A SYSTOLIC VLSI IMPLEMENTATION OF KALMAN-FILTER-BASED ALGORITHMS FOR SIGNAL RECONSTRUCTION

Daniel Massicotte

Department of Electrical Engineering
Université du Québec à Trois-Rivières
C.P. 500, Trois-Rivières, Québec, Canada, G9A 5H7

ABSTRACT

The problem of improving the performance of the implementation in VLSI technology of Kalman-based algorithms for signal reconstruction in real time is discussed. A systolic approach is proposed to develop architecture expressly for this specific application. Implemented algorithms are based on the steady-state version of the Kalman filter, which performs for a broad field of specific applications, but the use of a co-processor for the Kalman gain is allowed. We show that the autoregressive model of Kalman filtering is particularly adapted to parallel processing and is well suited for implementation. Although intended to improve signal reconstruction, other applications where a similar autoregressive model of Kalman filtering is required are allowed. The performance of the systolic architecture is validated by comparison with Motorola's general-purpose DSP56002 digital signal for real-world spectrometric signal reconstruction.

1. INTRODUCTION

Signal reconstruction is a very common inverse problem in such fields as telecommunications, metrology, biomedical engineering, seismology and spectrometry [1], [2], [5]-[7]. It consists of estimating a signal x, i.e., the ideal signal, when knowing the signal $\tilde{y}$, which is related to x by a causal relationship. A disturbing noise η_n affects the rough signal $\tilde{y}$, making the problem ill posed. The discrete form of the convolution equation is:

$$\tilde{y}_n = \sum_{m=1}^{M} h_{M-m}\, x_m + \eta_n \quad \text{for } n = 1,2,3,...,N \qquad (1)$$

where h_n is the impulse function. Numerous methods for solving the previous problem lead to high computational requirements, and an application-specific integrated circuit (ASIC) specifically for signal applications becomes necessary, especially when miniaturization of the electronic devices is required.

The choice of Kalman-filter-based algorithms [2], [5], [7] is justified by their broad field of applications. Implemented algorithms are based on the steady-state version of the Kalman filter, which performs for a wide field of specific applications. Indeed, the use of a co-processor for the Kalman gain is allowed when the non-stationary version of Kalman is required. Many authors (e.g. [3]) have suggested systolic architectures, which are oriented at non-stationary versions of the Kalman-filter. When applied to a stationary model of the data $\{\tilde{y}_n\}$ for $M \geq 64$, the specialized processors based on those architectures become very expensive and are also "resistant" to the introduction of constraints in the processing. To correct this inconvenience, the proposition of a systolic architecture in very large scale integration (VLSI) makes routage and cabled control use possible, and a subsequent speed increase is expected. The localization of the connections decreases the line size and, consequently, signal propagation time; as a result, integration is safer, more efficient and easier.

In Section 2, the implemented algorithm and the systolic approach are presented to take advantage of the parallelization of the algorithm, making a compromise between the area and computation time for the design. Section 3 describes the modules of the VLSI implementation, procedures and prototyping. In Section 4, an example of reconstruction is proposed and performance evaluation is done by showing a comparison with a commercial general-purpose DSP56002 digital signal processor. The conclusion is given in Section 5.

Throughout the paper, the notation of vectors and matrices is printed in bold type for clarity, and the $v_{i/j}(m)$ is the m^{th} element of the vector $\mathbf{v}$ at time i given the data available at time j.

2. SYSTOLIC IMPLEMENTATION

The following equations sum up the basis of the implemented Kalman-filter based algorithms for signal reconstruction. Integral versions are presented in [5]:

$$\hat{\mathbf{z}}_{n/n} = \hat{\mathbf{z}}_{n/n-1} + \mathbf{k}_\infty(\mathbf{h}, \breve{\beta})\left[\tilde{y}_n - \mathbf{h}^T \hat{\mathbf{z}}_{n/n-1}\right] \qquad (2)$$

$$\hat{\mathbf{z}}_{n/n-1} = \Phi\, \hat{\mathbf{z}}_{n-1/n-1} \qquad (3)$$

where the state matrix $\mathbf{\Phi}$ is sparse. The vector of the steady-state Kalman gain $\mathbf{k}_\infty$ may be calculated in advance and depends on the vector $\mathbf{h}$ and the regularization parameter $\breve{\beta}$ chosen to minimize the reconstruction error. The dimension of the matrix and vectors of the equations (1) and (2) are $M \times M$ and $M \times 1$, respectively. If one takes into account that for physical reasons x_n can be non-negative, the following constraint is imposed on the solution

$$\hat{z}_{n/n}(m) = \begin{cases} \alpha\, \hat{z}_{n/n}(m) & \text{if } \hat{z}_{n/n}(m) < 0 \\ \hat{z}_{n/n}(m) & \text{if } \hat{z}_{n/n}(m) \geq 0 \end{cases} \quad \text{for } m = 1,2,...,M \qquad (4)$$

to obtain an improved algorithm, where $\alpha \in [0,1]$ is a parameter to be optimized empirically. The estimates $\hat{x}_n$ of x_n are extracted algebraically from the estimates of the state vector $\hat{\mathbf{z}}_{n/n}$

$$\hat{x}_n = \hat{z}_{n+d/n+d}(1) \qquad (5)$$

where d is the delay of estimation given by the fixed-lag smoothing included in the model without the addition of calculations. An iterative version of the algorithms has been developed [6] and can also be implemented.

The estimate $\hat{y}_{n+1}$ is computed in parallel with the estimate $\hat{\mathbf{z}}_{n/n}$, necessary for obtaining the data $\tilde{y}_{n+1}$ used in the next sample. The particular form of the matrix $\mathbf{\Phi}$ makes possible the following recursive equations [1]

$$\hat{z}_{n+1/n}(m-1) = f\left(\hat{z}_{n/n-1}(m) + k_{\infty}(m)\, I_n\right) \quad (6)$$

$$\hat{y}_{n+1}(m-1) = h(m-1)\,\hat{z}_{n+1/n}(m-1) + \hat{y}_{n+1}(m-2) \quad (7)$$

for $n = 1,2,3,\ldots,N$, $m = 1,2,3,\ldots,M$, and f is the non-negativity function. These equations allow properties of localized communications, regularity and recursiveness making possible the use of the systolic approach. What's more, the equations uniformity, which is the basis of the algorithm local nature, allows us to consider a systolic architecture because a processing element (PE) is connected only with its close neighbors.

In comparison with a sequential architecture, the algorithm running time can be decreased in a significant way by using a classic parallelization method called "divide-to-conquer" [8]. It consists of dividing the problem into S sub-problems, which are solved separately. Afterwards, the initial problem solution is obtained by combining the partial problem solution. When this approach is used, an improvement in running time of about one S factor can be reasonably expected in comparison with a sequential architecture. In the case where S is an integer multiple of M, let s=1,2,3,...,S then $m = s, s+S, s+2S, s+3S, \ldots, s+(M/S-1)S$ or $m = s + (r-1)S$ with $r = 1,2,3,\ldots,R$ where $R = M/S$. The uniform and recurrent equations (6) and (7) become

$$\hat{z}_{n+1/n}(s-1+(r-1)S) = f\left[\hat{z}_{n/n-1}(s+(r-1)S) + k_{\infty}(s+(r-1)S)\, I_n\right] \quad (8)$$

$$\hat{y}_{n+1}(s-1+(r-1)S) = h(s-1+(r-1)S)\,\hat{z}_{n+1/n}(s-1+(r-1)S) + \hat{y}_{n+1}(s-2+(r-1)S). \quad (9)$$

The following notation is adopted: $C \rightarrow E$, where C is a set of conditions of the indexes and E an equation. The limit conditions on the indexes are:

$$n = 1, 1 \le r \le R, 1 \le s \le S \rightarrow \hat{z}_{n/n-1}(1+(r-1)S) = 0 \quad (10)$$

$$1 \le n \le N, r = 1, s = 1 \rightarrow \hat{y}_{n+1}(s-2+(r-1)S) = 0, \rightarrow \hat{y}_{n+1}(s-1+(r-1)S) = 0, \rightarrow h(s-1+(r-1)S) = 0, \rightarrow \hat{x}_n = \hat{z}_{n+1/n}(s-1+(r-1)S) \quad (11)$$

$$1 \le n \le N, r = R, s = S \rightarrow \hat{z}_{n+1/n}(s+(r-1)S) = \hat{z}_{n+1/n}(s-1+(r-1)S), \rightarrow I_{n+1} = \tilde{y}_{n+1} - \hat{y}_{n+1} \quad (12)$$

This last condition (20) corresponds to the measurement signal of a new sample $\tilde{y}_{n+1}$ to obtain the innovation I_{n+1}. Modifications during the computation process are given by the limit conditions at the beginning of the sampling when $n = 1$, on the first elements of computation when $r = 1$, and on the last elements of computation when $s = S$.

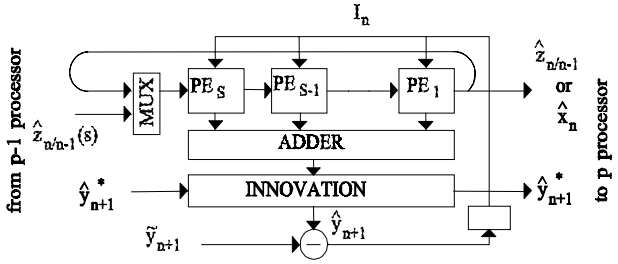

Figure 1. Linear semi-systolic architecture with ring topology (SYSKAL).

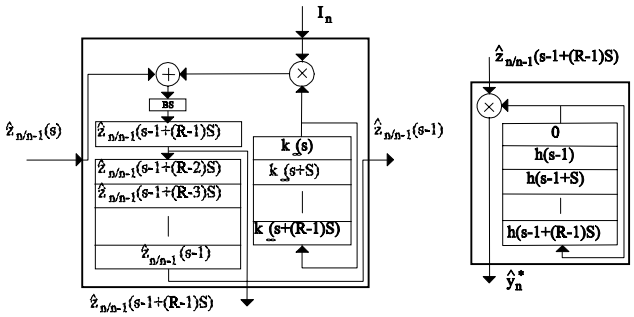

Figure 2. PE architecture: a) C_z cell and b) C_y cell.

According to these previous equations, the whole computation process can be done on a processor array, assuming that the number of processors is finite, the array topology is regular, the connections are localized, and that each processor executes a single task at each time. A ring topology of S processing element (PE) array is proposed in Fig. 1 resulting from uniform and recurrent equations (8) and (9). A common clock synchronizes all S PEs, and the data flow moves at the same time between PEs via their internal registers. This architecture is called "linear semi-systolic architecture with ring topology" (SYSKAL), according to the definition of the term semi-systolic given in [4]. The solution of the initial problem, estimate $\hat{y}_{n+1}$, is obtained by combining the solutions of the estimate S sub-problem using a fast adder. The estimate $\hat{y}_{n+1}$ yields the innovation I_{n+1} used to reconstruct the next sample $\hat{x}_{n+1}$.

Fig. 2 shows a PE architecture composed of two cells C_z and C_y: C_z cells calculate Eq. (8) and C_y cells calculate Eq. (25), producing the estimate of signal $\tilde{y}_{n+1}$ required to calculate innovation I_{n+1}.

All PE_S have the same structure, except the first, PE_1, and the last, PE_S. Limit conditions (11) and (12) are respectively satisfied by PE_1, which owns one out register less in cell C_z and one zero value register in cell C_y, and by PE_S, which differs by an added multiplexer. Each C_y cell owns one register set to zero that acts like a switch for the synchronization clock signal of the array on internal registers shifting.

3. VLSI IMPLEMENTATION

The non-negativity constraint requires multiplication by a constant α of the value $\hat{z}_n(m)$ if it is negative. To avoid one or more

additional cycles, it has been demonstrated that if α equals one of the values $0, \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}$ or 1, then an improvement of about 50% in the reconstruction can be expected [5]. This point is particularly interesting when one knows that dividing by a multiple of 2 is equivalent to a left shifting of a point and that any additional clock cycle is not needed. Note that a Barrel shifter is used that requires no clock cycle to do an asynchronous shifting. The shifter (block BS in Fig. 2a) is placed in the last pipeline stage of the appropriate M/A, then the shifting is done by a simple sign detection (positive or negative) of the M/A out result.

A surface criterion becomes an important physical constraint when VLSI implementation of the previously developed architecture is done. In fact, the number of PEs to be implemented, S, and the number of registers per PE are limited. As a result, the impulse response **h** is restricted, and consequently, the field of applications supported by the processor is reduced. The SYSKAL architecture may be modular, allowing parallel running of several SYSKAL processors (we note N_P the number of SYSKAL processors). They keep their linear structure when they are linked together in cascade. To respect the data flow through PEs of processor P, a multiplexor is available at the PE_S input to receive the data $\hat{\mathbf{z}}_{n/n-1}(s-1)$ calculated according to (8) and sent by a previous PE_1. Another multiplexor is also available at the first PE_1 output to add a register in the last pile of $\hat{\mathbf{z}}_{n/n-1}(s)$ if the SYSKAL processor is not the last one . Each processor connected in cascade can access the partial results $\hat{y}^*_{n+1}$ of the estimates $\tilde{y}_{n+1}$ to calculate the innovation I_{n+1}. If N_P=1, these results are accumulated in an internal register. In the case of N_P>1, the result of $\hat{y}^*_{n+1}$ is propagated to the next p+1 processor and each P processor obtains locally the result of innovation I_{n+1}, after N_P-1 clock cycles. Next, the innovation is sent to all PEs of P processors.

C_z cells calculate (8), the implementation of this equation, using the principle of redundant numbering as shown in Fig. 3a. During the first step, a Wallace tree executes $k_\infty(s+(r-1)S)I_n$ product, reducing the number of partial products to two quantities. Then, both quantities are injected in a carry save adder (CSA) with the $\hat{\mathbf{z}}_{n/n}$ operand to be added. The CSA reduces the three quantities to two quantities and another adder calculates the entire operation $\hat{z}_{n/n-1}(s+(r-1)S)+k_\infty(s+(r-1)S)I_n$ at the tree root level. The economy is one adder per PE.

C_y cells execute partial computations of $\hat{y}_{n+1}$. As shown in Fig. 3b, the principle of redundant numbering is still used. A Wallace tree with Booth encoding makes multiplication operations. All the tree roots produce eight partial products, which are reduced to two quantities after passing through an extension of the Wallace trees. This extension consists of two CSA layers, which reduce four quantities to two (CSA4_to_2). The two resulting quantities are added using an adder to produce $\hat{y}_{n+1}$. The economy is one adder per PE and $\log_2 S$ adders in the tree.

The SYSKAL prototype was simulated and synthesized using Mentor Graphics software tools. A parameterized model in VHDL

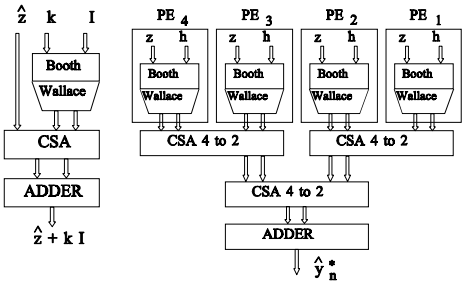

Figure 3. PE VLSI implementation: a) C_z cell and b) C_y cell.

language of SYSKAL was designed to obtain more rapidly and efficiently a design adapted to one application with special requirements like, for example, the number of bits for $\hat{\mathbf{z}}_{n/n}$, **h**, and $\mathbf{k}_\infty$, the data or the number of PEs or registers per PE. The prototype SYSKAL is composed of four PEs containing 16 registers (C=16) for $\hat{\mathbf{z}}_{n/n-1}$, $\mathbf{k}_\infty$, and **h**, and a state machine controlling these PEs according to the executed algorithm with non-negativity constraints. This state machine makes it possible to use a co-processor for the computation of the Kalman gain. The prototype use a 16-bit word length for $\tilde{y}$, $\hat{\mathbf{z}}$, $\hat{x}$ and a 8-bit word length for **h** and $\mathbf{k}_\infty$. These four PE, data word lengths, and 16 registers are used to form an acceptable compromise between speed, number of applications with dim(**h**) = 64, and silicon area. Nevertheless, the prototype allows the use of many processors connected in parallel when the dimension of a vector **h** is larger than 64 elements. The synthesis is done with a low optimization effort on the speed and area and uses a 1.2 µm CMOS technology. One multiplier used 4 557 transistors, C_z cells used 14 916 transistors, C_y cells used 7 509 transistors, the state machines used 3 596 transistors, and the calculation of innovation used 5 036 transistors, for a total of 102 208 transistors for the ASIC. The clock frequency of the prototype is 40 MHz and is principally limited by the speed of the multiplier.

4. PERFORMANCE EVALUATION

The computing performance of the proposed SYSKAL systolic architecture, using as a criterion the number of clock cycles necessary to execute the reconstruction of one sample with or without non-negativity constraint, is $(M/S+2(N_{PS}+1))$. Moreover, SYSKAL has the following latency and throughput $(M/S+2(N_{PS}+1))t_{PE}+d$ and $(M/S+2(N_{PS}+1))t_{PE}$ respectively, where t_{PE} is the slower PE running time, including fast adder if it is not pipelined; N_{PS} is the number of pipeline stages of the multipliers in cells (N_{PS} =0 if not pipelined); and d is the estimation of $\hat{x}$ delay.

The previous section has shown that a SYSKAL prototype can be implemented with fewer components than a general-purpose DSP such as the DSP56002, which contains about 200 000 transistors in 0.65 µm CMOS technology with a clock at 80 MHz. Naturally, the word lengths in both devices are not the same, but in a custom design such as ours, one can tailor the word length to the exact

requirements of the application to reduce implementation costs, energy consumption and heat dissipation, which is very important in some systems applications.

The expected computing performance of the systolic architecture has been estimated in the following way: (i) the Kalman-filter-based algorithm of reconstruction was programmed for the Motorola general-purpose DSP56002; (ii) the performance was assessed using the number of cycles required for the execution of the algorithm. The comparison was based on the assumption that the clock frequency is the same for both processors, that the number of PEs is fixed to obtain the equivalent area (S=8), that the pipeline of DSP56002 is full, and that the data of the DSP56002 are in the internal memory. Results obtained with another programmable processor dedicated to the same class of algorithms, previously developed and called DSPKAL [1], are added. All results of the comparison for S=8, N_P =1, N_{PS} =0, and N=1 are shown in Fig. 4. These results indicate that the architecture SYSKAL asked 20S times less than the DSP56002.

To illustrate the whole running of the proposed prototype architecture, an example of spectrometric signal reconstruction is shown in Fig. 5. This was obtained by treating real-world data acquired by means of the ANRITSU MV02 series optical spectrum analyzer. In this example, the computation times of reconstruction with DSP56002 and SYSKAL prototype are 5.08 ms and 65.6 µs respectively. But, if we considered the technology and the number of transistors of the DSP56002 for our prototype, we can obtain theoretically a clock frequency of 75 MHz with S=8 and the times of reconstruction become only 20 µs.

5. CONCLUSION

A new systolic architecture for signal reconstruction algorithms is proposed which we assume performs better than a general-purpose digital signal processor such as the DSP56002. In comparison with the general-purpose DSP56002, using the same criterion to execute a reconstruction of one sample is 20S times faster. Moreover, the proposed architecture can be pipelined to obtain a clock rate N_{PS} times faster than that of the DSP56002. The gain in performance, however, is obtained to the detriment of the architectural flexibility for treating different algorithms. In fact, the set of algorithms accepted by our proposed architecture is limited to the algorithms based on the autoregressive model of Kalman filter presented in [5], [6]. Although intended for signal reconstruction, this architecture can be used for other applications where a similar autoregressive model of Kalman filtering is required. The SYSKAL architecture owns properties of modularity and locality and is adaptable to the dimension of the impulse response. In that case, computation time is only affected by the longer propagation time between two neighboring processors in the ring topology. Results of performance evaluation show that the proposed architecture presents a linear-rate speed-up, i.e., it achieves an $O(S \cdot N_P)$ speed-up, in terms of processing rates where S and N_P are number of PEs and SYSKAL processors respectively.

6. REFERENCES

[1] A. Barwicz, D. Massicotte, Y. Savaria, M.-A. Santerre and R. Z. Morawski, "An Integrated Structure for Kalman-Filter-Based Measurand Reconstruction", IEEE Trans. Instrum. Meas, Vol. 43, No 3, June 1994, pp. 405-410.

[2] G. Demoment and R. Reynaud, "Fast Minimum Variance Deconvolution", IEEE Trans. ASSP, Vol. 33, No. 4, 1985, pp. 1324-1326.

[3] S.Y. Kung and J.N. Hwang, "Systolic Array Designs for Kalman Filtering", IEEE Trans. Signal Proc., Vol. 39, No. 1, January 1991, pp. 171-182.

[4] H.T.Kung, "Why systolic architecture", IEEE Computer Magazin, vol.15, 1982, pp. 37-46.

[5] D. Massicotte , R. Z. Morawski and A. Barwicz, "Incorporation of a Positivity Constraint Into a Kalman-Filter-Based Algorithm for Correction of Spectrometric Data", IEEE Trans. Instrum. Meas, Vol. 44, No 1, February 1995, pp. 2-7.

[6] D. Massicotte, R.Z. Morawski and A. Barwicz, "Kalman-Filter-Based Algorithms of Spectrometric Data Correction, Part 1: An Iterative Algorithm of Deconvolution, IEEE Trans. Instrum. Meas., Vol. , No. , June 1997.

[7] R.Z. Morawski, "Unified Approach to Measurand Reconstruction", IEEE Trans. Instrum. Meas., Vol. 43, No 2, April 1994, pp. 226-231.

[8] E. Horowitz and A. Zorat, "Divide-and-Conquer For Parallel Processing", IEEE Transactions on Computers, vol. 32, N° 6, 1983, pp. 582-585.

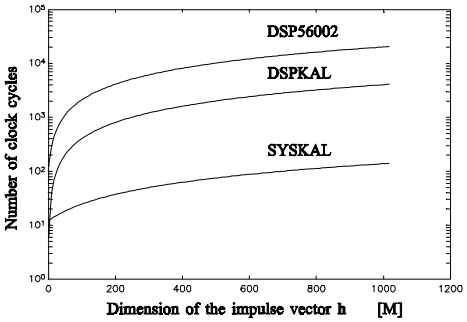

Figure 4. Number of cycles evaluation for different dimension of impulse vector **h** with S=8, N_P =1, N_{PS} =0, and N=1.

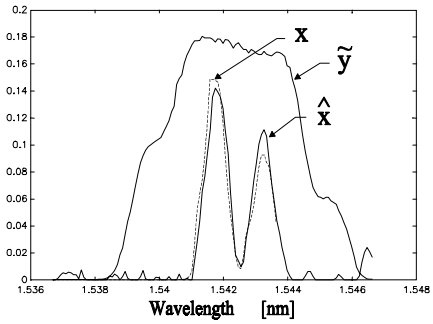

Figure 5. Example of spectrometric signal reconstruction using SYSKAL prototype with $\tilde{\beta}=3\cdot10^8$,S=4, N_P =1, N_{PS} =0, N=125,and M=76.

This work was supported by Natural Sciences and Engineering Research Council of Canada, Canadian Microelectronics Corporation, and Mentor Graphics.