SELF-TIMED APPROACH FOR REDUCING ON-CHIP SWITCHING NOISE

Johanna Tuominen, Pasi Liljeberg, and Jouni Isoaho

Electronics and Communication Systems
Dept. of Information Technology
University of Turku, Finland
{joeltu | pakrli | jisoaho}@utu.fi

ABSTRACT

This paper describes a design methodology to reduce on-chip power supply switching noise caused by current peaks related to synchronous clock. In the proposed approach, a synchronously operating system module is modified so that the clock based control is replaced by a self-timed one. At the same time the external interface is kept intact, without the need for synchronizers. The method is applied to the path metric unit of the Viterbi decoder. This reduces the peak current by 87 % compared to the fully synchronous design. Furthermore, this approach allows the reduction of the area devoted to the on-chip decoupling capacitance needed to suppress power supply noise.

1. INTRODUCTION

As technology scales down into the deep submicron regime and the size of chips grow larger the noise immunity will be one of the most important design metric [12] to system design. It is more difficult to manage different types of noise, such as power supply, crosstalk, and leakage noise, because of the continuous reduction of supply and threshold voltages [1]. If noise is not handled properly it will introduce additional signal transition delays and might even cause false switching leading to unreliable operation of the circuit. Power supply noise or unwanted fluctuation of the supply voltage within a digital ULSI chip mainly originates from simultaneous clock-induced switching of CMOS circuits which causes high peak current draws from the power source. The total power supply noise is the sum of IR voltage drop and the inductive switching noise $\Sigma L\Delta I/\Delta t$ where L and R are the effective wire inductance and resistance, respectively [3]. The ΔI is the total current change during the rise and fall time Δt of the concurrently transitioning signals. Hence, dealing with high current peaks, rather than average current, is a key issue to dealing with noise induced by power distribution network.

One way to counter attack the power supply noise is to use more and more on-chip decoupling capacitors to minimize voltage fluctuations. However, the area needed for decoupling capacitors increases with the size and complexity of the chip making this approach less attractive. More attractive approach is to focus directly to the source of current spikes. There is not much that can easily be done with parasitic resistances and inductances or the total amount of switching. However, it is possible to decrease the number of simultaneous events so that the current peaks will be lower, and consequently decrease noise. This is done by tuning the timing of the circuit using self-timed logic so that the switching of registers and logical operations can be time-interleaved. The clock lines are replaced by asynchronous communication links, which are used to control the registers. In fact, combinational logic and registers can be designed and implemented as synchronous ones, only the clock line is controlled asynchronously.

In this paper we apply such a method to the path metric unit of the Viterbi decoder. The eight processing elements of the path metric unit are controlled in a self-timed manner so that their internal operation is time-interleaved. At the same time the functionality of the path metric unit is kept intact as well as external interface. Furthermore, three different self-timed inter-processor communication schemes are studied in order to decrease current spikes caused by interconnects between processing elements.

2. VITERBI ALGORITHM

Error correcting codes are commonly used in transmission of digital data due to their ability to reduce the error probability [10]. These codes operate by adding redundancy into signal so that some of the errors that occur during transmission can be eliminated in the decoder. Convolutional codes are one of the major families of the error correcting codes. A Viterbi decoder is used to decode convolutionally encoded data [2]. The Viterbi algorithm can be described as an algorithm which decodes the most likely data values through a trellis from given set of observations. The trellis in this case represents a path of a finite set of states from a finite state machine [8]. This relation is illustrated in Figure 1. Each node represents a state and each edge represents a possible transition between two states at discrete time intervals. The algorithm uses a set of path metrics to describe the various costs of different paths through the trellis. These metrics are used to decide which path is the most likely path to follow. It was shown by Viterbi [10] that the likelihood function used to determine the shortest path can be reduced to a minimum distance measure, known as a Hamming distance. The Hamming distance can be defined as a number of bits that are different between the observed symbol at the decoder and the sent symbol from encoder.

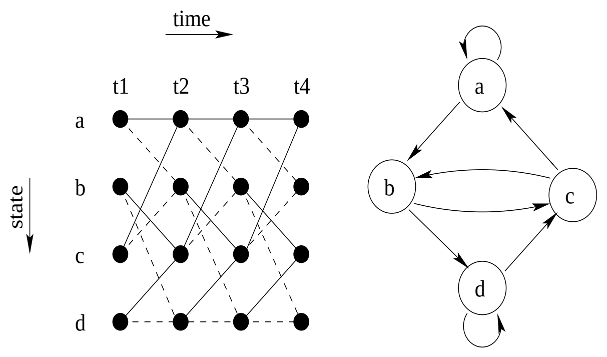

Fig. 1. Trellis spread over time and the corresponding state diagram of the FSM.

The functionality of the Viterbi algorithm can be described as follows. For all transitions at a time t into a particular state, it decides which of them was the most likely to occur by selecting the smallest path metric value. If two or more transitions have an equal path metric value the decision is made randomly. The best metric is then assigned into the states survivor path and other metrics are discarded. The survivor path is generated by adding the previous path metric at time $t - 1$ into the present metric at time t. In the end of the metric calculation the survivor path is determined as before but the selection of the shortest path has to be made. This is carried out by choosing the survivor path with the smallest path metric value and again if one or more paths have an equal smallest value the most likely path will be chosen randomly.

3. SYSTEM ARCHITECTURE DESIGN

The path metric unit (PMU) is the core of the computation in the Viterbi decoder and therefore it dominates the overall power consumption. For instance with a random input bit sequence the power dissipation of the PMU can be as high as 90 % of the overall power consumption [2]. This motivates to concentrate design efforts to the PMU in order to smoother the current profile of the circuit.

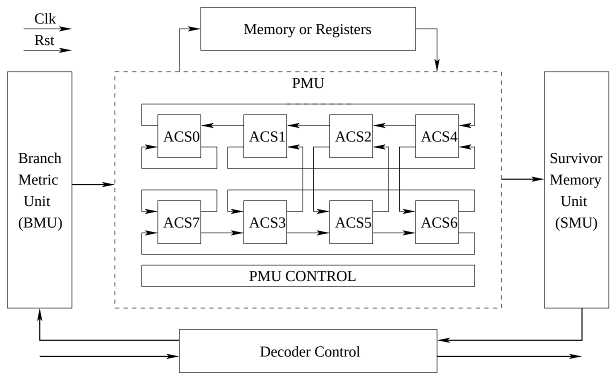

Fig. 2. The structure of the Viterbi decoder

3.1. Synchronous Architecture

Synchronous decoder is implemented using hard-decision decoding [11]. It uses one bit quantization on the received bits and Hamming distance [8] to update the path metrics. The architecture of the decoder is illustrated in Figure 2. The branch metric unit calculates the branch metrics according to the Hamming distance method. These metrics are sent to the path metric unit, which consists of eight ACS processing elements and a control unit [11]. Each of those processing elements contains two ACS cells which all have similar functionality with three different basic operations: addition, comparison and selection. One ACS cell contains two adders, a comparator, a path metric out unit, and a selector. The path metric out unit is used to store the results of the comparison at the current counting cycle and to transfer data. The selector chooses the smallest path metric value and counts the corresponding state. The interconnects between processors are used to transmit every clock cycle the previous path metric values which are needed to calculate the new ones. At every rising clock edge each of those processing elements calculates new path metric values and decision values. The survivor memory unit is used to store the decision values, which are the results of the ACS counting. The trace back operation reads these values backwards and outputs the original bit sequence.

3.2. Method for de-synchronization

The purpose of de-synchronization is to decrease the number of simultaneous switching events so that the current peaks will be lower and consequently decrease power supply noise and electromagnetic interference (EMI) [6]. This is done by replacing the clock lines with asynchronous communication links that are used to control simultaneous events in the system. The method does not require to re-design of the entire system, it is necessary only to add a self-timed control, re-route existing clock lines, and in some cases add register levels. Before applying the method, sources of highest current peaks needs to be identified. This can be done by investigating switching activity of the circuit and locating power consuming parts of the circuits such as arithmetic units, pipelines, interconnect drivers, etc. A more reliable way is to use some place and route tool with a support of graphical representation of voltage drops in different locations of the chip.

A system module containing several processing elements is de-synchronized in two phases. In the first phase the internal blocks of processing elements that do not have input data dependencies need to be identified. Clock lines of those blocks can be controlled in a self-timed manner so that their operations are time-interleaved. In the case that there are input data dependencies or the data originates from two or more already time-interleaved sources, a register level needs to be inserted in front of a block. With this possible redundant switching of combinational logic caused by different data arrival times can be avoided. The height of current peaks can be adjusted by changing the granularity during partitioning into blocks. With a coarse grain granularity processing element are partitioned only to a few time-interleaved blocks and the size of the self-timed control will be small. When a more smoother current profile is required the element need to be partitioned into several blocks. On the other hand, with a finer granularity the timing analysis and verification of correct operation will be more complex as well as the size of the self-timed control increases. In the second phase the data dependencies and timing requirements of processing elements are analyzed. The procedure is similar as with internal circuit blocks, clock lines are replaced by asynchronous handshake channels. Only exception is the control of inter-processor communication which cannot be ignored. This is due to parasitic properties of interconnects between processing elements with considerable load and power consuming drivers. This will be discussed in Section 4.

3.3. Self-timed structure of the path metric unit

The PMU of the Viterbi decoder was synchronized following the guidelines given in previous section. The external interface and the original timing restrictions of the self-timed PMU are equal to the synchronous one. Furthermore, the number of processing elements and the inter-processor connections were kept the same as they were in the synchronous implementation. The structure of the self-timed PMU is shown in Figure 3. The synchronous decoder operates with 25 MHz clock which sets the limit for the duration of one counting cycle in the self-timed structure. The counting cycle is the time period that it takes to calculate and output one decoded bit. The number of counting cycles depends on the number of states used. One decoded bit requires 10 cycles for 64 states decoder and 38 cycles for 512 states decoder.

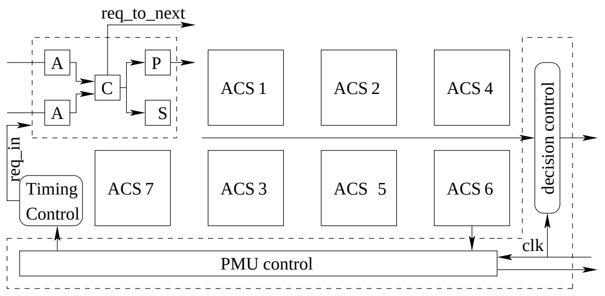

Fig. 3. The structure of the self-timed PMU

The self-timed PMU can be divided into two parts: synchronous and asynchronous. The synchronous part consist of PMU control and decision control modules. The PMU control unit acts as an interface between synchronous and asynchronous domains. It controls the handshake logic that activates the calculation and communicates with other parts of the decoder. The decision control unit is used to collect the decision values that arrive asynchronously. The reading operation of the decision values is clocked because it sends the decision values to the synchronous survivor memory unit.

The asynchronous part consist of the eight ACS processing elements and a timing control. Timing control is used to interleave the operation of the ACS processing elements via self-timed communication channels. Each of those elements are divided to four separate stages. The stages are controlled in a self-timed manner so that their operations are time-interleaved. The counting cycle starts by activating two adders, the first two stages, at slightly different time. After completion of both additions the comparator and the next processing element are activated. Hence, comparison, third stage, and additions in the next processing element are executed in parallel. Last stage, selection and transfer of the new path metric value, is activated immediately after comparison. Meanwhile stages at the different processing elements are activated in a domino like fashion. After all ACS processing elements have completed their tasks the timing control is acknowledged and a new counting cycle can begin.

4. INTER-PROCESSOR COMMUNICATION

As circuit density increases and technology scales down the parasitic effects of interconnect wires will have substantial impact to power consumption, noise, and speed [9]. The effect of capacitive and resistive parasitics increases with the length of the interconnect and consequently larger drivers are needed to drive those interconnects. In a system with several interconnects a significant portion of total power is dissipated in data transfers between different modules [4]. The simultaneous data transfer along capacitive interconnects in such a system causes high peak current draws from the power supply [7]. In this study the interprocessor communication was implemented with three different methods. The purpose was to find a method that reduces most the peak current values caused by driving the interconnects between the processing elements.

The first implementation is asynchronous bundled data approach where the interconnects are controlled using four-phase handshake protocol [5]. The term four-phase refers to the number of communication actions: Firstly the sender sets request high which informs the receiver that data is valid. Secondly the receiver absorbs the data and sets the acknowledgement high. Thirdly the sender responds by setting the request low which means that data is no longer valid. Fourthly the receiver acknowledges this by setting the acknowledgement signal low. In this implementation it is assumed that the sender is the active party which initializes the communication cycle. In this implementation certain amount of delay is needed between two consecutive events to make sure data has enough time to stabilize. The difference to the synchronous interconnects is that these interconnects do not switch simultaneously. Processing element activates the data transfer every time when new data has become valid. At the receiver the data is collected into register in order to avoid overlapping between the new data and the current data.

In the second implementation data bits are transmitted in a time-interleaved fashion. A message is partitioned into four 4-bit groups which are transfered at a slightly different times with respect to each other. Hence, the power hungry bus drivers will not switch exactly at the same moment of time, instead only a set of drivers will switch simultaneously. This considerably reduces the peak current draw and therefore the switching noise is decreased. In this approach n-bit message requires n wires for data and 2 wires for the handshake signals. Obviously, this method sacrifices the performance to the increased noise margin. However, by keeping the delays between bit groups relatively small, considerably reduction in noise can be achieved with a minor performance loss.

The third implementation utilizes delay insensitive four-phase dual-rail encoding. This protocol encodes the request into data signals using two wires per bit. A data bit is encoded onto two wires; transmitting n-bit data then requires $2n + 1$ wires, $2n$ for data and one for acknowledge. 4-phase dual-rail encoding has three legal states: ’00’ for idle, ’10’ for valid zero and ’01’ for valid one. The combination ’11’ is illegal state. Transmission of a bit requires transition from the idle state to either the valid 0 or the valid 1 state. After the sender has received the acknowledge, it must made transition back to the idle state.

5. ANALYSIS

Physical analysis was performed using Cadence design tools with two different technologies, 0.35 μm and 0.18 μm. After synthesis a verilog netlist and a general constraint format file was generated for the Silicon Ensemble place and route tool. This was necessary to get information from the capacitive loads of the interconnects. In order to get comparable results the row utilization was 85 % in each implementation. The interconnects between the processing elements were modeled using the capacitive and resistive loads obtained from the place and route analysis. The current profile analysis was performed to the PMU consisting of 64 states. The duration of the PMU simulation was rather long therefore with 512 states only the interconnects were simulated.

5.1. Current profile

The motivation for reducing the peak current values is illustrated in Figure 4. During the active phase of the processing elements in the synchronous design the clock related current peaks are as high as 1.2 A. Furthermore, the idle time between the calculation causes peak current values around 1 A. Hence in a synchronous system many flip-flops switch without having an actual input to process because they are connected to the clock.

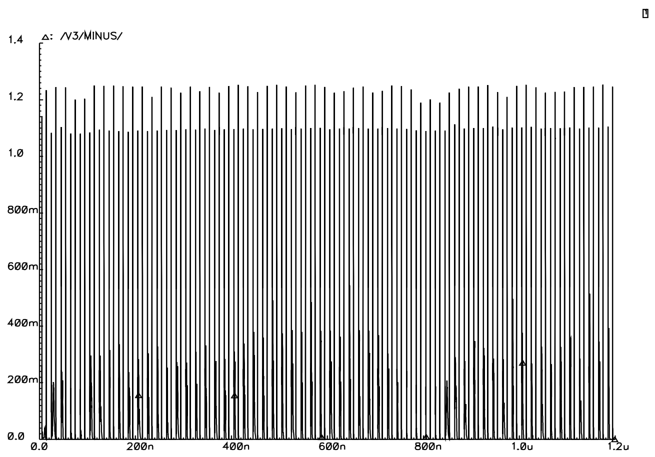

Fig. 4. Current profile of the synchronous 512 states PMU

The average capacitive loads of the interconnects are from 100 fF to 240 fF in the 0.35 μm technology. The corresponding average resistive loads are from 5 Ω to 10 Ω. Above are the minimum and the maximum values obtained from analysis performed after place and route which are a function of the interconnect length. Therefore the values used to model the interconnects in the current profile analysis are spread between above values. The 64 state PMU was analyzed with 0.35 μm technology using three different inter-processor communication methods between the processing elements as explained earlier. Each bit of the 16-bit wide interconnects was modeled separately with its actual capacitive and resistive load.

Corresponding analysis was performed with 0.18 μm technology. The average capacitive and resistive loads of the interconnects were from 37 fF to 72 fF and from 39 Ω to 75 Ω respectively. The separate interconnect analysis was performed to illustrate the effect of technology scaling. The effect of the technology scaling was a decrease in capacitance values and an increase in resistance values when compared towards the 0.35 μm technology, as can be expected.

The current profile of the 64 state PMU with 0.35 μm technology is shown in Figure 5. During the active phase of the processing elements, the current peaks are around 550 mA which is only half of those in the corresponding 512 state implementation. Similarly with the 512 state PMU the decrease is minor during the idle period of the processing elements. With the self-timed implementation using asynchronous interconnects the current peaks are reduced by 75 % during the active phase. If applied to the 512 state implementation the peak values during counting should decrease from 1.2 A to 300 mA.

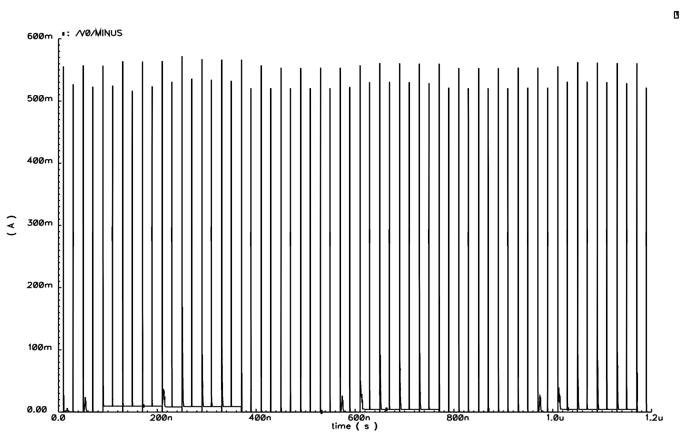

Fig. 5. Current profile of the synchronous 64 states PMU

The peak current values are reduced most when the self-timed PMU is implemented with time-interleaved interconnect method. This current profile is shown in Figure 6. The reduction is 87 %, from 550 mA to 70 mA, during the counting cycle. One of the advantages of the self-timed method can be seen during the idle period between the counting cycles. This is illustrated in Figure 6 as a significantly lowered current peak values, the reduction is 97 %. Above values results from the partitioning of data transfer and the natural support of the power down during the idle period in the self-timed implementation.

The results of the current profile analysis for the 0.35 μm technology are collected in Table 1. These values are measured as peak to peak values. Separate peak current values are presented for interconnects utilizing the loads from the 64 state and 512 state implementations. It shows that all

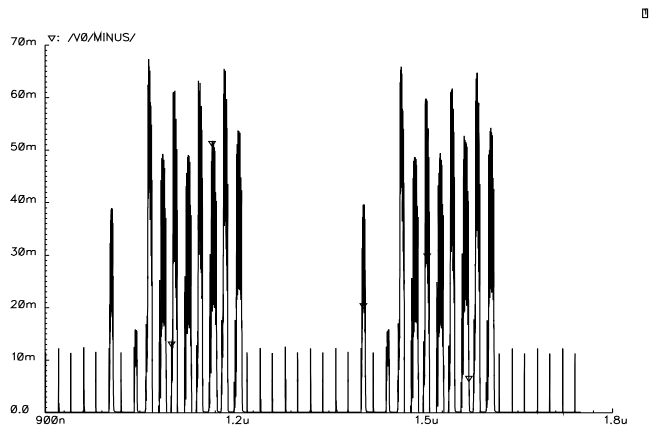

Fig. 6. Current profile of the self-timed 64 states PMU with time-interleaved interconnects

asynchronous implementations provide lower current peaks than the corresponding synchronous one.

Table 1. Peak current values for 0.35 μm technology.

Design	PMU(64)	Int.co.(64)	Int.co.(512)
Sync.	550 mA	36 mA	39 mA
Async.	140 mA	4.5 mA	6.2 mA
Interleaved	70 mA	3.2 mA	3.2 mA
Dual-Rail	80 mA	8.4 mA	9.3 mA

The current profile analysis was also performed with the 0.18 μm technology for the interconnects, shown in Table 2. Compared to the results gained with the 0.35 μm technology the peak current values are significantly lower. The optimal interconnect implementation is the asynchronous time-interleaved data transfer where the message is partitioned into four 4-bit groups. The current peaks are reduced by 92 %, from 16 mA to 1.3 mA, regardless of the number of states used. In this implementation the interconnects were relatively short. Therefore the effect of the interconnects on the current profile was rather small. All of the three design methods reduce the current peaks considerably compared to the synchronous one. The self-timed PMU implemented with the time-interleaved interconnects gives the most optimal result regardless of the technology. Above results show that a large amount of the switching noise can be reduced by using self-timed design approach.

Table 2. Peak Current Values for 0.18 μm Technology.

Design	Int.co.(64)	Int.co.(512)
Sync.	15 mA	16.5 mA
Async.	1.9 mA	3.2 mA
Interleaved	1.3 mA	1.3 mA
Dual-Rail	4.7 mA	4.8 mA

5.2. Area

In addition to above current profile analysis, the impact of the presented self-timed design approach to the area was also analyzed. The area comparisons between the synchronous and asynchronous implementations were made according to the results gained from the synthesis tool. The differences in areas between the two different design approaches with 0.35 μm technology are shown in Table 3. In order to make the area comparison between implementations straightforward, relative areas are used where synchronous 64 state PMU serves as a reference point. In the asynchronous 64 state implementation the area is 21 % larger than the area of the corresponding synchronous system. As the size of the decoder increases the area penalty decreases, with 512 state implementation the difference in total area is 2 %.

Table 3. Relative areas in the 0.35 μm technology.

Number of states	Comb. area	Seq. area	Total area
Sync. 64	1	1	1
Async. 64	1.13	1.42	1.28
Sync. 512	4.07	5.63	4.85
Async. 512	4.48	5.38	4.93

Similar comparison made with the 0.18 μm technology is shown in Table 4. The relative area decreases in every design compared to the corresponding areas from 0.35 μm technology. The asynchronous 64 state implementation is 16 % larger than the corresponding synchronous one. Similarly as with the 0.35 μm technology the area penalties decreases when the size of the decoder increases. In fact the total area of the asynchronous 512 state implementation is the same as it is in the synchronous one.

Table 4. Relative areas in the 0.18 μm technology.

Number of states	Comb. area	Seq. area	Total area
Sync. 64	1	1	1
Async. 64	1.14	1.22	1.18
Sync. 512	3.67	5.37	4.55
Async. 512	4.38	4.70	4.55

The absence of the clock circuitry in the asynchronous implementations saves the area that can be used for sequential logic. However, the amount of combinatorial logic increases due to the self-timed control circuitry. In the 0.35 μm and 0.18 μm technologies the asynchronous implementations with 512 state has about the same total area than the corresponding synchronous one. When the size of the decoder increases the amount of sequential area increases significantly. This lead to the situation where the area needed to the clock circuitry compensates the area needed for the self-timed control circuitry.

6. CONCLUSION

An approach to minimize high current peaks caused by simultaneous clock induced switching of circuits has been introduced that consequently reduces on-chip power supply noise. This approach is based on de-synchronization method which allows to time-interleave the operation of distinct system modules so that current draw from the power supply can be folded to a longer period of time. The path metric unit of the Viterbi decoder was used as a case study to exploit the possibility of reducing current peaks. The reduction of current peaks was 87 % compared to the synchronous version. However, this came with an area penalty of 21 %. The area penalty decreases as the size of the decoder increases and the technology scales down. Reducing noise has an immediate impact, boosting both precision and performance. The case study considered in this paper revealed the possibility to decrease the current spikes using self-timed approach.

7. REFERENCES

[1] H. Bakoglu. "Circuits, Interconnections, and Packaging for VLSI." Addison-Wesley, 1990.

[2] L.E.M. Brackenbury. " Large-Scale Asynchronous Designs: An Asynchronous Viterbi Decoder. In J. Sparso and S. Furber, Editors, Principals of Asynchronous Circuit Design - A System Perspective", Kluwer Academic Publishers, Boston, 2001.

[3] H.H. Chen and D.D. Ling. "Power Supply Noise Analysis Methodology for Deep-Submicron VLSI Chip Design", in Proc. DAC Conf., Anaheim, California, 1997.

[4] W.J. Dally and J.W. Poulton. "Digital System Engineering", Cambridge University Press 1998.

[5] A. Davis and S.M Nowick. "Asynchronous Circuit Design: Motivation, Backround and Methods. In G. Birtwistle and A. Davis, Editors, Asynchronous Digital Circuit Design", Springer, 1995.

[6] S. Hayashi and M. Yamada. "EMI-Noise Analysis Under ASIC Design Environment", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol.19, No.11, November 2000.

[7] P. Liljeberg, I.B. Dhaou, J. Plosila, J. Isoaho and H Tenhunen. "Interconnect Peak Current Reduction for Wavelet Array Processor Using Self-Timed Signaling",in Proc. ISCAS Conf., May 2002.

[8] T.K. Moon and W.C. Stirling. "Mathematical Methods and Algorithms for Signal Processing", Prentice Hall 2000.

[9] J.M. Rabaey, A. Chandrakasan and B. Nikolic. "Digital Integrated Circuits - A Design Perspective, Second Edition", Prentice Hall Electronics and VLSI Series, 2003.

[10] M.S. Ryan and G.R. Nudd. "The Viterbi Algorithm", Warwick Research Report 238 February 1993, University of Warwick, England.

[11] M. Savolainen. "Reusable Viterbi Decoder Implementation", M. Sc Thesis, Tampere University of Technology, October 2001.

[12] K.L. Shepard and V. Narayanan. " Noise in Deep Submicron Digital Design ", in Proc. ICCAD Conf. 1999.