique of folding, [9], we fold the systolic architecture by 14 times to operate on a single composite cell. Figure 1 shows the folding order (or the ordered folding set, [9]), of the cells. The systolic array in Figure 1 is pipelined along the cutsets shown with 5 delays each. This is necessary to be able to efficiently pipeline the folded architecture in a fine-grain manner. The cells are folded row-wise from top to bottom.

*This research was done at University of Minnesota and supported by the office of Naval Research under contract number N00014-91-J-1008.

†Presently at AT&T Microelectronics, Murray Hill, NJ 07974.

This is the simplest folding set and it should be noted that other folding sets could be used leading to different architectures. The folding set was so chosen as to use minimal hardware overhead.

The folded hardware will process the data 14 times slower or in other words the input is 14 slow. There is hence a speed loss due to folding. However, folding by a factor of 14 scales the number of delays by 14 times. Thus, in the recursive loops we have 70 delays instead of 5 delays. The recursive loops can be pipelined at a finer level now. Now we can have a clock rate that is 14 times higher, which compensates for the speed loss due to folding. Hence, there is no loss in speed due to folding. This is a general result and can be used for any algorithm. To the best of our knowledge this result has not been exploited before.

In the final chip circuit the available delays are moved around using retiming [10] so that all operators are bit-level pipelined. The maximum propagation delay is kept below that of a bit adder at all parts in the circuit. We have used 14-slow delays [9] in the circuit wherever possible leading to savings in number of delays.

2.2 Wordlength and Arithmetic

We used fixed-point arithmetic since the arithmetic operators would be easier to design. The dynamic range of the quantities is also not so high as to warrant a floating-point representation. In [11], a finite-precision analysis was done and that knowledge can be used to determine the wordlengths to be used. In the design of this chip, we decided not to use rounding since that would increase the delay in the arithmetic operators. A direct truncation is instead used. This would mean that more bits would be needed than that is suggested by the analysis in [11]. Hence, simulations with fixed-point arithmetic were carried out to decide the word-lengths to be used. It was found that 12 bits for cell contents and outputs, and 8 bits for rotation parameters are sufficient. The performance is almost indistinguishable as compared to the infinite precision results.

We use redundant number system arithmetic in this chip because of the low computation delay. Adders are carry-free and hence we need only one level of pipeline to make it bit-level pipelined. This reduces the number of delay latches required. Since redundant representation requires twice the number of bits to represent, they can be costly if long lines have to be routed. Hence a hybrid representation is used [12]. Internal to arithmetic operations (such as divider and multiplier), the representation is redundant but the outputs and inputs are in binary. This makes the routing easier, but, the results in redundant representation need to be converted to binary form at the outputs.

Conventional dividers have a delay proportional to the square of the wordlength. The divider used in this chip is a non-restoring redundant divider [12]. The delay here is $O(n \log n)$, where n is the wordlength. We used a redundant arithmetic based Booth encoded tree multiplier. The partial products here are added in a tree fashion giving a computation delay of $O(\log n)$.

2.3 Timing

Since the architecture is pipelined at bit-level, the registers (or delay latches) account for a major part of the chip area. We also need a register that can be useful for high clock rates. We have hence used the true single phase registers that were proposed in [13]. This register is compact needing only 11 transistors and is designed for high-speeds. Only one clock line needs to be routed since these are single-phase registers. The two main concerns are clock buffering and race-through due to clock-skew. The clock lines have high loads and hence the load is distributed using multiple-level buffering. The problem of race-through is avoided using reverse direction clock routing [14],[13].

The reverse clock distribution cannot directly be used in cases where there are recursive or feedback loops. In such cases a dummy register is used. The dummy register is an extra register which does not form a part of the logic. The clock line is so routed that only the dummy buffer is affected by the skew. This results in a race through but does not affect the logic.

2.4 Floor-plan and clock-routing

The floor-plan for the chip is as shown in Figure 2. The hardware modules are arranged in the form of three column. The different hardware modules (the divider, multipiers) have comparable widths and so such an arrangement becomes feasible. The GND and Vdd lines for all modules run vertically on the sides. Hence, such a column arrangement also makes it easier to route the GND and Vdd lines from top to bottom. The smaller units are designed to fill in the available spaces. Since most the control lines go to the reset switch and the data mux, these are kept close with the control unit in between.

The routing of the clock has to be in the direction opposite the data flow. The routing of the main clock line is as shown in Figure 3. The load on the main buffers is reduced wherever possible by branching through other buffers. The clock line branch with maximum delay goes through 2 main primary buffers and a small buffer. The estimated delay on the clock line is about 5ns. The maximum propagation delay for any combinational part of the chip is about 4.5ns. This gives us an operating frequency of 100MHz. This estimate is based on the assumption that there is no overlap between the clock delay and the propagation delay. However, the combinational portion of the chip starts computation even before the clock line reaches the end of the network. Thus, we expect the speed of the chip to be higher than 100MHz. The features of the chip are summarized in Table 1 and the layout of the chip is shown in Figure 3. This full-custom chip was designed using MAGIC tools.

Technology	1.2μ CMOS
Number of Transistors	127,000
Power Supply	5V
Die Size	6.8x8.9mm^2
Active Area	6.0x8.3mm^2
Projected Speed	100-150 MHz
Architecture	Bit-parallel, Bit level pipelined

Table 1: STAR chip summary

III CONCLUSIONS

A new STAR-RLS adaptive filtering algorithm is implemented on a VLSI CMOS chip. This is the first ever single chip implementation of RLS adaptive filter (not counting the wafer-scale chip of Rader [8]). It demonstrates the practicality of the pipelined architectures developed earlier. The chip is expected to run at speeds in excess of 100MHz.

The chip uses fixed point arithmetic with 12 bit and 8 bit words. Innovative redundant arithmetic based modules were used in this design. The chip is programmable to the extent that the forgetting factor can be chosen by the user and it can be used for filters of order less than or equal to 4.

References

[1] W. M. Gentleman and H. T. Kung, "Matrix triangularization by systolic arrays," *Proc. SPIE, Real Time Signal Processing IV*, vol. 298, pp. 298–303, 1981.

[2] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, NJ: Prentice Hall, 1986.

[3] K. J. Raghunath and K. K. Parhi, "High-speed RLS using scaled tangent rotations (STAR)," *Proc. of IEEE Intl. Symp. on Circuits and Systems (ISCAS-93)*, pp. 1959–1962, May 1993.

[4] K. J. Raghunath, "High-Speed RLS adaptive filters," *Ph.D. Thesis, University of Minnesota*, November 1994.

[5] K. K. Parhi and D. G. Messerschmitt, "Pipeline interleaving and parallelism in recusive digital filters- part I: Pipelining using scattered look-ahead and decomposition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 1118–1134, July 1989.

[6] K. J. Raghunath and K. K. Parhi, "Pipelined implementation of high-speed STAR-RLS adaptive filters," *Proc. SPIE, Advanced Signal Processing ALgorithms, Architectures, and Implementations IV*, vol. 2027, 1993.

[7] S. Maginot, F. Balestro, C. Loanblanq, P. Senn, and J. Palicot, "A general-purpose high speed equalizer," *IEEE Journal of Solid-State Circuits*, pp. 209–216, March 1991.

[8] C. M. Rader, "Wafer-scale integration of large systolic array for adaptive nulling," *Lincoln Laboratory Journal*, vol. 4, no. 1, pp. 3–28, 1991.

[9] K. K. Parhi, C. Y. Wang, and A. P. Brown, "Synthesis of control circuits in folded pipelined DSP architectures," *IEEE Journal of Solid-State Circuits*, pp. 29–43, January 1992.

[10] C. E. Leiserson, F. Rose, and J. Saxe, "Optimizing synchronous circuits by retiming," *Proc. of the third Caltech Conference on VLSI*, pp. 87–116, March 1983.

[11] K. J. Raghunath and K. K. Parhi, "Fixed and floating point error analysis of QRD-RLS and STAR-RLS adaptive filters," *Proc. of IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 81–84, April 1994.

[12] H. R. Srinivas and K. K. Parhi, "High-speed VLSI arithmetic processor architectures using hybrid number representation," *Journal of VLSI Signal Processing*, vol. 4, pp. 177–198, 1992.

[13] J. Yuan and C. Svensson, "High-speed CMOS circuit technique," *IEEE Journal of Solid-State Circuits*, pp. 62–70, February 1989.

[14] M. Hatamian and K. K. Parhi, "An 85MHz 4th order programmable IIR digital filter chip," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 175–183, February 1992.

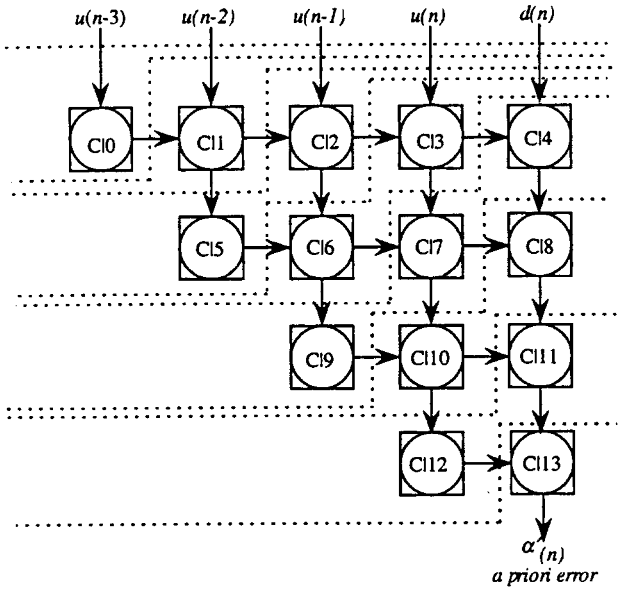

Figure 1: Systolic array for 4-tap PSTAR-RLS adaptive filter (folding set also shown).

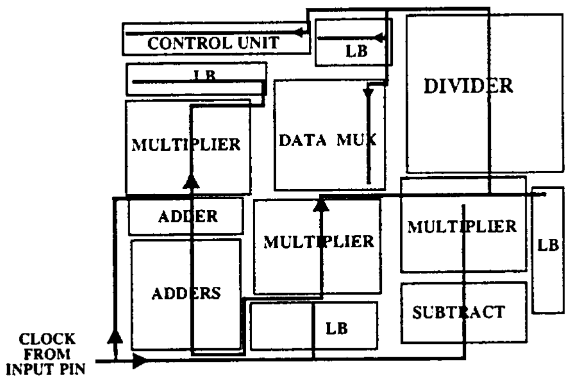

Figure 2: Chip layout and clock routing (LB implies a latch bank).

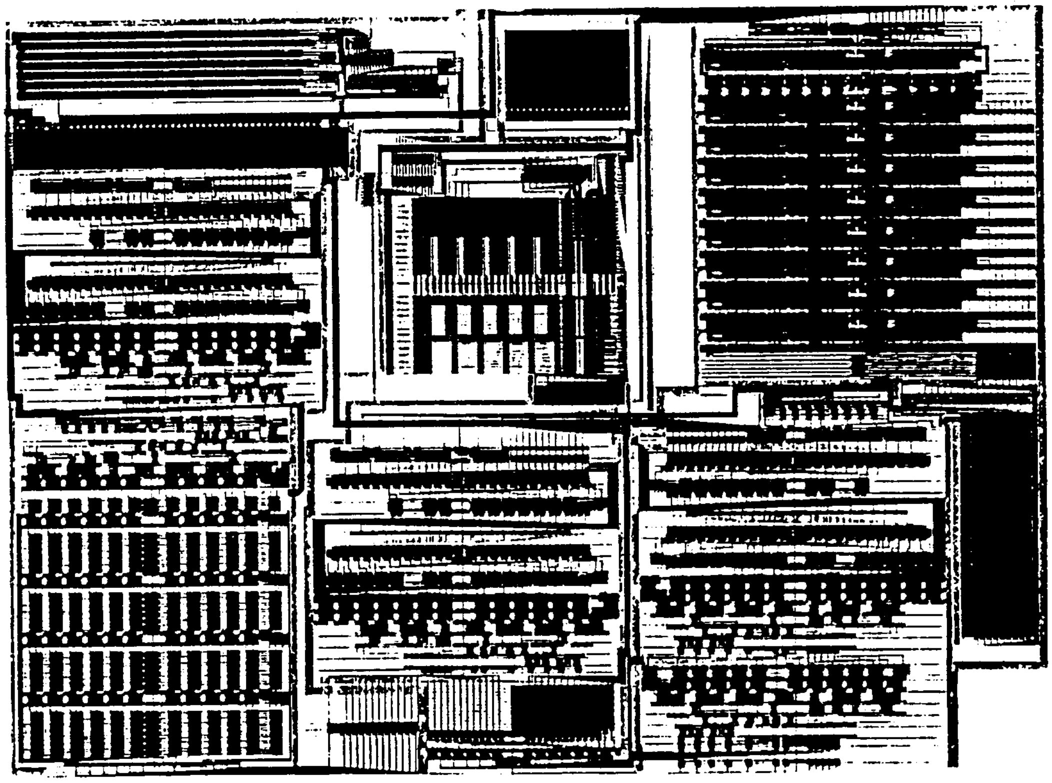

Figure 3: CHIP LAYOUT