Proceedings of the 6th Nordic Signal Processing Symposium - NORSIG 2004
June 9 - 11, 2004
Espoo, Finland

Pipelining of Digit-Serial Processing Elements in Recursive Digital Filters

Magnus Karlsson [1], Mark Vesterbacka [2], and Wlodek Kulesza [1]

1. University of Kalmar, SE-391 82 Kalmar, Sweden, magnus.karlsson@hik.se, wlodek.kulesza@hik.se
2. Linköping University, SE-581 83 Linköping, Sweden, markv@isy.liu.se

ABSTRACT

In this paper, performance trade-offs between throughput, and energy consumption, in implementation of recursive digital filters are presented. Digit-serial arithmetic with different degree of pipelining are used in the implementations. As a demonstration object, a bireciprocal third-order lattice wave digital filter is used. Simulations with HSPICE show that a maximum throughput is obtained using pipelined processing elements with a digit-size equal to the fractional bits in the filter coefficient. The use of non-pipelined processing elements yields minimum energy consumption. A trade-off between throughput and energy consumption can be made by pipelining only some of the processing elements.

1. INTRODUCTION

The digit-serial approach is interesting for performing a trade-off between area, throughput, and power consumption, [1,2]. In digit-serial arithmetic a number of bits is processed concurrently, i.e. the digit-size, D. If D is unity the system reduces to a bit-serial system, while for $D = W_d$, where W_d is the data word length, the system reduces to a bit-parallel system.

In recursive algorithms the critical loop limits the maximum throughput, f_{max}, [3]. f_{max} and its reciprocal the minimal sample period, T_{min}, are given as

$$f_{max} = 1/T_{min} = N/T_{op} \tag{1}$$

where N is the number of delay elements and T_{op} is the total latency due to the operations in the critical loop. The latency is defined as the time needed to produce an output with the same significance as the corresponding input. It is convenient to write the latency as, [4],

$$T_{op} = L_{op} \cdot T_{clk} \tag{2}$$

where L_{op} is the algorithmic latency in terms of clock cycles, and T_{clk} is the clock period which is determined by the critical path. T_{min} can in the same way be divided into L_{min} and T_{clk}, where L_{min} is the minimal sample period in terms of clock cycles.

As an example of a recursive algorithm, a bireciprocal third-order Lattice Wave Digital Filter, LWDF, is used. The filter shown in Fig. 1 has earlier been implemented using parallel carry-save arithmetic, [5], and bit-serial arithmetic, [6]. The filter coefficient is $\alpha = 0.375 = 0.011_2$. Hence, the number of fractional bits is $W_f = 3$. In digit-serial arithmetic, a sign-extension circuit is required in front of the multiplier to produce the most significant bits. The multiplication increases the word length with $W_f = 3$ bits, which are removed by the quantization block in order to keep the word length constant in the loop. By allowing the output y to be quantized and sign-extended, these operations can be located as shown in Fig. 1. The input word length is assumed to be 12 bits, which is sign-extended with W_f bits at the input, yielding an internal word length of $W_d = 15$ bits, in order to equalize the execution time for all operations. The W_f extra bits are also sufficient to prevent overflow in all nodes. Thus, no over/underflow saturation circuits are required, which otherwise would have reduced the throughput. The extra bits do not decrease the throughput, since it is independent of W_d. The maximum throughput for this slightly modified filter algorithm is

$$f_{max} = 2/(2T_{add} + T_{mult} + T_{Q,se}) \tag{3}$$

where T_{add}, T_{mult}, and $T_{Q,se}$ are the latencies for the adder, multiplier, and the combined quantization/sign-extension circuit, respectively.

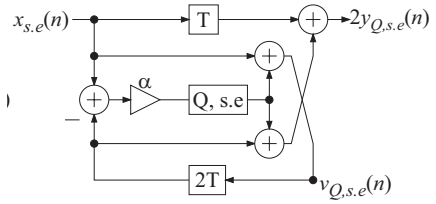

Fig. 1. A third-order bireciprocal LWDF

2. PROCESSING ELEMENTS

The latency and pipeline level of the Processing Elements, PE, e.g., adders and multipliers, are modelled by the Latency Model order, LM

$$Latency\ Model\ order = \frac{1}{n_{Adder}} \tag{4}$$

where n_{Adder} is the number of adders between each pipeline register. Thus, with $n_{PE} \to \infty$ the LM equals zero, LM 0, which corresponds to a non-pipelined adder, shown in Fig. 2 a. The contribution to the algorithmic latency in the loop equals $L_{PE0} = 0$. Hence, the latency in terms of clock cycles equals zero, while the clock period, T_{clk0}, is determined by the critical path. A cascade of LM 0 adders, yields an increased critical path and thus T_{clk0} and total latency in the cascade increase.

Latency Model 1, LM 1, corresponds to a pipelined adder, according to Fig. 2 b, with algorithmic latency, $L_{PE1} = 1$, i.e., the contribution to the latency in the loop in terms of clock cycles equals one, while the clock period, T_{clk1}, is determined by the delay of one adder. A cascade of LM 1 PEs yields an unchanged T_{clk1}, while the algorithmic latency L in the cascade increases.

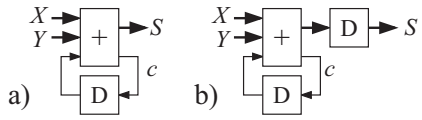

Fig. 2. Adders with a) LM 0, b) LM 1

2.1 Fixed coefficient multipliers and sign-extension

The algorithmic latency for multiplication with LM 0 is

$$L_{mult0} = \lceil W_f/D \rceil \tag{5}$$

due to the initial computation of the W_f least significant bits, which are overwritten by the quantization/sign-extension block with the most significant bits from the former multiplication. T_{clk0} is determined by the critical path, which in the worst case is one full-adder and one multiplexer.

The algorithmic latency for the quantization/sign-extension block is $L_{Q,\check{s}e0} = 0$. By inserting one pipeline stage with D flip-flops between the adder and the quantization/sign-extension block the multiplier yields LM 1, hence the adder in the multiplier is of LM 1 type, and the latency for the multiplication with LM 1 increase by one.

$$L_{mult1} = \lceil W_f/D \rceil + 1 \tag{6}$$

The T_{clk1} is somewhat prolonged, compared with the LM 1 adder, due to the multiplexer in the quantization/ sign-extension block.

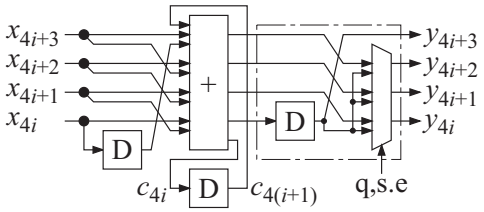

Fig. 3. Fixed coefficient digit-serial multiplier $D = 4$

Since the coefficient in the example is fixed, $\alpha = 0.011_2$, a general digit-serial/parallel multiplier can be simplified, removing partial product generators, and replacing adders with zero inputs with a feed-through, [7]. A multiplier of type LM 0 and $D = 4$, is shown in Fig. 3. When W_f is not a multiple of D, an alignment stage at the output of the multiplier is required to align the bits in the input and output digits by shifting $\lceil W_f/D \rceil D - W_f$ bits. The dashed frame shows the LM 0 combined quantization/sign-extension/alignment block.

3. THE FILTER IMPLEMENTATION

From the block diagram of the filter in Fig. 1 a precedence graph, shown in Fig. 4, is obtained. The graph shows the executable order of the operations.

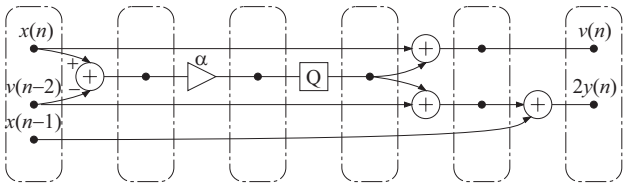

Fig. 4. Precedence graph of the filter

By introducing timing information of the operations, a computation graph is obtained from the precedence graph. Examples of computation graphs over a single sample interval N_i for LM 0 and 1 with $D = 3$ are shown in Fig. 5 a) and b) respectively. The shaded area indicates the execution time, while the darker area indicates latency. By inserting one pipeline register in the LM 0 computation graph by changing it to a LM 1 multiplier, the critical path equals three adders that yields a fractional order LM 1/3. By this choice of insertion point, the path from input to output is divided into two equal paths with two adders.

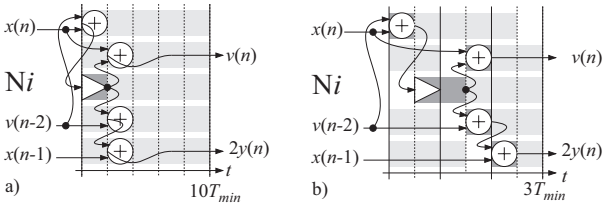

Fig. 5. Computation graphs for the filter, $D = 3$, a) LM 0, b) LM 1

3.1 The maximally fast schedule

When the execution time, E_W, in terms of clock cycles of the PEs is larger than the minimal sample period, L_{min}, or the critical loops contain more than one delay element, the maximally fast schedule can only be obtained by cyclic scheduling over a number of sample periods, m, [8]. Hence, the algorithm must be unfolded m times and concurrently compute m samples. For digit-serial arithmetic a lower bound of m is derived in the following. The execution time for a digit-serial PE is $E_W = W_d/D$. W_d is chosen as a multiple of D, and in this design $W_d \geq 15$ bits. The ratio between E_W and L_{min} determines the number of digits N_D that need to be processed concurrently

$$N_D = E_W/L_{min} \tag{7}$$

For simplicity, if only integer number of digits can be processed concurrently, and for special cases, e.g., several critical loops, the minimal number of sample periods m required in the schedule is given by the inequality.

$$m \geq \lceil N_D \rceil = \lceil W_d/(DL_{min}) \rceil \tag{8}$$

In the case with a non-integer N_D, a slack in the schedule occurs as shown in Fig. 5 b). The slack can be handled in

several ways. First, using a multiplexed cyclic schedule over $\lfloor N_D \rfloor$ and $\lceil N_D \rceil$ sample intervals so the mean value equals N_D. This strategy introduces additional complexity, hardware, power consumption, and decreases the throughput. Second, the approach in this work is to increase W_d until $N_D = m$, which increases the dynamic range beyond the specification. However, in a larger system the requirements on other parts can be relaxed. The maximally fast schedule is obtained by taking m sets of N_i and skew them L_{min} in time, the result for LM 1, $D = 3$ is shown in Fig. 6.

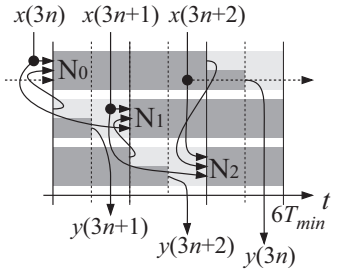

Fig. 6. Maximally fast schedule for LM 1, $D = 3$

In the case with a non-integer L_{min}, a non-uniform skew with alternating $\lfloor L_{min} \rfloor$ and $\lceil L_{min} \rceil$ integer clock cycles between the sets N_i can be used. This aligns all samples with the same clock phase. Since the filter has two delay elements in the loop, the state variable $v(n)$ from set N_i is connected to set N_{i+2}. Hence, an even m yields two separate loops with $m/2$ sets of N_i in each loop, and an odd m yields one loop with m sets of N_i.

3.2 Isomorphic mapping to hardware

An isomorphic mapping, shown in Fig. 7, of the operations in the cyclic schedule to hardware yields a maximally fast and resource minimal implementation. The branches in the computation graph with different start and stop time are mapped to shift-registers, i.e., the time difference is referred to as Shimming Delay, SD. Properties of a maximal fast implementation are that the critical loop has no SD, and the delay elements in the algorithm are totally absorbed by the latencies of the operations in the critical loop.

4. THROUGHPUT VERSUS DIGIT SIZE AND LATENCY MODEL

A model for optimal level of pipelining was developed in [4]. The use of that model is somewhat limited, since the PEs delay was modelled as a linear function of D, which is only valid in a small range for RCA, [9]. A more general result is developed in the following for three special cases, LM 0, fractional LM, and LM 1.

4.1 Latency Model 0

Combining (3) and (5) yields L_{min0} for the filter with LM 0.

$$L_{min0} = \lceil W_f/D \rceil/2 \tag{9}$$

L_{min0} decreases with increasing digit-size until $D = W_f$, while the T_{clk0} is limited by the critical path in the loop. Each set N_i has a critical path of three adders and a multiplexer. Thus, the total critical path is

$$T_{clk0} = \begin{cases} \frac{m}{2}(3t_{add0} + t_{mux0}) & \text{for even } m \\ m(3t_{add0} + t_{mux0}) & \text{for odd } m \end{cases} \tag{10}$$

It is not easy to calculate the minimum T_{clk0}, since m decreases with increasing D while t_{add0} increases. Hence, in this case all digit-sizes in the range $1 < D \le \lceil W_d/2 \rceil$ have to be evaluated for maximum throughput.

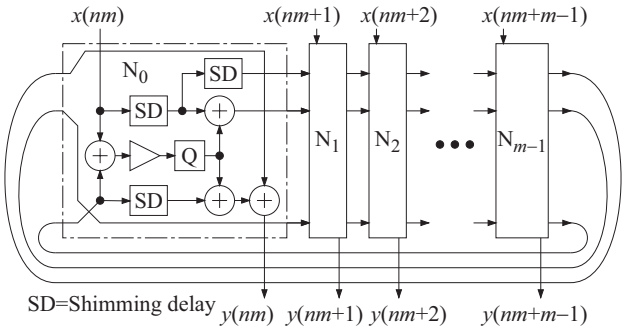

Fig. 7. Hardware structure for the unfolded filter

4.2 Fractional Latency Model

Combining (3), (6) and use of $L_{add} = 0$ yield the minimal sample period L_{min} and a LM (1/3) for the filter.

$$L_{min(1/3)} = [\lceil W_f/D \rceil + 1]/2 \tag{11}$$

L_{min} has decreased by one clock cycle compared with the LM 1 filter. The critical path consists of two LM 0 adders, t_{add0}, one LM 1 adder, t_{add1}, and a multiplexer, t_{mux0}, shown in (12). The critical path has increased with $2t_{add0}$ compared with the LM 1 filter. It is however much shorter than the critical path of the LM 0 filter.

$$T_{clk(1/3)} = 2t_{add0} + t_{add1} + t_{mux0} = 2t_{add0} + T_{clk1} \tag{12}$$

The $L_{min(1/3)}$ decreases with increasing digit-size until $D = W_f$, while the $T_{clk(1/3)}$ increase with digit-size. Hence, maximum throughput is obtained for a digit-size in the range $1 \le D \le W_f$. This result can be generalized for all fractional LMs due to the number of adders in the critical path is not dependent on the digit-size but is constant. Hence, the critical path increases with increasing digit-size.

4.3 Latency Model 1

Combining (3) and (6) yields the minimal sample period L_{min1} for the filter with LM 1

$$L_{min1} = [\lceil W_f/D \rceil + 3]/2 \tag{13}$$

L_{min1} decreases with increasing digit-size until $D = W_f$,

while the clock period T_{clk1} is limited by the critical path that consist of an adder, t_{add1}, cascaded with a multiplexer, t_{mux0},

$$T_{clk1} = t_{add1} + t_{mux0} \quad (14)$$

where t_{add1} increases with digit-size and t_{mux0} is constant. Hence, the maximum throughput is obtained for a digit-size in the range $1 \le D \le W_f$.

5. COMPARISON

Estimates of the throughput and energy consumption of the filters are obtained by HSPICE simulations of the netlists extracted from the layout for the RCA and CLA digit-serial adders, [9], which were implemented in a standard 3-metal layer 0.35 µm CMOS process.

For the LM 1 and LM 1/3 filter, the maximum throughput is obtained using $D = W_f$ and RCA adders, shown in Fig. 8. The maximum throughput for the LM 0 filter is obtained using $D = 8$ and CLA adders. Another maximum is obtained with RCA adders using $D = W_f$.

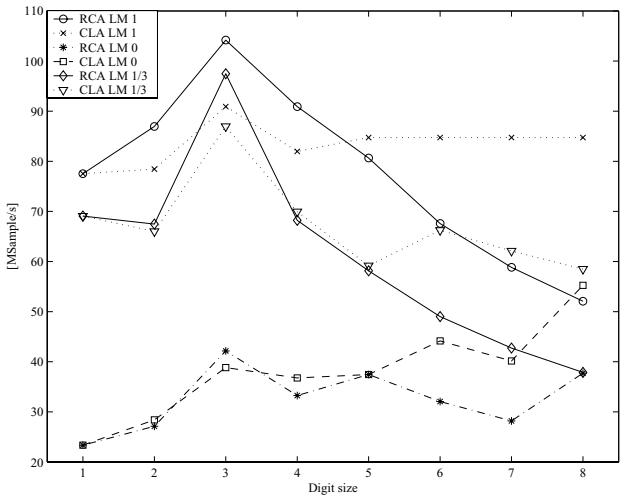

Fig. 8. Throughput versus digit-size

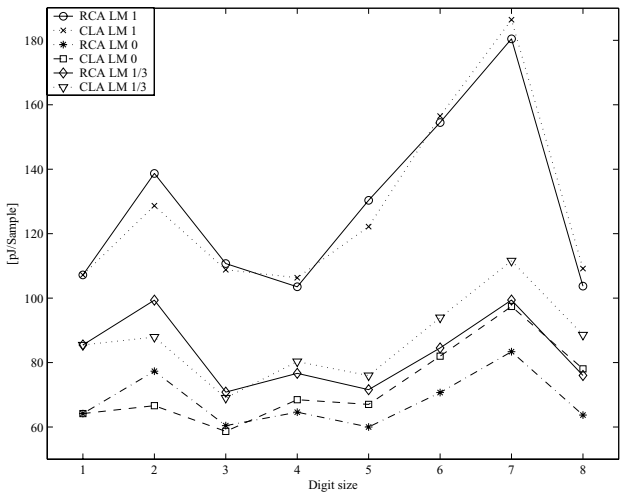

Fig. 9. Energy consumption per sample versus digit-size

The energy consumption per sample for the LM 1 filters shown in Fig. 9 has a minimum for $D = 4$ using RCA adders. The energy consumption for the LM 1/3 filters has a minimum for $D = 3$ with CLA adders. The energy consumption for the LM 0 filters has a minimum for $D = 5$ with RCA adders. With CLA adders the minimum is for $D = W_f$. For maximum throughput, a use of RCA adders and $D = W_f$, LM 1 is recommended. For low energy consumption, a use of RCA adders, $D = W_f$, LM 0 is recommended. A trade-off between energy consumption and throughput can be made by using a fractional LM order. By inserting just one pipeline register in each set of operations for a single sample interval LM 1/3 is obtained, yielding at the same time near maximum throughput and near minimum energy consumption.

6. CONCLUSION

Design trade-offs in pipelining of processing elements implementations of recursive algorithms using digit-serial arithmetic by means of investigating a bireciprocal third-order LWDF have been presented.

For maximum throughput, a use of pipelined processing elements with $D = W_f$ and RCA adders is recommended. This implementation is 150 percent faster than using non-pipelined processing elements.

For minimum energy consumption, a use of non-pipelined processing elements with $D = W_f$ and RCA adders is recommended. This implementation has 45 percent lower energy consumption than using pipelined processing elements.

A trade-off between energy consumption and throughput can be made by introducing some pipelined processing elements in the non-pipelined filter design. Using three adders between every pipeline register, and $D = W_f$, yields a near maximum throughput only decreased with 6 percent and a near minimum energy consumption only increased with 17 percent.

REFERENCES

[1] S. G. Smith, and P. B. Denyer, *Serial data computation*, Kluwer, 1988.

[2] R. I. Hartley, and K. K. Parhi, *Digit-serial computation*, Kluwer, 1995.

[3] M. Renfors, Y. Neuvo, "The maximum sampling rate of digital filters under hardware speed constraints," *IEEE Trans. on Circuits and Systems*, Vol. 28, No. 3, pp. 196-202, March, 1981.

[4] O. Gustafsson, and L. Wanhammar, "Optimal logic level pipelining for digit-serial implementation of maximally fast recursive digital filters", *in Proc. RVK-02*, Karlskrona, Sweden, 2002

[5] U. Kleine, and M. Böhner, "A high-speed wave digital filter using carry-save arithmetic," *in Proc. ESSCIRC'87*, Bad-Soden, pp.43-46, 1987.

[6] M. Vesterbacka, K. Palmkvist, and L. Wanhammar, "Implementation of fast bit-serial lattice wave digital filters," *in Proc. IEEE Symp. ISCAS'94*, Vol. 2, pp. 113-116, London, May 30-June 1, 1994.

[7] L. Wanhammar, *DSP integrated circuits*, Academic Press, 1999.

[8] D.A Schwartz, and T.P. Barnwell III, "Cyclo-static solutions: optimal multiprocessor realizations of recursive algorithms," *VLSI Signal Processing II*, IEEE Press, 1986.

[9] M. Karlsson, M. Vesterbacka, and W. Kulesza, "Ripple-carry versus carry-look-ahead digit-serial adders," *in Proc. IEEE Conf. NORCHIP'03.*, pp. 264-267, Riga, Latvia, Nov., 2003.