Low Cost & Fast Turnaround: Reconfigurable Graph-Based Execution Units

J. Smit, M. Stekelenburg and C.E. Klaassen, University of Twente, Dept of EE
S. Mullender, G. Smit and P. Havinga, University of Twente Dept of INF

Abstract

New devices with the efficiency of full-custom designs and the programmability of FPGAs will ease many aspects of the design of complex systems, without the high cost of mask production. The possibility of in-circuit programming and even dynamic reconfiguration offers great advantages over the traditional design approach. One instance of a fully programmable architecture which offers a platform for rapid prototyping, quick market and application evaluation, is introduced in the form of a field programmable function array (FPFA). The design of such a device is extremely challenging as the aspects of physical design for speed and low-power, the construction of an ALU which is optimal for as many applications as possible, as well as highly efficient mappings of algorithms, are extremely important for a successful device which suits many applications. This paper introduces the reader with the concept of reprogrammable devices with graph-based execution of arithmetic expressions, the corresponding principles of operation, the aspects of low-power operation of the proposed design, the corresponding physical design of the ALU, algorithmic mappings of systems on a chip and the performance aspects compared to other architectures and implementations.

1. Introduction

The design of high performance application-specific VLSI circuits is costly and prone to errors. This is why application specific full-custom design is mainly restricted to consumer products. However advanced innovative applications not related to mass market production, tend to get too little attention or are even fully neglected as a market opportunity due to lack of experienced designers, the high initial cost due to the need of mask-production, the uncertainty of the success of a new standard, and the uncertainty of the expected market-share. Substantial losses in the initial product development cycle are hence quite usual, due to overestimation of the market pull and/or late market introduction. Price erosions, on the other hand, make it necessary to update the high performance product each time that a new process is introduced, i.e. each 1.5 year, even if the volume is not reached. An additional unfortunate complication is that almost none of the high volume designs can be scaled in performance, in a way similar to a microprocessor, to take full advantage of a new process, due to the adoption of a clock-synchronous design style in which the clock-rate is directly proportional to the rate of the incoming signal (e.g. Radio or TV). The implication is that the arithmetic blocks, busses, memories and the controller(s) have to be reconfigured/redesigned in order to obtain a cost-effective solution in the new process. The time to market gets fully out of hand as the chips grow in complexity [7], hence it becomes extremely difficult in the near future to follow the market trend when the current full-custom design-style is to be kept as is. The existing practice in the area of logic design is quite different. The introduction of Field Programmable Gate Arrays (FPGAs) has given the designer a rather fast turnaround time at extremely low cost levels using the latest high performance process. Very few designs are converted into a dedicated chip, as the FPGAs offer in most cases the most cost effective solution. The use of compilers makes it possible to migrate a design with relatively little effort to a new process, without any need to reconsider a redesign nor to worry about testing. The extension of the FPGA approach to the level of arithmetic operations is getting more attention nowadays. For instance the reconfigurable pipelined datapath (RaPiD[5]) design merges multipliers with FPGA primitives to implement for instance systolic functions. Another development is found in run-time reconfigurable FPGAs ([4]), which implement an increased functional density, through on the fly reprogramming of the logic network. This concept works only when the state of the computation can be kept in a RAM. The reconfiguration within the RaPiD chip may work well, provided that a solution can be found for the efficient saving of the state in the pipeline used in the systolic algorithm. The realization of a cascadable multiplier primitive takes 10 to 30.000 times as much energy [6] as a dedicated multiplier, when realized with an energy efficient ROM and slightly more energy, when realized with a RAM, using word-lengths ranging from 2 to 8 bits. FPGA-based realizations of arithmetic blocks which use LUTs to implement their logic behavior, have hence a comparable overhead when realized in the same process. This shows that multiplication is energy inefficient in general on an FPGA. An exception should be made for a multiplication with a constant with a wordlength ranging from 6 to at most 8 bit. Hence it can be concluded that FPGAs are too power inefficient for the implementation of general purpose algorithms. Power consumption is however one of the main issues when over one hundred ALUs with a 16\Theta16 multiplier core are to be operated at a target clock-rate ranging from 40 to 200Mhz. The aspects of low power consumption, high throughput minimal wiring and excellent utilization of chip real-estate, have been balanced in the highly optimized FPFA design, described in this paper. The paper shows the status of ongoing research which was initiated late 1996 with an MSc assignment [1] targeted at the aspects of the design of a high performance re-programmable device which could be used to execute many algorithms, including an instance of the Super Resolution Volume Rendering Algorithm [3].

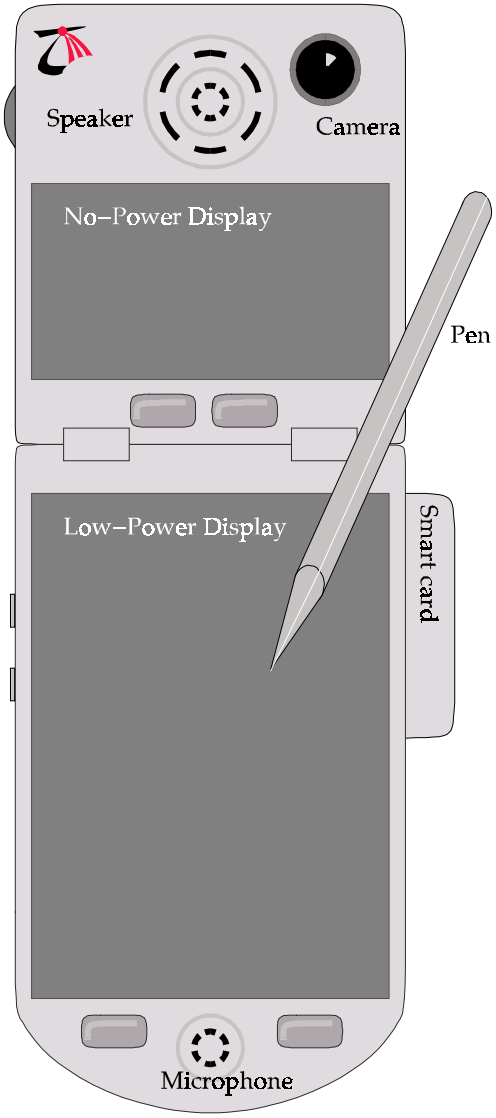

Figure. 1. The proposed personal communicator device with its user interfaces

A second M-Sc. assignment [2] focused on the detailed design using VHDL of the ALU and a comparison with highly optimized Wallace tree multiplier designs based on full-custom layouts. The current focus is on completeness of the ALU, programmability, wireability and ALU utilization for a wide class of applications, as a function of the interconnect between operators, multiplexers and busses within the ALU blocks. These aspects are studied using typical algorithmic benchmarks taken from a wide range of applications. One of the innovative products, which we are pioneering, is the personal communicator shown in Figure 1. It uses a high speed radio-link within a micro-cellular radio system. The Orthogonal Frequency Modulation/Demodulation protocol uses an FFT to make the link insensitive to bit errors due to multi-path reception. The proposed high-speed link can be used to transmit audio signals with voice as well as music quality. The additional capability to transmit real-time graphics and/or TV sequences, as well as encrypted business information including e-cash makes the digital communicator really multipurpose. Even applications which are not known and/or standardized today can be implemented using on the fly re-programmability. The successful design of a reconfigurable CMOS platform will certainly lead to a large amount of applications in almost any part of the industry, including telecommunications, audio, multimedia, transport, aviation, banking, etc. etc.

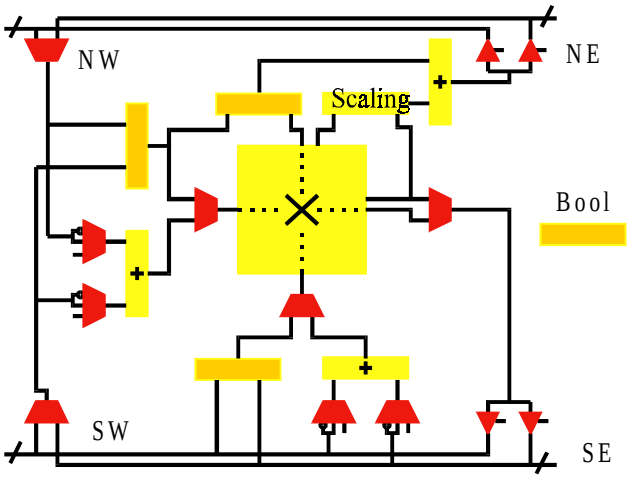

Figure. 2. Schematic diagram of a first generation ALU-datapath, not showing pipeline registers and latches

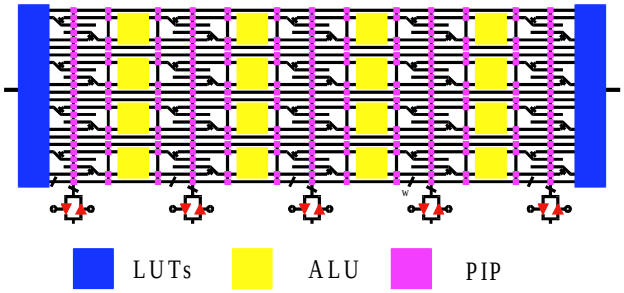

Figure 3. First generation ALU-block with Look-Up Tables, and Programmable interconnect points

2. Principles of operation

Figure 2 shows the schematic diagram of one first generation ALU. This ALU can execute 3 additions / subtractions, 1 multiplication, and 3 bitwise-boolean operations on two operands. The ALUs are grouped in ALU-blocks as shown in Figure 3. The ALUs communicate with each other and/or with look-up tables (LUTs) over their NW, NE, SE and SW I/O ports using a switch matrix with programmable interconnect points (PIPs). Some of the busses are rather short as they realize a nearest neighbor connection, whereas others are used to reach ALUs within a radius of 4 to 8 units. Communication between ALU blocks is supported using bi-directional switches. The LUTs can communicate over a global high speed IO-bus which can be used for internal, i.e. on-chip, communication as well as communication to a burst mode peripheral processor. The programmed datapath contains the execution graph, which is active each clock-cycle. There is no explicit controller, as the ‘control’ is embedded in the dataflow graph, in the same way as this is done in a normal program. It is the ‘controller’ part of the graph which causes other ALUs to read input signals from the LUTs for further processing.

Graph-based execution is a powerful concept, which comes close to general computing on one hand and the implementation of dedicated hardware on the other hand, provided that nested functions can be executed in-line using the dedicated addressing modes, treated in more detail later. Conditionals are executed within an ALU using multiplexers. The control signals of the multiplexers are computed in a small FPGA section, located near the ALUs,

which gets its inputs from the condition-codes computed on the datapath. Registers, which are not shown in the Figure 2, are placed in the datapath to make a pipelined operation of the graph at a high clock-speed possible. Latches used to store programmable constants are not shown either.

3. Low-power operation

The concept of graph-based execution has a power advantage against a DSP or a RISC engine because it does not fetch and decode an instruction each clock-cycle. Placement and wire-length is critical when low-power execution is the goal. A theory for low-power operation was developed [6] to ease the understanding of this issue. A crucial concept is the power radius of an arithmetic building-block. The power radius gives the length of the wires, i.e. inputs and outputs, such that the energy consumed in the wiring equals the energy consumed in the arithmetic building-block, assuming random inputs. The power radius for a full-adder is 0.7mm in a 1¯m CMOS process. This is in the order of 20 times its x- or y-dimension. The power radius of a tightly placed 16x16 multiplier is in the order of 30 to 40 times its x- or y-dimension. There is a tendency that the power radius decreases for 'better' processes. This implies that the wiring on a chip should be kept short whenever possible. Note that the assumption of random inputs assumes that the arithmetic blocks perform an arithmetic operation. The power radius of a well-placed adder is just 5=3 of the power radius of a full adder. There are many applications in which multiple additions are combined with a single multiplication. It is important to design an ALU in which multiple adders are placed adjacent to a single multiplier, in order to take advantage of the fact that the block with the small power radius is contained within the block with the larger power radius. These aspects are the main low-power features which have contributed to the ALU design given in Figure 2, in which a restricted set of I/O ports is used to let the ALU communicate with a network used to interconnect the ALUs. The resulting graph executes an expression of the inner loop of the application program in each clock-cycle. It follows from the concept of the power radius that the busses used to interconnect the ALUs should be kept short, i.e. they should have a length of at most 4 to 8 ALUs to allow for a 10% to 20% energy overhead in the wiring, depending on the process (1μm ... 0.18μm). Figure 3 shows one possible configuration of ALUs and Programmable Interconnect Points (PIPs) in an ALU-block. It is guaranteed that all arithmetic executed within one ALU-block is executed with minimal energy overhead due to the short interconnections used within the block. Another aspect of the theory of low-power operation is related to the balance between the amount of energy needed to execute arithmetic operations and the amount of energy needed to read and or write a small memory (LUT). Using the model introduced in [6] one can calculate that the amount of energy needed to read or write a well designed 16-bits wide LUT with 64 entries is 40% of the energy needed to perform a multiplication using full-adder cells designed for low-power operation.

4. Physical design of the ALU

The ALU shown in Figure 2 was described in VHDL, using the Compass VLSI design tools, to ease the preparation of an accurate description which could be implemented in full-custom design style. The description included the pipeline registers mentioned as well as a micro-PAL, used to implement arbitrary conditionals within one ALU. The size of the ALU pointed out to be 2.74 x 2.20 = 6.02mm^2 in an 0.8¯m, 5V two metal-layer CMOS process, using the high performance library. The Compass tools estimated a power-consumption of 171mW at 10MHz for the whole ALU and 66.29mW for the 16 x 16 Booth-recoded Wallace Tree multiplier-adder which can operate on signed as well as unsigned operands in any combination. Using the high speed library, an energy consumption of 6.6nJ for each multiply-add operation was found and a worst-case propagation delay of 41.11ns at 69.9 ^{o}C, in a cell measuring 1.947mm^2. A full-custom Wallace Tree multiplier with a fast final adder, using a carry select structure with ATLAS full-adders [8] was realized in the same process using the full-custom design tools. This multiplier-adder has an energy consumption of 540pJ for each multiply-add operation and a typical propagation delay of 22.7ns at 25 ^{o}C, in a cell measuring 0.306mm^2. A realization of the multiply-add primitive based on STD-cells was also made for a 0.35¯m, 3V five metal-layer CMOS process. This multiplier-adder, has a typical worst case propagation delay of 13.4ns at 2.7V and 85 ^{o}C, in a cell measuring 0.156mm^2. The execution of conditionals was initially mapped onto a micro-PAL. Two realizations of the micro-PAL were studied. The most straightforward one used registers to store the encoding of the conditionals. A more compact, RAM-based, design was considered as well. Both realizations resulted in a large block, which almost ruined the overall timing of the ALU, due to the fact that the timing analyzer came to the conclusion that the sum of all delays within the arithmetic blocks was the delay of the overall cell. A better solution is sought using the FPGA-blocks mentioned in section 2. These physical design studies resulted in a detailed description of the ALU, along with guidelines for the construction of a high-density, low-power design. Most striking are the differences in energy consumption for the multiply-add primitive, which ranges from 540pJ in full-custom to 6.6nJ using a VHDL based synthesis approach, the cell-area of this building-block ranges from 0.306mm^2 to 1.947mm^2. It may be clear that these differences should not be neglected in the realization of such a new design-platform. The routability of the interconnect busses may be a problem as well, however the presence of five to six metal layers in modern CMOS processes gives opportunities which can be integrated with relative ease in a full-custom design-style. It should be noted that power-consumption and area utilization are the main design issues. For instance a 100mm^2 chip can hold as few as 16 multiplier-add primitives with related ALU-overhead, using the STD cell approach. They will dissipate 16 x 6.629n x 40M = 4.2W in a 5V 0.8μm process. Approximately 108 multiplier-add primitives will fit using the full-custom approach, when the same amount of ALU-overhead is taken into account. The resulting power-consumption at the same clock-rate is: 108 x 540p x 40M = 2.3W. I.e. 6.75 as much arithmetic is performed at a substantially lower energy consumption level when the full-

custom approach is used. It is our objective to make these benefits of the design platform introduced, available to system level architects who will be freed from the burden of optimized physical design, testing, and process migration. It will require a much more detailed analysis before the last word about energy consumption and performance has been said. The problem gets even worse in advanced CMOS processes, like for instance 0.25 or 0.18¯m CMOS. The increased power density comes from the possibility to increase the clock-rate while keeping the area constant. Operation at VDD values of 1V or below, using a tunable Vt, may be needed to get the full advantage of the arithmetic capabilities of the proposed devices at acceptable power levels.

5. A system on a chip

Full-custom VLSI implementations tend to implement a complete system on a chip. The introduction of arithmetic in an FPGA like environment does not necessarily imply that all chip area should be devoted to the graph-based execution of highly optimized arithmetic expressions. Hence it will be needed to find a balance between the arithmetic section and the logic section of the design platform. Consider for instance the radio link used in the personal communicator introduced in section 1. The FFT and IFFT algorithm used in this system are used to prevent problems with multi-path reception. Error correction circuits should be added to recover bits which cancel due to interference. Complex addition and complex multiplication, together with appropriate routing and suitable wireability can be used to construct a butterfly unit from the resources in an ALU-block. Address generation and addressing of LUTs is needed to form an FFT from these butterflies. FPGA based units with built-in adders and on-chip memories are indispensable for the efficient construction of an error correcting circuit, like for instance a Viterbi decoder or a Reed-Solomon decoder. Similar systems may be needed for systems capable of speech recognition and/or pattern recognition. Graphics algorithms seem quite different, but at closer inspection they utilize quite similar resources, even when such diverse algorithms as (sparse) volume rendering and surface rendering are considered. The concept of reconfiguration / reprogrammability demands that the platform should suit all applications. This aspect of systems design is most challenging, as we should not try to solve the design problem of one application but for a family of applications. It would go much too far to discuss any separate systems which we have reviewed during our design effort. Even a single instance of a complex system, like the digital communicator, which should not only be capable to maintain a radio link, but should as well be able to maintain a compressed audio or video stream, is a good example of the complexity of the design problem of a high performance low-power device, which can be reprogrammed on the fly to support a wide range of applications. The design of a digital communicator is rather challenging due to its central role in the man-machine interface. The computational resources needed in this application domain hardly differ from the ones needed for the construction of a high performance digital copier, web-TV's and many other, related products.

6. Contemplating the architecture

Several applications are used to verify the project against the goals set. A complete volume-renderer was used in [1] to define the initial (basic) functionality of the proposed platform. The idea of a graph-based execution greatly helped to simplify the overall concept. The fact that all nodes in the graph execute an operation in one clock-cycle makes it very attractive (almost mandatory) to execute simple functions like: $\sin(x)$, $\cos(x)$, $1/x$, $shading(\alpha,\beta)$, $datavalue(x,y,z)$, etc. in one clock-cycle as well. These example functions show that there is a need for 1D, 2D and 3D addressing modes, used in conjunction with 1D, 2D and 3D interpolation functions, executed by 1, 3, or 7 ALUs to calculate intermediate function results not stored in the 2, 4 or 8 LUTs addressed for these evaluations of a continuous function. It was not needed to resort to linear interpolation only. Higher order interpolations could be mapped efficiently on the ALUs described, using the Bessel interpolation technique. The resulting initial ALU design contains hence at least one multiplier and two adders, which are needed for the efficient implementation of a linear interpolator which calculates: $F = (B - A)h + A$. The VHDL synthesis described in [2] showed that the micro PAL was not suited for this task. The initial design has four I/O ports, one at each corner of the ALU. This is sufficient for an efficient implementation of the linear interpolation function on one ALU. A total of at least 28 instances of this function was needed in the Super Resolution Volume Rendering algorithm [3]. The resulting ALU-structure is shown in Figure 2. The Super Resolution Volume Rendering algorithm can be implemented using 60 out/of 64 ALUs. The second M-Sc. project [2] focused on the implementation of the ALU-design in VHDL. First an initial implementation was made. The results were that the adders, the multiplier and the micro PAL are parts which cause the most delay. Also it showed that the micro PAL was much larger then expected and that it could generate algebraic loops. The adders and the multiplier were optimized resulting in a speed increase of 16%.

A FIR filter was used as a benchmark to be able to compare the speed to some available alternatives. Because of the parallel nature of the FPFA it can calculate one result of a FIR in one clock-cycle. Its high density of multipliers makes it easy to implement large filters. A Texas Instruments TMX 320C6201 running at 200MHz can calculate an 8 tap FIR filter in 4 clock cycles resulting in an output of 50MSPS. The Altera FLEX10K FPGA can achieve 61MSPS for a 16 taps 8 bit FIR filter. The current VHDL implementation of the ALU of the FPFA can run at 25MHz, but expectations based on Altera's projections on scaling from 0.8μm with two metal layers to 0.35μm with four metal layers will increase the speed with a factor of 2.5. This results in 63MHz. The FIR filter uses for an 8 tap filter 12 ALUs. Four extra ALUs are needed because one tap uses 5 I/O ports. When constants are added or the number of I/O ports is increased can the structure calculate an 8 tap filter in 8 ALUs. Current research uses a butterfly of a Fast Fourier Transform (FFT) to optimize the design of the ALU. The original implementation needs 8 ALUs to implement one butterfly. The butterfly uses subtraction followed by a complex multiplication and an addition. If the amount of multipliers and adders is considered it should be

possible to implement this on four ALUs. However the restrictions generated by the number of I/O ports and the structure of the ALU cause that eight are needed. The FIR filter already showed that the number of ports has to be increased to at least five. This change and an increase of the flexibility of the ALU make it possible to implement one butterfly four ALUs. The uses of larger wordlengths could also be very useful. For an efficient implementation of a multi-word multiplication an ALU needs 4 inputs and 2 outputs. The current ALU design can be used to implement a linear interpolation and the other primitives needed for a renderer. Currently it is considered to replace the micro PAL with a small FPGA. The utilization can be increased by increasing the number of inputs and the flexibility of the ALU.

7. Conclusion

The concept of graph based execution has been shown using ALUs and a programmable interconnect. Typical tests, which could be expected to be hard to map successfully, were chosen as benchmarks to contemplate the viability of the platform. A complete Super Resolution Volume Renderer, capable of real-time visualization was taken as the initial application, the digital communicator, including FFT algorithms and error correcting circuits is another full fledged application under consideration. On the algorithmic level, a FIR filter was studied, as well as complex multiplication and addition. In all cases a good utilization of resources was obtained, provided that sufficient busses were used. The initial ALU design, described in detail in this paper was using 4 I/O ports, which could be configured as 2 inputs and 2 outputs, or as 3 inputs and one output. It can be shown that 3 inputs and 2 outputs are needed when one section of an adaptive FIR-filter should be mapped on a single ALU. Multi-precision multiplication requires 4 inputs and 2 outputs, when it should be mapped efficiently on a single ALU. This indicates that advanced algorithmic kernels require substantial I/O to be efficient. This fits in an ideal way in a full-custom design style, in which all metal layers of modern processes can be used for interconnect, rigid power supply rails, de-coupling etc. It has been shown that the speed of one single ALU does not need to be slow, compared to the speed of a single RISC CPU used in modern computers. The aggregate speed of a 200 ALU 0.5μm, 3V, CMOS chip is considerably higher, due to the implicit parallelism. The energy consumption is remarkably low due to the fact that the design style does not have to focus on high speed in the presence of large wiring capacitance. Numerous applications would benefit from the fast time to market, offered by the platform. The low initial cost, due to the reconfiguration concept, will most likely serve new applications as well as replace full-custom chips in traditional applications as soon as a prototype of the platform can be produced.

References

[1] G.B.Remmelink, *Design of an architecture for a field programmable function array*, Master thesis, University of Twente, Enschede, the Netherlands, June 1997. report code: EL-BSC- 038N97.

[2] C.E.Klaassen, *A VHDL based implementation of an ALU of a field programmable function array* Master thesis, University of Twente, Enschede, the Netherlands, March 1998, report code: EL-BSC- 001N98

[3] M.K.Bosma, J.Terwisscha.v.Scheltinga, *Efficient super resolution volume rendering*, Master thesis, University of Twente, Enschede, the Netherlands, June 1995. report code: EL-BSC-098N95

[4] .G.Eldredge, B.L.Hutchings, *Run-Time Reconfiguration: A method for enhancing the functional density of SRAM-Based FPGAs* In: Journal of VLSI signal processing systems for signal, image, and video technology, ISSN 1387-5485, 12(1), 1996, pp. 67-86

[5] C.Ebeling, D.C.Cronquist, P.Franklin, *RaPiD Reconfigurable Pipelined Datapath* In : Lecture notes in computer science, ISSN 302-9743, 1142, 1996, pp. 126-135.

[6] J.Smit, J.H.Huisken, *On the energy complexity of the FFT* In: Proceedings of the Fifth International Workshop PATMOS-'95, Oldenburg, Germany, ISBN 3-8142-0526-X, pp. 119-132

[7] R.W.Hartenstein, *The Microprocessor is no more General Purpose: why Future Reconfigurable Platforms will win* In: Proceedings of the International Conference on Innovative Systems in Silicon, ISIS'97, Austin, Texas, USA, October 8-10, 1997.

[8] D.J.Kinniment, J.D.Garside, B.Gao, *A comparison of Power Consumption in Some CMOS Adder Circuits*. In: Proceedings of the Fifth International Workshop PATMOS-'95, Oldenburg, Germany, ISBN 3-8142-0526-X, pp. 106-118.