

## VLSI Implementation of Digit-Serial Arithmetic Modules

L. Bisdounis<sup>a</sup>, D.E. Metafas<sup>a</sup>, A.M. Maras<sup>b</sup>, C. Mavridis<sup>a</sup>

<sup>a</sup>VLSI Design Laboratory, Department of Electrical Engineering, University of Patras, 26110 Patras, Greece.

<sup>b</sup>Department of Electronics & Computer Engineering, University of Crete, 73100 Chania, Crete, Greece.

This article describes an implementation of arithmetic modules which is based on the transmission of arithmetic data serially one digit at a time. For some applications bit-serial architectures may be too slow, and bit-parallel architectures may be faster than necessary and require too much hardware. The desired sample rate in these applications can be achieved using the digit-serial approach.

### 1. INTRODUCTION

The digit-serial techniques are ideal in Digital Signal Processing as they offer the flexibility which is needed in order to exploit the potential of the available technology. Bit-serial implementations process one bit at a time and are suitable for low speed applications such as communications and speech processing [1]. On the other hand very high sample rates in radar, video and image processing call for fully bit-parallel implementations which process all input bits of a word or sample in one clock cycle [2].

In digit-serial computations, data words are divided into digits and transmitted serially one digit at a time between operators. Architectural synthesis based on digit-serial modules offers the structured design approach needed in order to find the best solution for the application that concerns the tradeoffs between cost, speed, efficient area utilisation, throughput, clock distribution, I/O pin limitation and power dissipation. Hartley et al. [3] explored some digit-serial structures, while Parhi [4] proposed an approach to the design of digit-serial arithmetic modules that transforms a bit-serial structure into a digit-serial structure using an unfolding transformation algorithm. Although the approach presented in [4], eased the complexity of the processor design procedure, its major drawback is that the designer has not control over the architecture of the final digit-serial structure, because it is completely specified by the initial choice of the bit-serial realisation and the application of the unfolding algorithm, with the corresponding unfolding factor.

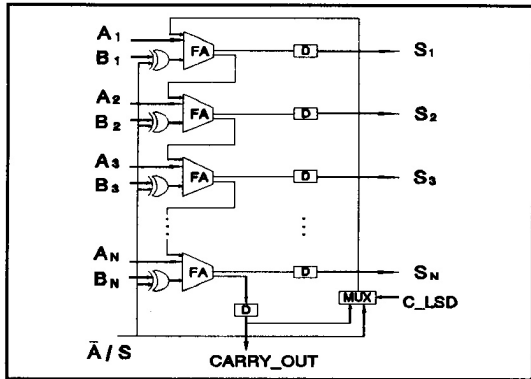
In this paper the architecture of digit-serial

add/subtract cell and the architectures of very efficient digit-serial arithmetic modules of multiplier, divider and square root extractor are presented. Also, adaptations of the algorithms are proposed in order to be used in digit-serial implementations. The advantage of the presented design procedure is its simplicity and that it will allow the designer to find the best solution in a methodical manner. Also, these arithmetic modules are highly pipelined in order to achieve high throughput.

### 2. DIGIT-SERIAL ADD/SUBTRACT CELL

A basic digit-serial operator is the digit-serial adder/subtractor shown in figure 1. It is assumed that the digit size is  $N$ . The two operands  $A$  and  $B$  are fed one digit at a time to the cell. The operation (add/subtract) is done  $N$  bits at a time, with the carry rippling from one full-adder to the next. The delayed carry-out from the bottom of the cell, is returned to the top to be fed back into the first full-adder, during the next clock cycle, when the next digits of the inputs have arrived. The carry-out also gets an initial value on the least significant digits of the inputs. This is ensured by the signal  $\bar{A}/S$ , which also controls the inversion of the  $B$  input bits. There is one control signal  $C\_LSD$  which is high during the least significant digits of the inputs. This signal is used to define, when the carry-out gets an initial value.

To do addition,  $\bar{A}/S$  is set to zero causing the carry to be reset to a zero value and addition to be performed in a regular manner. To do subtraction,  $\bar{A}/S$  is set equal to one, causing the  $B$  input to be complemented and a logic one to be



**Figure 1:** Digit-serial Add/Subtract cell carried into the initial carry-in position, resulting in a two's complement subtraction of B from A.

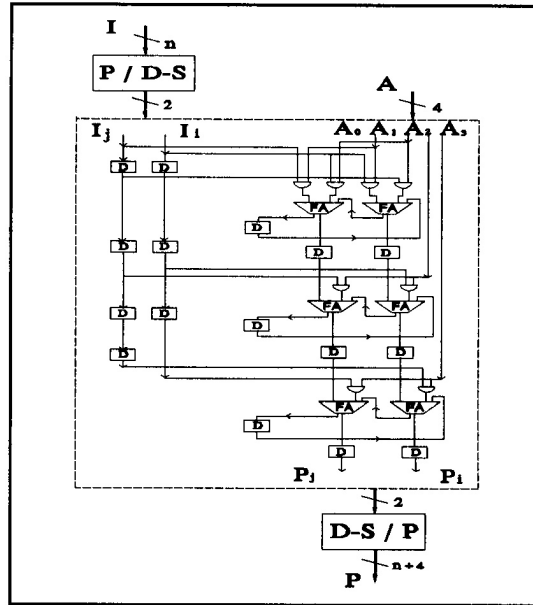
**3. DIGIT-SERIAL MULTIPLIER**

In this section a digit-serial multiplier implementation, with fixed coefficient, is presented [4],[5],[6]. Consider the multiplication of an n-bit integer I with a 4-bit coefficient A. The size of I is variable, while the size of A is fixed. The multiplicand I, the coefficient A (multiplier), and the (n+4)-bit product P are represented as:

$$I = I_{n-1} \dots I_1 I_0, \quad A = A_3 A_2 A_1 A_0, \quad P = P_{n+3} \dots P_1 P_0$$

respectively. The digit-serial multiplication with a digit size of two bits, uses the architecture shown in detail in figure 2. The number of rows of full-adders is set equal to the number of bits of the multiplier A minus one, while the number of columns in the basic circuit, to the digit size. The idea in the proposed architecture is that the partial products of the multiplication are divided into equally sized groups, and the carry ripples horizontally in the next group of partial products. The bits of the multiplicand are supplied one digit at a time, starting with the least significant digit, whereas the bits of the multiplier are supplied in parallel form. In order to increase the throughput of the multiplier, the architecture is highly pipelined.

Since there is no gap between successive words of data, a zero digit is inserted after each word. In this way, it is also achieved to get from the output the most significant bit of the product at the end of the multiplication operation. In order to increase the digit size, all we have to do is to add columns in the basic circuit. In figure 2 the parallel-to-digit-serial and the digit-serial-to-parallel interfaces are also shown.



**Figure 2:** Digit-serial Multiplier

**4. DIGIT-SERIAL DIVIDER**

In the following, the non-restoring division algorithm which has been adopted for digit-serial implementation and the architecture for digit-serial implementation of two's complement division are presented [3],[6].

**4.1. Non-restoring Division Algorithm:**

Consider two binary fractions A and B. A is the divided, B the divisor and  $Q = A/B$  the quotient. We assume that  $|A| < |B|$  and  $B \neq 0$ . The flow-chart of the algorithm used, is given in figure 3. Division process requires controlled add-subtract (CAS) operations. Whether the next operation is an addition or subtraction, is controlled by the result of the current operation.

In the case that A, B are of different sign, the first operation is an addition, otherwise the first operation is a subtraction. The number of bits of the quotient is equal to the times the second loop is performed. The procedure ends when i reaches the number of bits ( $\kappa$ ) desired in the quotient Q.

**4.2. Digit-serial Divider Implementation**

The architecture of the digit-serial divider, which uses least significant digit (LSD) first data format, is shown in figure 4. It is assumed that the word size is W, the digit size is N, where W is constrained to be an integer multiple of N. The first

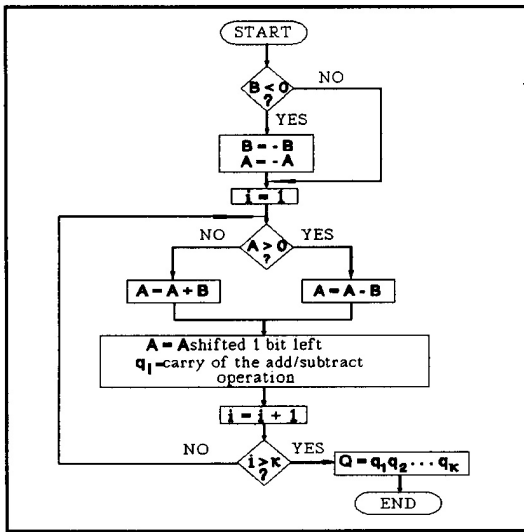


Figure 3: Non-restoring Division Algorithm

part of the circuit detects whether  $B < 0$ , and if it is, changes the sign of both  $A$  and  $B$  in order to ensure that  $B \geq 0$ . Then the division is continued by repeatedly adding or subtracting  $B$  from  $A$  with the decision to add or subtract being determined by the carry of the previous operation.

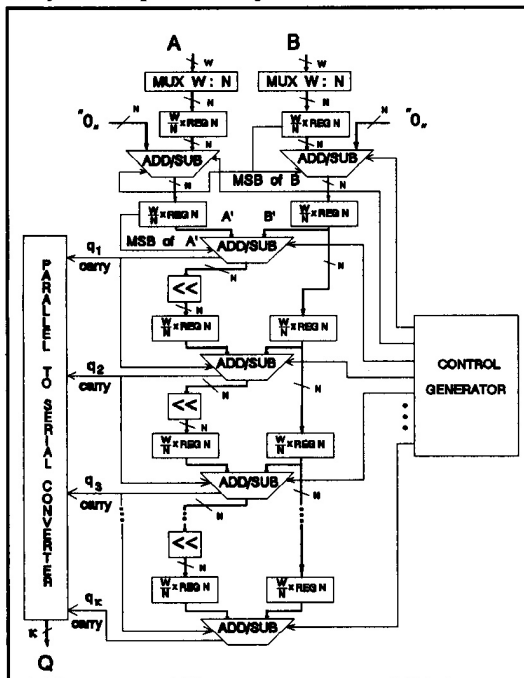


Figure 4: Digit-serial Divider

The presented architecture is fully pipelined. The cells denoted as "W/N X REGN",

consist of  $W/N$  simple registers of  $N$  bits each, used to move the data in the pipeline. In order to have the serial transmission, one digit at a time, of the data words, two multiplexers with  $W$  inputs and  $N$  outputs are used at the inputs of the circuit. The ADD/SUB cells which have been presented in section 2, perform controlled addition or subtraction of  $W$  bits numbers in a digit-serial manner. In these cells, the required control signals are provided by a control unit (CONTROL GENERATOR) which is a circular shift register, with only one flip-flop being set at any clock cycle while all others are cleared. Each control signal is high for exactly one clock cycle, every  $W/N$  clock cycles. In this way, only  $W/N$  different control signals are required, because of periodicity. For simplicity, in figure 4 we use the convention that the input shown on the left of each ADD/SUB cell is the  $\bar{A}/S$  input, and the input shown on the right is the  $C_{LSD}$  input. The cells denoted as  $\ll$ , are one bit left shifters. Finally, in order to get the total quotient in a serial format, a parallel-to-serial interface is used.

### 5. DIGIT-SERIAL SQUARE ROOT EXTRACTOR

This section describes the non-restoring square root algorithm which has been adopted for digit-serial implementation and presents the architecture for digit-serial implementation of two's complement square root operations.

#### 5.1. Non-restoring Square Root Algorithm

Consider two positive binary fractions  $A$  and  $Q$  such that  $Q = \sqrt{A}$ . We call  $Q$  the square root of  $A$ . Note that:

$$A = 0.a_1a_2\dots a_{2n-1} \text{ and } Q = 0.q_1q_2\dots q_n$$

The flow-chart of the non-restoring square root algorithm is illustrated in figure 5. As the division process, square root process requires controlled add-subtract (CAS) operations. Whether the next operation is an addition or subtraction, is controlled by the carry of the current operation [6],[7]. The first operation is subtraction. The number of bits of the operand  $D_0$  is set equal to the number of bits of  $A$ , by appending  $2n-2$  zeros to the right of it. The  $\kappa$  most significant bits of the operands  $D_1, D_2, \dots, D_{n-1}$ , are zeros. Also, the number of bits of these operands is set equal to the number of bits of  $A$  by appending  $2(n-\kappa)-2$  zeros to the right of them. The number of bits of the result ( $Q$ ) magnitude, is equal

to the times the loop of the algorithm is performed. This is the number of bits pairs of A. The procedure ends when  $\kappa$  becomes equal to n.

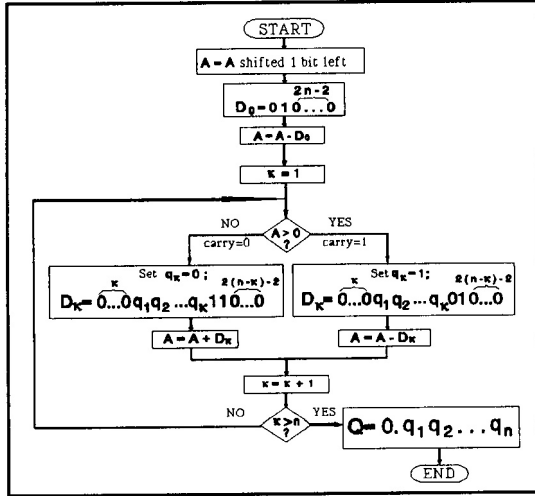


Figure 5: Non-restoring Square Root Algorithm

5.2. Digit-serial Square Root Implementation

The architecture of the digit-serial square root extractor is shown in figure 6. Consider that the word size is W and the digit-size is N. The word size W must be an integer multiple of N. A number of "W to N" multiplexers and "W/N X REGN" specific registers are used for the reasons which have been presented in sub-section 4.2.

The first part of the circuit shifts one bit left the input number, so the algorithm could be applied to the magnitude of the number. The first ADD/SUB cell, always performs the subtraction of the operand  $D_0$  from A. After the first subtraction, the square root process is continued by repeatedly adding or subtracting the operands  $D_1, D_2, \dots, D_{n-1}$  from A, with the decision to add or subtract being determined by the carry of the previous operation. The control signals which are required in the ADD/SUB cells, are provided by the CONTROL GENERATOR. The general structure of this subcircuit has been described in section 4.2. The output unit of the circuit is a parallel-to-serial converter, in order for the result to be get in serial manner.

6. CONCLUSIONS

In this article the digit-serial pipelined architectures of three arithmetic modules has been presented in full detail, as well as the algorithms on

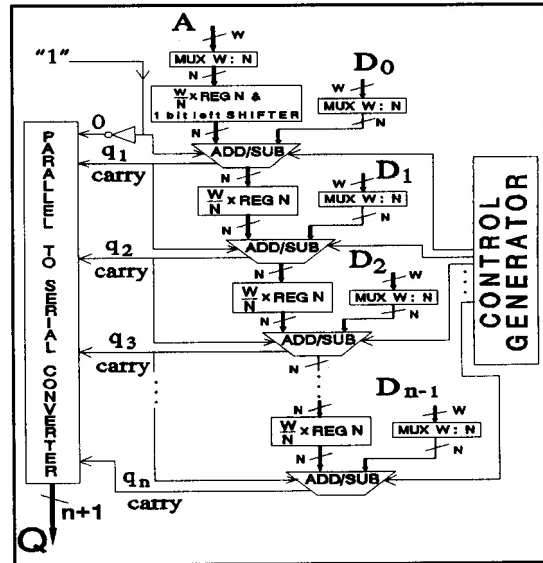


Figure 6: Digit-serial Square Root Extractor

which they are based. These high throughput modules are very efficient in Digital Signal Processing applications.

REFERENCES

1. P. Denyer and D.Renshaw, VLSI Signal Processing, A Bit-Serial Approach. Reading, MA: Addison-Wesley, 1985.
2. M.Hatamian and G.Cash, "Parallel bit-level pipelined VLSI designs for high speed signal processing", Proc. IEEE, vol. 75, pp. 1192-1202, Sept. 1987.
3. R.Hartley and P.Corbett, "Digit-Serial Processing Techniques", IEEE Trans. on Circuits and Systems, vol. 37, No. 6, June 1990, pp. 707-719.
4. K.Parhi, "A Systematic Approach for Design of Digit-Serial Signal Processing Architectures", IEEE Trans. on Circuits and Systems, vol. 38, No. 4, April 1991, pp. 51-58.
5. D.Ait-Boudaoud, M.KIbrahim and B.R.Hayes, "Novel cell architecture for bit level systolic arrays multiplication", Proc. IEE, vol. 138, No. 1, January 1991, pp. 21-26.
6. K.Hwang, Computer Arithmetic: Principles, Architectures and Design. New York: Wiley, 1979.
7. P.Montuschi and M.Mezzalama, "Survey of square rooting algorithms", Proc. IEE, vol. 137, No. 1, January 1990, pp. 31-40.