An area-efficient topology for VLSI implementation of Viterbi decoders and other shuffle-exchange type structures

Sparsø, Jens; Jørgensen, Henrik Nordtorp; Paaske, Erik; Pedersen, Steen; Rübner-Petersen, Torben

Published in:
IEEE Journal of Solid State Circuits

Link to article, DOI:
10.1109/4.68122

Publication date:
1991

Document Version
Publisher's PDF, also known as Version of record

An Area-Efficient Topology for VLSI Implementation of Viterbi Decoders and Other Shuffle-Exchange Type Structures

Jens Sparsø, Henrik N. Jørgensen, Erik Paaske, Steen Pedersen, and Thomas Rübner-Petersen

Abstract—**In this paper we present a topology for single-chip implementation of computing structures based on shuffle-exchange (SE) type interconnection networks. The topology is suited for structures with a small number of processing elements (i.e., 32–128) whose area cannot be neglected compared to the area required for interconnection.**

The processing elements are implemented in pairs which are connected to form a ring. In this way three-fourths of the interconnections are between neighbors. The ring structure is laid out in two columns and the interconnection of nonneighbors is routed in the channel between the columns. The theoretical basis for this structure is that an SE type interconnection network can also be viewed as a deBruijn graph in which it has been shown that it is always possible to find Hamilton cycles.

The topology has been used in a VLSI implementation of the add–compare–select (ACS) module of a fully parallel $K = 7$, $R = 1/2$ Viterbi decoder. The paper describes both floor-planning issues and some of the important algorithm and circuit level aspects of this design.

The chip has been designed and fabricated in a 2-μm CMOS process using MOSIS-like simplified design rules. The chip operates at speeds up to 19 MHz under worst-case conditions ($V_{DD} = 4.75$ V and $T_A = 70$°C). The core of the chip (excluding pad cells) measures 7.8×5.1 mm^2 and it contains approximately 50 000 transistors. The interconnection network occupies 32% of the area.

I. Introduction

CONVOLUTIONAL coding with Viterbi decoding is an efficient technique to reduce the bit error rate on noisy digital communication links [1]. For information bit rates above 5 Mb/s, decoders are based on fully parallel implementations of the Viterbi algorithm in order to achieve the required speed. Such a fully parallel implementation involves a large number of identical processing elements interconnected by a shuffle-exchange type network. Placement and routing of these processing elements is known to be a difficult and area-intensive task, and it is one of the key issues in an efficient implementation [2].

With the evolution of VLSI it has become possible to implement Viterbi decoders in an economical way, and a number of specific designs have been reported [3]–[6]. The last two are commercially available (or announced) products for the $K = 7$, $R = 1/2$ code [7], operating at 25 and 17 MHz, respectively. Another body of literature deals with layout structures for the interconnection network [2], and reformulation of the algorithm in order to localize communication and enable trade-offs between area and speed [8].

We have found that very little information is available about the topological organization of the processing elements in the above-mentioned designs. Furthermore, we found that the optimal shuffle-exchange (SE) graph layout from [9] applied to the implementation of Viterbi decoders as proposed in [2] yields layouts with a poor utilization of silicon area for the type of decoders we are considering, because the area requirement of the processing elements is significant.

In the following, we present a novel two-column ring topology that has a regular and area-efficient layout that can be generally applied to the layout of computing structures with interconnection networks of the SE or deBruijn type. We also describe the implementation of the decoder in detail, and present a number of area and speed measures for a fabricated test chip.

The paper is organized as follows. In Section II we give a brief introduction to convolutional coding and Viterbi decoding in order to formulate the design problem. In Section III we discuss the application of SE graph layouts to Viterbi decoders. Section IV presents the two-column ring topology that is based on Hamilton cycles in deBruijn graphs. Finally, Section V describes circuit implementation, layout of the chip, and test results.

II. The Viterbi Decoding Algorithm

A. Basic Principle

In order to establish a notation and briefly introduce the operation of the decoder, it is convenient to start by describing the encoder. The encoder for the standardized $K = 7$, $R = 1/2$ code [7] is shown in Fig. 1. It is a synchronous state machine with one input u_i, two outputs $x_{1i}x_{2i}$, and 64 states (Fig. 2).

Expansion of this diagram by drawing all states for each clock cycle yields a lattice. The problem of decoding can then be expressed as finding the path through the lattice that most closely matches the received sequence as measured by a calculated *path metric* (PM). As described in [1, ch. 6], this involves three steps: 1) branch metric (BM) computation,

Manuscript received May 15, 1990; revised August 23, 1990. Part of this work was supported by the Danish Technical Research Council under Grant STVF/FTU 5.17.5.6.27.

J. Sparsø and S. Pedersen are with the Department of Computer Science, Technical University of Denmark, DK-2800 Lyngby, Denmark.

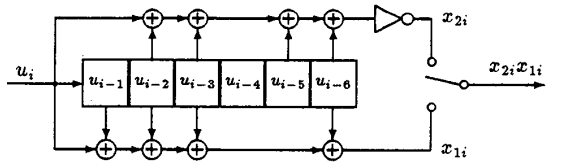
H. N. Jørgensen was with the Department of Computer Science, Technical University of Denmark, DK-2800 Lyngby, Denmark. He is now with the Microelectronic Test Center, ElektronikCentralen DK-2970, Hørsholm, Denmark.

E. Paaske is with the Institute of Telecommunication, Technical University of Denmark, DK-2800 Lyngby, Denmark.

T. Rübner-Petersen, deceased, was with the Institute of Telecommunication, Technical University of Denmark, DK-2800, Lyngby, Denmark.

IEEE Log Number 9040725.

0018-9200/91/0200-0090$01.00



Constraint length: $K = 7$
Code rate: $R = 1/2$
Memory depth: $\nu = 6$ (i.e. 64 states)
Generator Polynomials: $G1 = D^0 + D^1 + D^2 + D^3 + D^6$
$G2 = D^0 + D^2 + D^3 + D^5 + D^6$

Fig. 1. Encoder for the $K = 7$, $R = 1/2$ code [7].

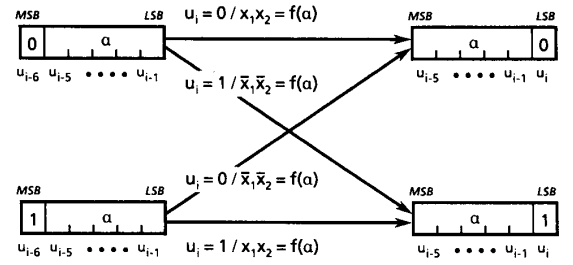


Fig. 2. State graph for the $K = 7$, $R = 1/2$ encoder, $\alpha \in [0, 31]$. Output x_1x_2 is a function of α.

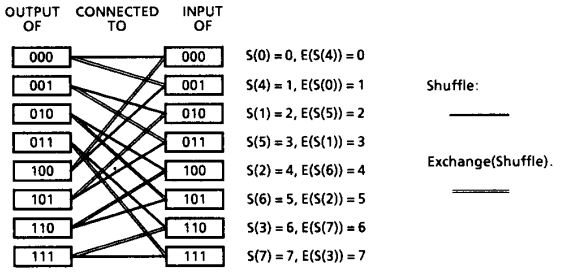


Fig. 3. PM interconnection network for $K = 4$, derived from Fig. 2 with $\alpha \in [0, 3]$.

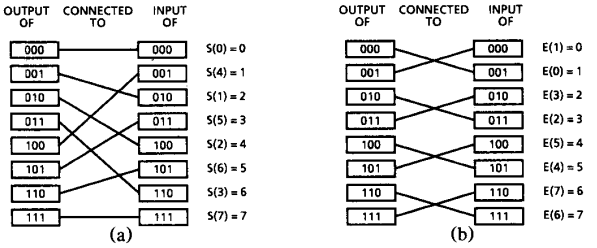


Fig. 4. SE interconnection network implementing both the shuffle interconnections shown in (a), and the exchange interconnections shown in (b). The figure shows an example with eight processing elements.

2) PM updating and storage, and 3) path storage and output sequence selection.

In this paper we restrict our attention to the implementation of the PM updating and storage operation. A parallel implementation of the $K = 7$ algorithm consists of 64 processing elements, one for each state in a column of the lattice. In every clock cycle each processing element performs the following simple operations. For the two paths entering a state, the accumulated PM is calculated by adding the BM associated with the state transition and the PM of the preceding state. The two sums are compared, and the smaller of the two is stored as the new PM of the state, and the decision (that represents the most likely path into the state) is output. For obvious reasons the processing elements are commonly denoted *add–compare–select* processing elements (ACS elements), and we use the term *ACS module* for the whole array of ACS elements.

It should be noted that the PM's increase over time. Therefore some mechanism that reduces all PM's at appropriate times is needed. This is called *rescaling*. This and other choices regarding the implementation of the ACS elements are discussed in Section V.

B. Physical Implementation—The Wiring Problem

In a fully parallel VLSI decoder the interconnection of the ACS elements takes up a significant part of the chip area, and minimizing the wiring area is an important task. The wiring can be grouped into two categories:

1) Global signals that have to be distributed to all the ACS elements. These are *four BM buses*, *clock signals*, and *power supply*.

2) The PM interconnection network, illustrated in Fig. 3, for a $K = 4$ code (the terms shuffle and exchange are explained in the next section). In this simple case the interconnection network consists of 16 PM buses. For $K = 7$ the 64 ACS elements are connected by 128 PM buses.

III. Shuffle-Exchange Graph Layouts

It appears that, for parallel implementations of the Viterbi decoding algorithm, only two systematic approaches to the physical organization have been reported in the literature [2]: the *shuffle-exchange* (SE) organization that is applicable to this binary alphabet $K = 7$, $R = 1/2$ Viterbi decoder and the *cycle-connected cubes* (CCC's) described in [2] as a structure that "contains even more interprocessor wiring area, but has some applications of its own." In the following, we discuss the SE approach to the layout of the Viterbi decoder PM interconnection network.

A. The SE Interconnection Network

The SE interconnection network has many applications, and is described in many textbooks, e.g., [10] and [11]. Given N processing elements, $N = 2^n$, an SE network is a network in which the output of processing element P, with the binary representation $p_{n-1}p_{n-2}\cdots p_0$, is connected to the input of processing elements P' and P'', where

$$P' = Shuffle(P) = Shuffle(p_{n-1}p_{n-2}\cdots p_0) = p_{n-2}\cdots p_0 p_{n-1}$$

$$P'' = Exchange(P) = Exchange(p_{n-1}p_{n-2}\cdots p_0) = p_{n-1}p_{n-2}\cdots \bar{p}_0.$$

Fig. 4 shows eight processing elements interconnected by an

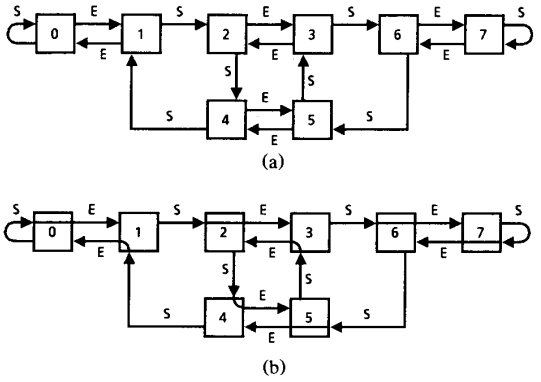


Fig. 5. (a) Possible layout graph for the eight-node SE network of Fig. 4, and (b) implementation of the Viterbi decoder PM interconnection network of Fig. 3 on the same SE network by adding feedthroughs in the nodes.

SE interconnection network, and a corresponding graph representation is shown in Fig. 5(a).

Based on the assumption that the wire area dominates the processor area, optimal layout organizations have been found [9], [12]. In this work, layout area is expressed in terms of the so-called Thomson grid model where processing elements are represented by points located at the intersection of grid lines, and wires connecting pairs of processing elements follow along grid lines [13].

B. Application to Viterbi Decoders

A comparison of Figs. 3 and 4 clearly shows the close correspondence between the SE interconnection network and the Viterbi decoder PM interconnection network. Expressed in terms of the functions *Shuffle* and *Exchange* defined above, the PM (output) of processing element P in a Viterbi decoder is connected to the inputs of processing elements P' and P'' (see also Fig. 3), where

$$P' = Shuffle(P) = p_{n-2} \cdots p_0 p_{n-1}$$

$$P'' = Exchange(Shuffle(P)) = p_{n-2} \cdots p_0 \bar{p}_{n-1}.$$

This interconnection structure can be implemented on a SE network with a small modification of the nodes as shown in Fig. 5(b). Based on this observation [2] proposes to adopt the results of [9] and [12] to the layout of Viterbi decoders.

C. Discussion

For the fully parallel $K = 7$, $R = 1/2$ Viterbi decoder the area of the ACS elements is significant, and a direct implementation following the optimal SE-graph layout [9, fig. 6] would result in large "white areas" as sketched in Fig. 6.

The layout organization of Fig. 6 could be used as a starting point for a compaction procedure, but for several reasons this is a difficult task. The number of nodes in the different columns and rows varies and many of the interconnections in the original layout are obtained by abutting nodes, and these interconnections will either be broken (creating new wires), or they will impose constraints on the compaction. In addition, the distribution of the global signals (BM buses, clock signals, and power supply) will be more irregular, and consequently take up a larger area.

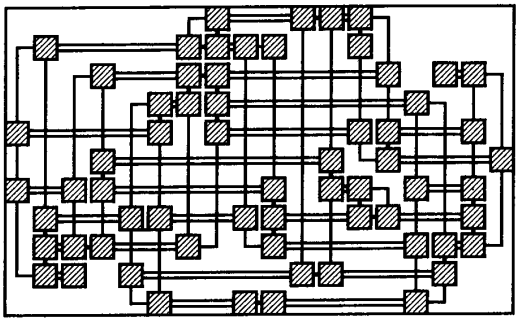


Fig. 6. Direct implementation of the Viterbi decoder following the optimal grid layout from [9, fig. 6].

The area of our implementation of the ACS module is 39.5 mm^2 (Section V), and based on the actual layout area of a single ACS element in our implementation (see Table I), it is estimated in [14] that the area of the uncompacted SE-graph layout would be 90 mm^2 and the area of a compacted layout would be at least 60 mm^2.

To conclude, the Viterbi decoder that we are considering does not fulfill the assumptions from which the optimal SE layout graphs are derived (that the wires dominate the area), and attempts to compact layouts based on the optimal SE layout graphs do not yield sufficiently dense layouts.

IV. The Two-Column Ring Topology

In our design studies we had two goals: 1) a regular floor plan with a simple distribution of global signals, and 2) optimization of the PM-interconnection network. Furthermore, we compared the real layout area of different solutions.

A. Basic Principle

It is a well-known practice to implement the ACS elements in pairs, as they share the same PM inputs two by two (Fig. 7(a), (b)) [1]. This reduces the interconnection problem by a factor of 2, i.e., the number of "global" PM buses is reduced by a factor of 2. Notice that it is the shuffle interconnections that remain, and that the network connecting the double ACS elements (DACS elements) is of exactly the same type as the original one.

To further optimize, we arranged the DACS elements in such a way that most communication is between neighbors or near neighbors. The DACS elements can be laid out forming one directed cycle, in which one of the two outgoing PM buses of a DACS element connects directly to the neighbor. By organizing the ring as two columns, routing of the remaining PM buses—one quarter of the original number—can be done in the channel between the two columns. Fig. 7(d) illustrates this for the simple eight-node example.

The key to the ring organization was found in a body of mathematical literature normally not related to floor planning [15], [16]. The Viterbi decoder and the SE interconnection networks are examples of deBruijn graphs. A deBruijn graph is a directed and connected graph in which the number of vertices is a power of 2, and in which every vertex has exactly two ingoing and two outgoing edges. For this type of graph it is always possible to find a Hamilton cycle, i.e., a

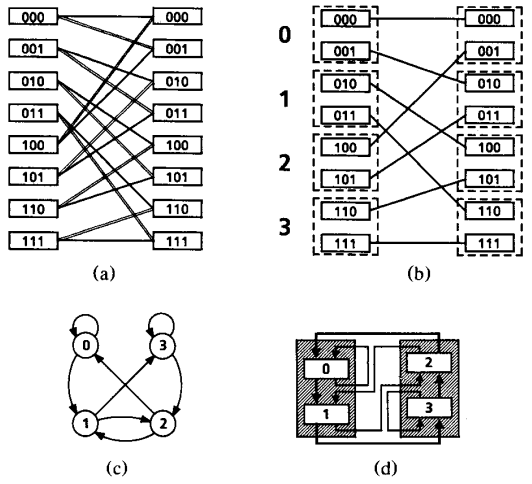


Fig. 7. The basic steps leading to the two-column ring topology: (a), (b) organization in pairs, (c) identification of Hamilton cycles, here (0,1,3,2,0), and (d) the resulting two-column floor plan.

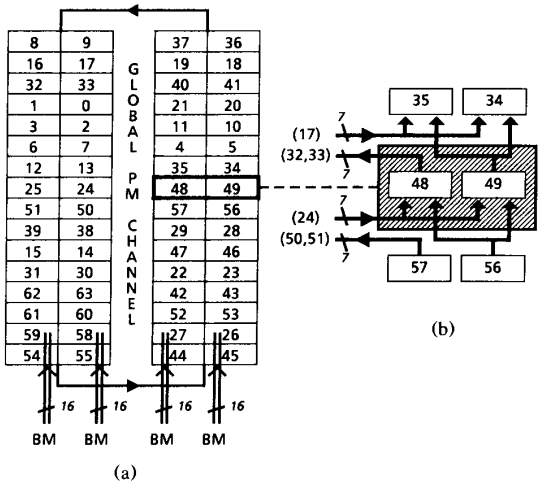


Fig. 8. Floor plan of the complete ACS module.

path which goes through all of the vertices once and only once. Fig. 7(c) shows the graph for the eight-node example. In this simple case there is only one Hamilton cycle (0,1,3,2,0), and it is easy to find. The general case is more complicated and requires computer programs.

The resulting floor plan for the ACS module of the $K = 7$, $R = 1/2$ Viterbi decoder is shown in Fig. 8(a), and the details are explained below. The required interconnections are derived from Fig. 2.

The two-column ring topology has the advantage that routing of the remaining PM buses has been reduced to a channel routing problem that is well understood. Furthermore the structure is very regular and the global signals can easily be routed vertically through the four columns of ACS elements. It should also be noted that I/O signals of the ACS module (i.e., the decision signals and the global signals) are easily made available at the periphery of the structure, whereas the PM buses that are internal signals in the ACS module are routed in the channel in the middle of the structure.

B. Optimization

The height of the structure is minimized if the ACS element whose output connects to the neighbor DACS element is placed outermost as shown in Fig. 8(b). This allows the two outgoing PM buses of a DACS element to share a horizontal track in the "local channel" between the DACS elements.

The width of the structure can be minimized by selecting a placement of the DACS elements that can be routed using as few vertical PM-bus tracks as possible in the global channel. This involves two steps.

First, all possible rings of the DACS elements must be found. This problem is equivalent to the problem of finding all Hamilton cycles in the corresponding deBruijn graph. This is treated in [16], in which algorithms for generating the total number of $2^{2^{n-1}-n}$ different cycles for an nth order graph can be found together with algorithms for generating only a fraction of the total number. Interestingly enough $n = 5$ (corresponding to 32 DACS elements) results in a total of 2048 different rings, which is the largest number where an exhaustive evaluation is possible.

Second, every ring can be placed in a number of ways on the two-column structure. For the 64-node Viterbi decoder having 16 pairs of DACS elements in each column, 32 possible placements are obtained by cyclically shifting the elements one place at a time. Because of symmetry this yields 16 different channel routing problems for each ring.

The placement of ACS elements that is shown in Fig. 8(a) is selected from one of the 2048×16 possible placements that requires the fewest vertical tracks. We used a simple routing procedure allowing only one vertical segment per net, and we obtained a number of solutions using 11 vertical tracks, and a worst case of 24 tracks.

C. Discussion and Generalization

In our design the global channel accounts for one-fourth of the wiring area, or approximately 10% of the total area (see Table II). If the number of processing elements is changed, these *relative* measures will only be affected by the number of tracks in the global channel (and not by the height of the structure). This means that the fraction of the area required for wiring is relatively insensitive to the number of processing elements. We therefore conclude that the two-column topology is applicable to problem sizes of 16–128 processing elements, depending on the area and form factor of the processing elements.

As mentioned earlier, it is not possible to generate all possible rings in a reasonable time if the number of nodes exceeds 32. Neither would it be possible to solve all possible channel routing problems. Because a few extra tracks add very little to the total area, a practical approach is to stop searching when a good (near optimal) solution has been found. This procedure could be based on one of the algorithms in [16] that generates a subset of all possible cycles.

Finally, it should be noticed that exactly the same two-column ring topology can be used when the interconnection

structure is a SE network. All the exchange interconnections will be local to the double elements, and the remaining interconnections are the shuffle interconnections, i.e., exactly the same as in the Viterbi decoder. In fact, the topology can be used for all deBruijn graphs in which the successors of every node are neighbors.

V. Implementation of the Viterbi Decoder Chip

A. Algorithm Implementation Considerations

Since the area and time complexity of the ACS module is proportional to the number of bits used to represent the PM's, it is desirable to keep this number small [1, pp. 258–261], [17]. Although 5-b PM's are sufficient for the $R=1/2$ code [14], we use 7-b PM's and 4-b BM's with rescaling when the minimum PM exceeds 32, because it allows for extensions to $R=1/3$ or $R=1/4$ codes, and because 5-b PM's require addition with saturation, a slightly more complicated rescaling technique, and special precautions in relation to the path storage and output sequence selection block. With the bit-slice implementation described below, reduction to 5-b PM's is, however, straightforward.

B. Floor Plan

The overall floor plan has already been presented in Fig. 8. The DACS elements have been implemented using a bit-slice organization as shown in Fig. 9. Note that the BM buses and power rails run vertically, through each of the four columns of (single) ACS elements, and are integrated in the bit-slice implementation of the ACS elements. Besides the regularity, this organization allows codes with different generator polynomials to be implemented very easily by reprogramming of the contacts/vias that connect the ACS elements to the BM buses. Notice also that the long PM wires have been divided into three sections in a very straightforward way, by placing buffers between the global and the local channels.

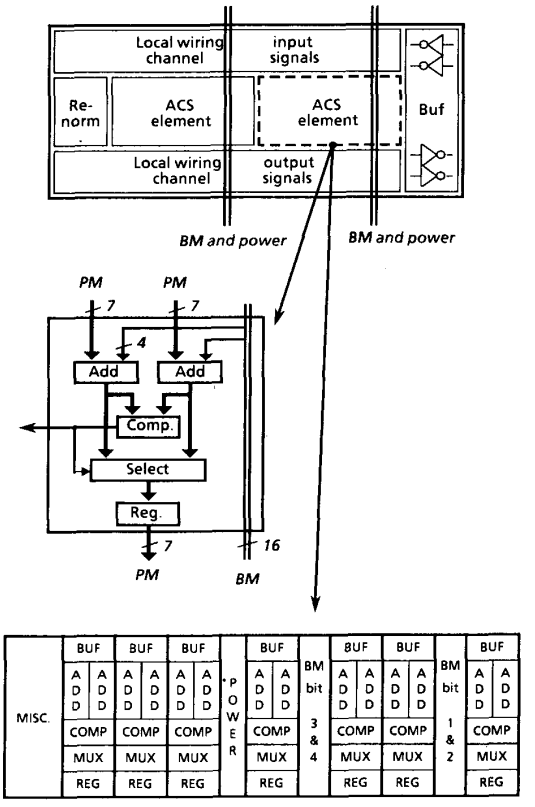


Fig. 9. Floor plan of a DACS element.

C. Circuit Implementation

The circuit design is straightforward, and except for the global rescaling calculation, we use static circuits. Each ACS element contains an 8-b register: 7 b for the PM and 1 b for the decision. The adders and comparators use ripple carry propagation [18, figs. 8.2, 8.4, 8.5]. This is a simple but fast solution, because the comparator starts calculating as soon as the least significant bits of the sums are calculated. The total delay equals nine circuit stages from inputs to decision (the carry and sum stages in the least significant bit of the adder and the seven carry stages of the comparator).

The multiplexor implementing the select operation of the ACS function has been extended to include a clear function and a test function that connects the PM and decision registers into a scan path. This is controlled by a 2-b function code.

The rescaling mechanism is implemented using a global precharged wire that any ACS element can decharge if its PM is less than 32. The signal is input to a register in each ACS element, and this register drives the three most significant bits of the BM operand to the adder.

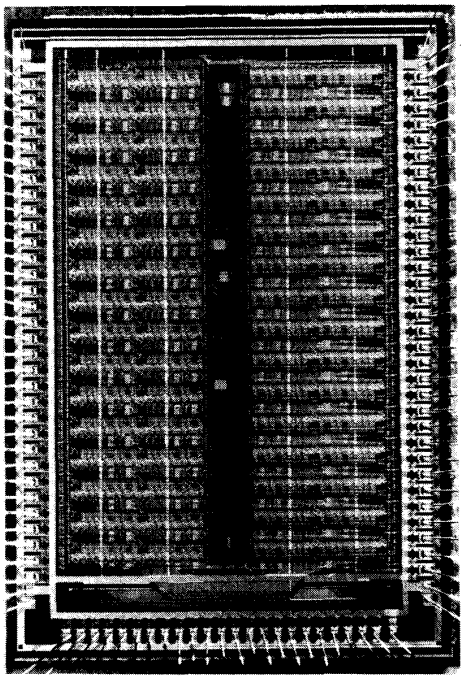


Fig. 10. Microphotograph of the complete ACS module.

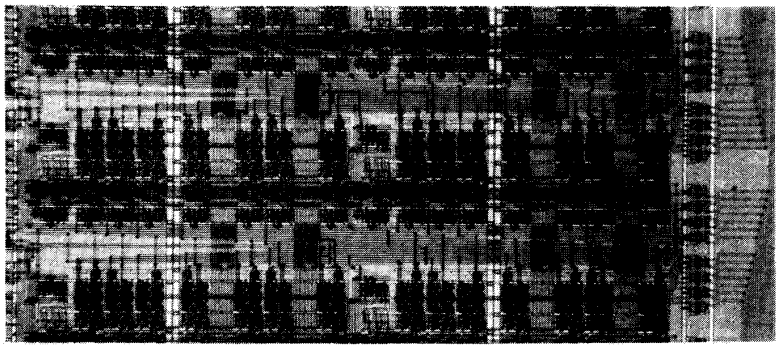


Fig. 11. Microphotograph of a DACS element from the left column.

TABLE I
DIMENSIONS OF THE LAYOUT OF THE ACS MODULE

Dimensions (in micrometers)	height	width
Complete ACS module	7750	5097
Global PM channel	7750	491
Column of 16 DACS elements	7750	2303
DACS element incl. local channel	483	2303
Local PM channel	126	2167
Single ACS element	357	1015

TABLE II
PERCENTAGE WIRING AREA IN THE ACS MODULE

Global PM channel	9.6%
Local PM channels	22.0%
BM and power "slices"	10.5%
Total	42.1%

D. Physical Implementation and Test Results

The chip containing the described ACS module has been designed and laid out using the MAGIC design system [19] and NORCHIP's simplified design rules for a 2-μm-gate, double-metal, CMOS n-well process [20]. These rules are quite similar to MOSIS revision 6 with simpler contacts (6×6 μm).

The chip was fabricated via NORCHIP (the Scandinavian CMOS IC prototype implementation service) on the May 1989 run at Austria Mikro Systeme International GmbH. A microphotograph of the complete chip (for comparison with Fig. 8) is shown in Fig. 10, and a more detailed photo showing a section of the left column is shown in Fig. 11 (for comparison with Fig. 9). A number of area measures relating to Figs. 8–11 are listed in Tables I and II.

The chip works correctly and runs at speeds up to 19 MHz under worst-case conditions ($V_{DD} = 4.75$ V and $T_A = 70$°C). Except for bus drivers, all transistors in the design are identical: channel length is 2 μm and channel width is 6 μm. The speed could therefore be increased significantly by transistor sizing. Simulation studies have shown that the delay in the ripple carry adders can be reduced by 40% by proper sizing of the propagate transistors.

The total number of transistors used to implement the ACS module is 50500, corresponding to a density of 2100 transistors/mm^2 in the 57.9% of the area not occupied by wires. The design contains approximately 200 individually designed transistors resulting in a regularity factor of 250.

E. Comparison with Existing Designs

The NTT-SST chip [5] has been implemented in a 1.5-μm, three-layer-metal, CMOS process. The ACS elements are hand-designed and use 6-b PM's. The area of the ACS module is 32 mm^2 and the maximum operating frequency is 23 MHz.

The Qualcomm Q1401 chip [6] has a maximum operating frequency of 17 MHz. No implementation details have been published, but we understand it to be a sea-of-gates design with a die size of approximately 13×13 mm^2, implemented in a 1.5-μm, two-layer-metal, CMOS process. We expect the area of the ACS module to account for approximately 50% of the core area corresponding to around 60–70 mm^2.

For a comparable implementation of our design using a 1.5-μm, two-layer-metal, CMOS process, we estimate an area of 22 mm^2 and a speed of 30 MHz. This assumes a simple all-layer shrink that reduces all dimensions by a factor of 0.75 and sizing of the propagate transistors in the adders.

Although differences in technology, design style, and PM number range may make a comparison somewhat misleading, the trend is quite clear: the area of our two-column ring based implementation is considerably smaller and the speed is of the same magnitude. The latter is not surprising because all of the designs implement almost the same function.

VI. CONCLUSIONS

We have presented a two-column ring topology that can be applied to the implementation of parallel Viterbi decoders as well as other computing structures based on shuffle-exchange type interconnection networks with a small number of processing elements (i.e., 32–128). Together with the bit-slice organization of the processing elements, the topology constitutes an area-efficient, practical, and regular layout structure.

We have used this structure in a full-custom implementation of the add–compare–select module of a fully parallel $K = 7$, $R = 1/2$ Viterbi decoder. The chip has been fabricated in a 2-μm, double-layer-metal, CMOS process, and the area of the ACS-module itself is 39.5 mm^2. The PM interconnection network occupies 32% of the area and the global

signals occupy a further 10%. The design is based on simplified design rules and identical and unsized transistors, and the chip operates at 19 MHz under worst-case conditions ($V_{DD} = 4.75$ V and $T_A = 70$°C).

For a similar implementation fabricated in a 1.5-μm process, we estimate an area of 22 mm^2 and a speed of 30 MHz. In relation to existing comparable (commercial) chips, this represents a significantly smaller area and a similar speed. Finally, we mention that the area of our two-column structure is significantly smaller than the area of a shuffle-exchange graph based implementation.

Acknowledgment

The authors wish to thank J. Staunstrup and R. I. Sharp for their continuous encouragement and valuable suggestions.

References

[1] G. C. Clark, Jr. and J. B. Chain, *Error-Correcting Coding for Digital Communications*. New York: Plenum, 1981.
[2] P. G. Gulak and E. Shwedyk, "VLSI structures for Viterbi receivers: Part 1. General theory and applications," *IEEE J. Selected Areas Commun.*, vol. SAC-4, no. 1, pp. 142–154, Jan. 1986.
[3] R. M. Orndoff *et al.*, "Viterbi decoder VLSI integrated circuit for bit error correction," in *Proc. 16th Annual Asilomar Conf. Circuits Syst. and Computers*. New York: IEEE Computer Soc., Nov. 8–10, 1982, pp. 149–152.
[4] J. B. Cain and R. A. Kriete, "A VLSI $K = 7$, $R = 1/2$ Viterbi decoder," in *Proc. Nat. Aerospace and Electronics Conf.* (Dayton, OH), 1984, pp. 20–27.
[5] T. Ishitani, K. Tansho, N. Miyahara, S. Kubota, and S. Kato, "A scarce state transition Viterbi decoder for bit error correction," *IEEE J. Solid-State Circuits*, vol. SC-22, no. 4, pp. 575–582, Aug. 1987.
[6] *Q-1401 Single-Chip Viterbi Decoder*, Qualcomm Inc., San Diego, CA, Sept. 1987.
[7] Consultative Committee for Space Data Systems, *Recommendation for Space Data System Standards, Telemetry Channel Coding*, CCSDS 101.0-B-2 Blue Book, Jan. 1987.
[8] P. G. Gulak and T. Kailath, "Locally connected VLSI architectures for the Viterbi algorithm," *IEEE J. Selected Areas Commun.*, vol. 6, no. 3, pp. 527–537, Apr. 1988.
[9] F. T. Leighton and G. L. Miller, "Optimal layouts for small shuffle-exchange graphs," in *Proc. VLSI81*, J. P. Gray, Ed. New York: Academic, 1981, pp. 289–299.
[10] H. J. Siegel, *Interconnection Networks for Large-Scale Parallel Processing — Theory and Case Studies*. Lexington, MA: Lexington Books, 1985.
[11] J. D. Ullman, *Computational Aspects of VLSI*. Rockville, MD: Computer Science Press, 1984.
[12] D. Kleitman, F. T. Leighton, M. Leapy, and G. L. Miller, "New layouts for the shuffle-exchange graph," in *Proc. 13th Annual ACM Symp. Theory of Computing*, May 1981, pp. 278–292.
[13] C. D. Thomson, "A complexity theory for VLSI," Ph.D. dissertation, Carnegie-Mellon Univ., Pittsburgh, PA.
[14] K. Hansen and J. H. Jensen, "Viterbi decoding at speeds above 10 Mbit/sec," M.Sc. thesis (ID-E 464), Dept. Computer Sci. and Inst. for Telecommun., Tech. Univ. of Denmark, Lyngby, 1989.
[15] S. W. Golomb, *Shift Register Sequences*. San Francisco, CA: Holden-Day, 1967.
[16] H. Fredricksen, "A survey of full length nonlinear shift register cycle algorithms," *SIAM Rev.*, vol. 24, no. 2, pp. 195–221, Apr. 1982.
[17] A. P. Hekstra, "An alternative to metric rescaling in Viterbi decoders," *IEEE Trans. Commun.*, vol. 37, no. 11, pp. 1220–1222, Nov. 1989.
[18] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design — A Systems Perspective*. Reading, MA: Addison-Wesley, 1985.
[19] W. S. Scott, R. N. Mayo, G. Hamachi, and J. K. Ousterhout, Eds., *1986 VLSI Tools: Still More Works by the Original Artists*, Computer Sci. Div. (EECS), Univ. of Calif., Berkeley, Rep. UCB/CSD 86/272, Dec. 1985.
[20] "NORCHIP—Design rules for CMOS, 2 micron gate, double metal, N-well process," Nordic VLSI Inc., N-7079 Flatasen, Norway, rev. Oct. 1988.

Jens Sparsø was born in Silkeborg, Denmark, on December 19, 1955. He received the M.S. degree in electrical engineering from the Technical University of Denmark, Lyngby, in 1981.

Since 1982 he has been with the Department of Computer Science at the Technical University of Denmark, where he became an Associate Professor in 1986. He is teaching courses on VLSI and digital systems design, and his research interests are architecture and design of VLSI systems, i.e., design methods, circuit techniques, and the interplay between technology and system architecture.

Henrik N. Jørgensen was born in Roskilde, Denmark, on May 24, 1962. He received the M.S. degree in electrical engineering from the Technical University of Denmark, Lyngby, in 1988.

From 1988 to August 1990 he was with the Department of Computer Science at the Technical University of Denmark, working as a Research Assistant on the Viterbi decoder project. Since August 1990 he has been at ElektronikCentralen, Hørsholm, Denmark, in the Microelectronic Test Center. His main interests are digital systems design, including custom VLSI and high-speed components.

Erik Paaske was born in Odder, Denmark, on April 14, 1938. He received the M.S. and Ph.D. degrees in electrical engineering in 1963 and 1969, respectively.

From 1963 to 1965 he was employed by F. L. Schmidt and Company A/S working with factory automation. Since 1965 he has been at the Institute of Circuit Theory and Telecommunication, the Technical University of Denmark, Lyngby, from 1969 as an Associate Professor teaching and doing research in information theory and coding.

Steen Pedersen was born in Torslev, Denmark, on November 1, 1955. He received the M.S. degree in electrical engineering in 1980 and the Ph.D. degree in applied physics in 1986, both from the Technical University of Denmark (DTH), Lyngby.

From 1980 to 1985 he was a Research Assistant at the Laboratory of Applied Physics at DTH. From 1986 he has been with the Department of Computer Science at DTH, where he became Associate Professor in 1987. He is currently teaching courses on digital design and design methodology for digital ASIC's. His research interests are design of digital systems in VLSI technology and test of digital ASIC's.

Thomas Rübner-Petersen was born in Denmark on June 4, 1941. He received the M.S. degree in electrical engineering from the Technical University of Denmark (DTH), Lyngby, in 1966.

From 1966 to 1970 he was employed at the Laboratory of Basic Electronics, DTH, and from 1970 until his death on June 9, 1990 he was at the Institute of Circuit Theory and Telecommunication, DTH, where he became Associate Professor in 1972. His main interests were circuit analysis, multiprocessor systems, speech synthesis, and coding theory.

Mr. Rübner-Petersen received the Alexander Foss Gold Medal in 1977 for his development of the NAP2 circuit analysis program.