Design of hardware accelerators for demanding applications.

Design of Hardware Accelerators for Demanding Applications

Lech Jozwiak, Yahya Jan
Faculty of Electrical Engineering
Eindhoven University of Technology
Eindhoven, The Netherlands
L.Jozwiak@tue.nl

***Abstract*—This paper focuses on mastering the architecture development of hardware accelerators. It presents the results of our analysis of the main issues that have to be addressed when designing accelerators for modern demanding applications, when using as an example the accelerator design for LDPC decoding for the newest demanding communication system standards. Based on the results of our analysis, we formulate the main requirements that have to be satisfied by an adequate accelerator design methodology, and propose a design approach which satisfies these requirements.**

Keywords-hardware accelerator; architecture design; design-space exploration

I. ISSUES AND REQUIREMENTS OF ACCELERATOR DESIGN FOR DEMANDING APPLICATIONS

Hardware acceleration of critical computations has been intensively researched during the last decade, mainly for signal, video and image processing applications, for efficiently implementing transforms, filters and similar complex operations [1][3]-[10]. All these operations have in common that they mainly involve functional parallelism and require relatively simple and regular, limited in space and time local memory accesses between which relatively large portions of computations are performed. In consequence, the main problems of hardware accelerator design for applications of this kind are not related to memory or communication bottlenecks, but to an effective and efficient processing unit synthesis through an adequate parallelism exploitation of the register transfer level (RTL) operations needed for implementation of the required computations. The micro-architecture design for such accelerators can reasonably well be supported by the methods of high-level synthesis [1][3]-[10] and emerging commercial high-level synthesis tools [11].

However, many modern applications (e.g. various decoders in (wireless) communication and multimedia, network access nodes, encryption applications, etc.) require hardware acceleration of algorithms that involve complex interrelationships between the data and computing operations. This can result in complex (global) memory accesses and complex communication between the memories and computing units in the related hardware accelerators. For applications of this kind, the main design problems are related to an adequate resolution of memory and communication bottlenecks and to decreasing the memory and communication hardware complexity, what has to be achieved through an adequate memory and communication structure design. Moreover, for this kind of applications, the memory and communication structure design, and micro-architecture design for computing units cannot be performed independently, because they substantially influence each other. For example, exploitation of more data parallelism in a computing unit micro-architecture usually requires getting the data in parallel for processing, i.e. having simultaneous access to memories in which the data reside (what results in e.g. vector, multi-bank or multi-port memories) and simultaneous transmission of the data (what results e.g. in multiple busses or other interconnects), or pre-fetching the data in parallel to performing other computations. For applications of this kind complex interrelationships exist between the computing unit design and corresponding memory and communication structure design, and complex tradeoffs have to be resolved between the accelerator effectiveness (e.g. computation speed or throughput) and efficiency (e.g. hardware complexity, power and energy consumption etc.).

Furthermore, many modern applications involve algorithms with massive data parallelism at the macro-level or task-level functional parallelism (e.g. LDPC code decoders of the newest communication system standards like IEEE 802.11n, 802.16e, 802.15.3c, 802.3an, 802.15.3c, etc.). To adequately serve these applications, hardware accelerators with parallel multi-processor macro-architectures have to be considered, involving several identical or different concurrently working hardware processors, each operating on a different data sub-set. Each of these processors can also be more or less parallel. Moreover, there is a trade-off between the amount of parallelism and resources at the macro-architecture and micro-architecture level (e.g. similar performance can be achieved with less processors each being more parallel or better targeted to particular part of application, as with more processors each being less parallel or less application-specific). The two architecture levels are strongly interrelated and interwoven, also through their relationships with the memory and interconnection structures. In consequence, optimization of the performance/resources trade-off required by a particular application can only be

achieved through a careful construction of an adequate application-specific macro-/micro-architecture combination.

From the above it should be clear, that the existing high-level synthesis, specifically developed and limited to RTL-level micro-architecture synthesis, is only able to partly support the internal architecture design for particular computation units, and is not sufficient to adequately support the total complex accelerator architecture design process for the modern demanding applications. A new more complex and sophisticated design methodology is needed. In parallel to accounting for the computation unit micro-architecture synthesis, this new accelerator design methodology should to adequately address many more issues, including:

- memory and communication structure synthesis,
- macro-architecture synthesis of the multi-accelerator structures,
- strong interrelationships between the computation unit, memory and communication organization, and between the micro-and macro-architecture, and
- tradeoff exploitation between the micro and macro-architecture, and between the various aspects of the accelerator's effectiveness and efficiency.

Below, we use the accelerator design process for LDPC decoders to illustrate the above discussed issues and requirements of demanding accelerator design, and propose a design approach which satisfies the requirements.

II. ACCELERATOR DESIGN FOR LDPC

A systematic LDPC encoder encodes a message of k information bits into a codeword of length n with the k message bits followed by m parity checks. Each parity check is computed based on a sub-set of message bits. The codeword is transmitted through a communication channel to a decoder. The decoder checks the validity of the received codeword by re-computing the parity checks, using a parity check matrix (PCM) H of size mxn. To be valid, a codeword must satisfy the set of all m parity checks. In Figure 1 an example PCM for a (7,4) LDPC code is given. ``1" in a position $H_{i,j}$ of this matrix means that a particular bit participates in a parity check equation. Each parity check matrix can be represented by its corresponding bipartite Tanner graph [12]. The Tanner graph corresponding to an *(n, k)* LDPC code consists of n variable (bit) nodes (VN) and $m = n - k$ check nodes (CN), connected with each other through edges, as shown in Figure 1. Each row in the parity check matrix represents a parity check equation c_i, $0 \le i \le m-1$, and each column represents a code bit b_j, $0 \le j \le n-1$. An edge exists between a CN i and VN j, if the corresponding value $H_{i,j}$ is non-zero in the PCM.

Usually, iterative Message Passing Algorithms (MPA) [14] are used for decoding the LDPC codes. During decoding specific messages are exchanged among the nodes through the edges. The messages represent the log-likelihood ratios (LLRs) of the codeword bits based on the channel observations [13]. The algorithm starts with the so-called intrinsic LLRs of the received symbols based on the channel observations. Starting with the intrinsic LLR values, the algorithm iteratively updates the extrinsic LLR messages from the check nodes to variable nodes and from the variable nodes to check nodes and sends them among the VNs and CNs along the corresponding Tanner graph edges. If after several iterations the parity check equation is satisfied, the decoding stops, and the decoded codeword is created and considered to be a valid codeword. Otherwise, the algorithm further iterates until a given maximum number of iterations is reached. Since Tanner graphs corresponding to practical LDPC codes of the newest communication system standards involve hundreds variable and check nodes, and even more edges, LDPC decoding represents a massive computation and communication task. Moreover, the modern communication system standards require very high throughput in the range of Gbps and above, for applications like digital TV broadcasting, mmWave WPAN, etc. For the realization of the so high throughput complex highly parallel hardware accelerators are necessary.

In practical MPA algorithms, the node computations are implemented as additions or comparisons of the node inputs. Since each node has several inputs, the basic node operations are the multi-input additions or comparisons. In the corresponding accelerator, the spectrum of possible implementations of each of these multi-input operations spans between the two extremes of a fully serial slow processing in a simple two-input adder/comparator to a fully parallel fast processing in a complex multi-input parallel adder/comparator. When the variable nodes perform their computations the check nodes are waiting on the computation results and vice versa, but all nodes of a given kind, i.e. all the variable nodes or all

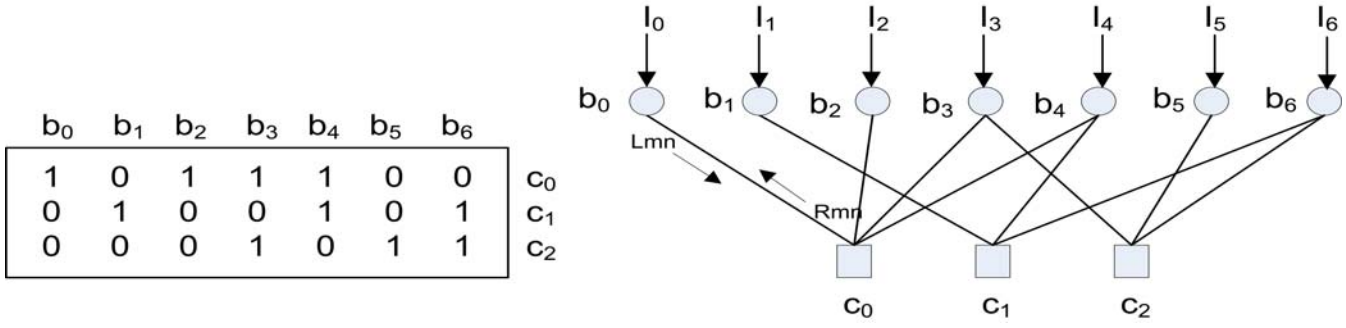

Figure 1. An example PCM for a (7,4) LDPC code and its corresponding Tanner graph
$\{b_0 \ldots b_6\}$ represents variable (bit) nodes, $\{c_0 \ldots c_3\}$ represents check nodes, and $\{I_0 \ldots I_6\}$ represents the input intrinsic channel information

the check nodes, may perform their computations in parallel. If all the nodes of a given kind would actually perform their computations simultaneously, this would require a complex parallel access to the memories of all nodes of the opposite kind, and could only be realized with a very distributed memory structure and very complex and expensive interconnection structure. In contrary, performing the computations corresponding to different nodes fully serially can require just one memory access at a time and result in reasonably simple corresponding memory and interconnection structures. Summing up, in the hardware accelerators for LDPC decoding, the possible micro-architectures span the full spectrum from a fully serial to a fully parallel, and the possible macro-architectures of the multi-accelerator structures span the full spectrum from a fully serial [16] to a fully parallel [17], with large variety of partially parallel architectures [18]-[21] between them. Also, complex tradeoffs are possible between the parallelism and resources at the micro-architecture level, and parallelism and resources at the macro-architecture level. Moreover, changing the parallelism for computations in the micro- or macro-architecture of the LDPC accelerator requires a corresponding change of the memory and communication structure. The data processed in parallel must also be accessible in parallel. Thus, the computation, memory and communication architectures are strictly interrelated and cannot be designed in separation. Moreover, a large number of possible macro-architecture/micro-architecture combinations and related node mappings possible leads to a large number of various tradeoff points in the LDPC accelerator design space representing various accelerator architectures with different characteristics.

III. Accelerator Design Approach

In this section we propose an accelerator design approach which addresses the issues and satisfies requirements of accelerator design for demanding applications as considered in Sections 1 and 2. From the discussion in these two sections one can conclude that a sophisticated design space exploration of accelerator architectures is necessary to arrive at high-quality accelerator designs, in which a set of some most promising architectures has to be efficiently constructed and analyzed, and the best of these architectures have to be selected for further analysis, refinement and actual implementation.

To enable an effective and efficient accelerator architecture exploration, we propose to adapt the quality-driven model-based design paradigm proposed by the first author of this paper [2]. According to this paradigm, *system design* is actually about a *definition of the required quality*, in the sense of a satisfactory answer to the questions: what quality is required and how can it be achieved? To bring the quality-driven design into effect, quality has to be modeled, measured and compared. In our approach, the quality of the accelerator required is modeled in the form of the demanded accelerator behavior, and structural and parametric constraints and objectives to be satisfied by its design, as described in [2]. Our approach exploits the concept of a pre-designed generic architecture platform, which is modeled as abstract generic architecture template. Based on the analysis results of the so modeled required quality, the generic architecture template is adequately instantiated and used to design space exploration that aims at analysis of various architectural choices and macro-/micro-architecture tradeoffs, and finally, at the construction of one or several most promising accelerator architectures supporting the required behavior and satisfying the demanded constraints and objectives. Our approach considers the macro-architecture and micro-architecture synthesis and optimization, as well as, the computing, memory and communication structures' synthesis as one coherent complex accelerator architecture synthesis and optimization task, and not as several separate tasks, as in the state-of-the-art methods. This allows for an adequate resolution of the strong interrelationships between the micro- and macro-architecture, and computation unit, memory and communication organization, as well as, for the effective tradeoff exploitation between the micro- and macro-architecture, and between the various aspects of accelerator's effectiveness and efficiency. According to our knowledge, the so formulated accelerator design problem is not yet explored in any of the previous works related to hardware accelerator design.

The exploration of promising architecture designs is performed as follows. For a given class of applications, a pool of generic architecture templates, including their corresponding processing units, memory units and other architectural resources, is prepared in advance by analyzing various applications of this class, and particularly, analyzing the applications' required behavior, and ranges of their structural and parametric demands. Each generic architecture template specifies several general aspects of the modeled architecture set, such as presence of certain modules types and the possibilities of the modules' structural composition, and leaves other aspects (e.g. the number of modules of each type or their specific structural composition) to be derived through the design space exploration in which a template is adapted for a particular application. In fact, the generic templates represent generic conceptual architecture designs which become actual designs after adequate further template instantiation, refinement and optimization. The adaptation of a generic architecture template to a particular application with its particular set of behavioral and other requirements consists of the design space exploration through performing the most promising instantiations of the most promising generic templates and their resources to implement the required behavior, when satisfying the remaining application requirements. In result, several most promising architectures are designed and selected that match the requirements of application under consideration to a satisfactory degree.

During the design space exploration two major aspects of the accelerator design, being its macro-architecture and micro-architecture, are considered and decided, as well as, the tradeoffs between these two aspects in relation to the design quality metrics (such as throughput, area, energy consumed, cost etc.). It is important to stress that these macro- and micro-architecture decisions are taken in combination, because both the macro- and micro-architecture decisions influence the throughput, area, and other important parameters, but they do

it in different ways and to different degrees. For instance, by a limited area, one can use more elementary accelerators, but with less parallel processing and related hardware in each of them, or vice versa, and this can result in a different throughput and different values of other parameters for each of the alternatives. To decide the most suitable architecture, the promising architectures constructed during the design space exploration are analyzed in relation to the quality metrics of interest and basic controllable system attributes affecting them (e.g. number of accelerator modules of each kind, clock frequency of each module, communication structures between modules, schedule and binding of the required behavior to the modules etc.), and the results of this analysis are compared to the design constraints and optimization objectives. This way the designer receives feedback, composed of a set of instantiated architectures and important characteristics of each the architectures, showing to what degrees the particular design objectives and constraints are satisfied by each of them. If some of the constraints cannot be satisfied for a particular application through instantiation of given templates and their modules, new more effective modules or templates can be designed to satisfy the stringent requirements, or the requirements can be reconsidered and possibly lowered. Subsequently, the next iteration of the design space exploration can be started. If all the constraints and objectives are met to a satisfactory degree, the corresponding final application specific architecture template is instantiated, further analyzed and refined to represent the actual detailed design of the required accelerator.

Based on the accelerator design approach proposed above, we are currently developing a complete design methodology and related tools to apply the approach to the design of hardware accelerators for LDPC code decoders for some of the newest demanding communication system standards and for some other applications.

IV. CONCLUSION

This paper presented the results of our analysis of the main problems that have to be solved in design of accelerators for modern demanding applications, formulated the main requirements that have to be satisfied by an adequate methodology of accelerator design for such applications, and proposed a quality-driven model-based accelerator design approach which satisfies the requirements. Based on this approach, we are currently developing a complete design methodology and related tools, to apply them to the design of hardware accelerators for several modern demanding applications.

REFERENCES

[1] L. Jóźwiak, A. Douglas: Hardware Synthesis for Reconfigurable Pipelined Accelerators, Proc. Of ITNG'2008 – IEEE International Conference on Information Technology: Mew Generations, Las Vegas, NV, USA, April 7-9, 2008, IEEE Computer Society Press, Los Alamitos, CA, USA, pp. 1123-1130.

[2] L. Jóźwiak: Quality-driven Design in the System-on-a-Chip Era: Why and How? Journal of Systems Architecture, Elsevier Science, Amsterdam, The Netherlands, 2001, Vol. 47/3-4, pp. 201-224.

[3] L. Jóźwiak, N. Nedjah and M. Figueroa: Modern Development Methods and Tools for Embedded Reconfigurable Systems – a Survey, Integration - the VLSI Journal, Vol., No, 2009, pp.

[4] R. Schreiber at al: High-level synthesis of nonprogrammable hardware accelerators, Proc. of ASAP'2000, pp. 113–124.

[5] K. Kuchcinski, C. Wolinski: Global approach to assignment and scheduling of complex behaviours based on HCDG and constraint programming, Journal of Systems Architecture, Vol. 49, 2003, pp. 489–503.

[6] S. Gupta, N. Dutt, R. Gupta, A. Nicolau, SPARK: A high-level synthesis framework for applying parallelizing compiler transformations, Proc. Int. Conf. on VLSI Design, 2003, pp. 461–466.

[7] Z. Guo, B. Buyukkurt, W. Najjar, and K. Vissers: Optimized generation of data-path from C codes for FPGAs, Proc. DATE'05, 2005, pp. 112–117.

[8] M. Puschel at al: SPIRAL: Code generation for DSP transforms, Proceedings of the IEEE, Vol. 93, No. 2, 2005, pp. 232–275.

[9] S. Sun, W. Wirthlin, M. J. Neuendorffer: FPGA Pipeline Synthesis Design Exploration Using Module Selection and Resource Sharing, IEEE Trans. on CAD, Vol. 26, No.2, 2007, pp. 254–265.

[10] S,P Mohanty, N. Ranganathan, E. Kougianos, P. Patra, P.: Low-Power High-Level Synthesis for Nanoscale CMOS Circuits, Springer, 2008, pp. 1–298.

[11] Synfora PICO platform for accelerator synthesis from C, http://www.synfora.com/.

[12] R. Tanner: A recursive approach to low complexity codes, IEEE Trans. on Inf. Theory, 27(5), 1981, pp. 533-547.

[13] D.J.C. MacKay: Good error-correcting codes based on very sparse matrices, IEEE Trans. on Inf. Theory, 45(2), 1999, pp. 399-431.

[14] G. Malema and M. Liebelt: Interconnection Network for Structured Low-Density Parity-Check Decoders, Proc. 2005 Asia-Pacific Conf. on Communications, 2005, pp. 537-540.

[15] M.M. Mansour and N.R. Shanbhag: High-throughput LDPC decoders. *IEEE Trans. on VLSI System*, 11(6), 2003, pp. 976-996.

[16] E. Yeo, P. Pakzad, B. Nikolic and V. Anantharam: VLSI Architectures for Iterative Decoders in Magnetic Recording Channels, *IEEE Trans. on Magnetics*, 37, 2001, pp. 748-755.

[17] A. Darabiha, A.C. Carusone and F.R. Kschischang: Multi-Gbit/sec low density parity check decoders with reduced interconnect complexity, Proc. *ISCAS'2005*, 2005, pp. 5194-5197.

[18] K. Gunnam, G. Choi, W. Wang and M. Yeary: Multi-Rate Layered Decoder Architecture for Block LDPC Codes of the IEEE 802.11n Wireless Standard, Proc. *ISCAS'2007*, 2007, pp. 1645-1648.

[19] K. Sangmin, G.E. Sobelman and H. Lee: Flexible LDPC decoder architecture for high-throughput applications, Proc. *APCCAS'2008*, 2008, pp. 45-48.

[20] L. Zhang, L. Gui, Y. Xu and W. Zhang: Configurable Multi-Rate Decoder Architecture for QC-LDPC Codes Based Broadband Broadcasting System, *IEEE Trans. on Broadcasting,* 54(2), 2008, pp. 226-235.

[21] Z. Cui, Z. Wang and Y. Liu: High-Throughput Layered LDPC Decoding Architecture, *IEEE Trans. on VLSI Systems,* 17(4), 2009, pp. 582-587.