ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

Memory Architecture Design for High-End Multiprocessors

Malladi Sunder Rao[1], G. Ramprabhu[2]

Research Scholar, Annamalai University, Annamalai Nagar, Taminadu, India[1]

Professor, Annamalai University, Annamalai Nagar, Tamilnadu, India[2]

ABSTRACT: In the modern environment the System on Chips (SoCs) made a very huge quantity of applications in the field of computer networking, Computer Vision, Audio Signal Processing, Video Signal Processing, Image Processing, Wireless Communication Networks etc. All these kind of applications the data handling can be done by parallelism especially the functional data in the computer vision as well as the Digital Image Processing. These multiple operations can be taken in the form of parallel operation by means of parallel processing with the choice of any type of datas taken for different applications. In the Multiprocessor Architecture used for certain Specific Application follows two types of Approach for Synthesizing. The first type is derived from the Directed Acyclic Graph (DAG) which gives the starting point that is taken from any of the applications given under verification This static scheduling makes more limitations due to the loops inside the scheduled task which is represented by the task graph in the single node for different application, the communication done in local is not properly modeled and this case is used for very worst condition which is scheduled which making the system with over headed design for single node used for multi tasks For these kinds of processes we used the Random Process Network Generator (RPNG) which is used to create the sample network processes which has a characteristic of speedy and aided in speedy validation and it was compared with the Multiprocessor Systems on Chip (MpSoC) techniques used for synthesis. In the Multiprocessor Architecture used for certain Specific Application follows two types of Approach for Synthesizing. The first type is derived from the Directed Acyclic Graph (DAG) which gives the starting point that is taken from any of the applications given under verification. For these kinds of processes we used the Random Process Network Generator (RPNG) which is used to create the sample network processes which has a characteristic of speedy and aided in speedy validation and it was compared with the Multiprocessor Systems on Chip (MpSoC) techniques used for synthesis.

KEYWORDS: Multiprocessor Systems on Chip, Random Process Network Generator, Interconnection Network, Directed Acyclic Graph, System on Chips.

I. INTRODUCTION

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. This research was focused on the monolithic processing unit synthesis with the so-called “high-level synthesis” (HLS) methods [7–14], and not on the massively parallel multi-processor accelerators required for the high end applications. Specifically, this research did not address the memory and communication architecture design of multi-processor accelerators. HLS only accounts for a simple memory in the form of registers and simple flat interconnect structure between the data path functional units and registers.

Although some research results related to the memory and communication architectures can be found in the literature [15] in the context of programmable on-chip multiprocessor systems, the memory and communication architectures were proposed there for the much larger and much slower programmable processors. They are not adequate for the small and ultra-fast hardware processors of the massively parallel multi-processor accelerators, due to a much too low bandwidth and scalability issues. The approaches proposed in the context of the programmable on-chip multiprocessors utilize time-shared communication resources, such as shared buses or network on chip (NoC). Such communication resources are however not adequate to deliver the data transfer bandwidth required for the massively parallel multi-

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

processor accelerators. In case of the massively parallel multiprocessor accelerators, the application-specific processors and the corresponding memory and communication architectures must be compatible (match each other) in respect to bandwidth (parallelism). Therefore, the communication architecture cannot be realized using the traditional NoC or bus communication to connect the processing and storage resources but requires point-to-point (P2P) communication architectures compatible with the parallel processing and memory resources. The traditional NoC-based communication architectures utilize a network of switches, as for instance, each switch connected to one resource (processor, memory, etc.) and four interconnected neighboring switches forming a mesh. This way a large number of resources can be connected without using long global wires and thus reducing the wire delays (scalability). However, the timeshared links introduce extra communication cycles, which negatively impact the communication and overall performance.

Figure 1. LDPC encoding and decoding process.

When implemented as embedded SRAM is almost *1.6 times* larger than when implemented as FF-based (implemented in TSMC 90nm LPHP Standard Cell Library) memory, and the area proportion grows fast with further decrease in memory sizes. Therefore, for the case of IEEE 802.15.3c LDPC decoders, the SRAM-based memories are only efficient (and considered in our DSE and experimental designs) for a combined processing parallelism of up to 84 only. Additionally, the memory and communication issues are not orthogonal in nature, resolving and optimizing one issue in separation heavily influences the other. Thus, the memory and communication architecture synthesis has to be realized as one coherent synthesis process accounting for the mutual influences and tradeoffs.

II. LITERATURE SURVEY

Summing up, the massive data, operation-level and task-level parallelism to be exploited to achieve the ultrahigh throughput required by the highly demanding applications, the complex interrelationships between the data and computing operations, and the combined parallelism exploitation at the two architecture levels (micro-/macro architecture) make the design of an effective and efficient communication and memory architecture a very challenging task. To effectively perform this task, the (heterogeneous) parallelism available in a given application has to be explored and exploited in an adequate manner in order to satisfactorily fulfill the design requirements through constructing an architecture that satisfies the required performance, area, and power tradeoffs.
The adaptation of a generic architecture template to a particular application with its particular set of behavioral and other requirements consists of the DSE 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 the application under consideration to a satisfactory degree.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

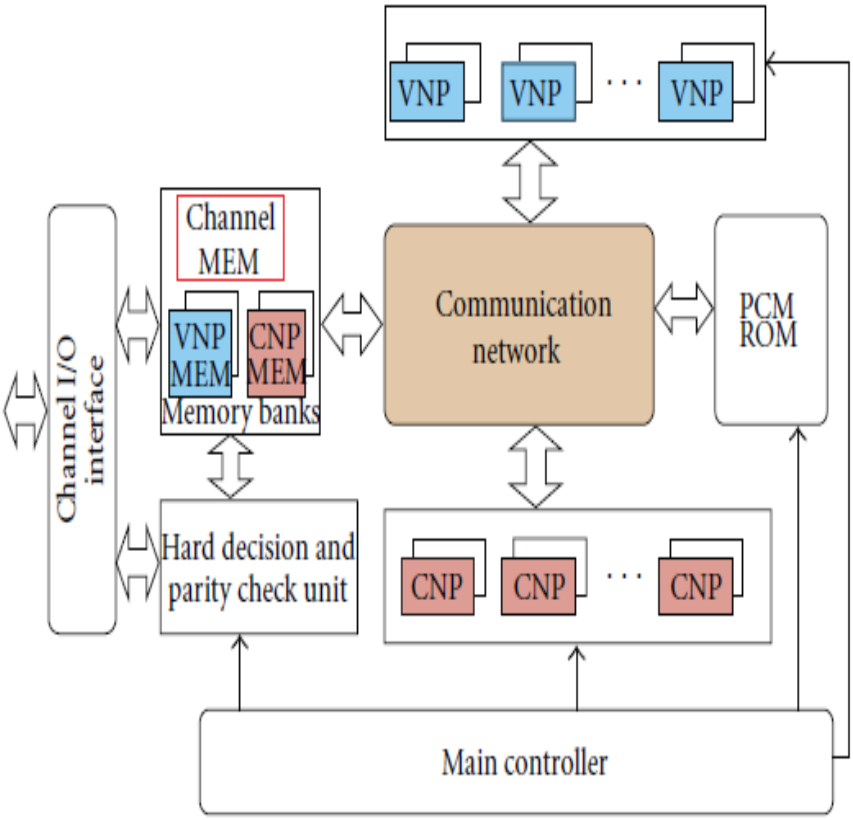

Figure 2. Example of a generic architecture template for LDPC decoding accelerators.

III. PROPOSED METHOD

In a large majority of practical cases, the throughput and clock speed are the hard constraints that must be satisfied, while the area, power, and their mutual tradeoffs are considered as the design objectives that have to be optimized. In these cases, the effectiveness of an accelerator is represented by the throughput and frequency constraints, while the area, power, and their mutual tradeoffs reflect its efficiency. This way the required accelerator quality is modeled, and this quality model is used to drive the overall architecture exploration and synthesis process that carefully stepwise constructs the most promising architectures. It is performed in the following three stages, each corresponding to one of the main design issues (sub problems) that have to be solved to result in complete accelerator architecture:

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

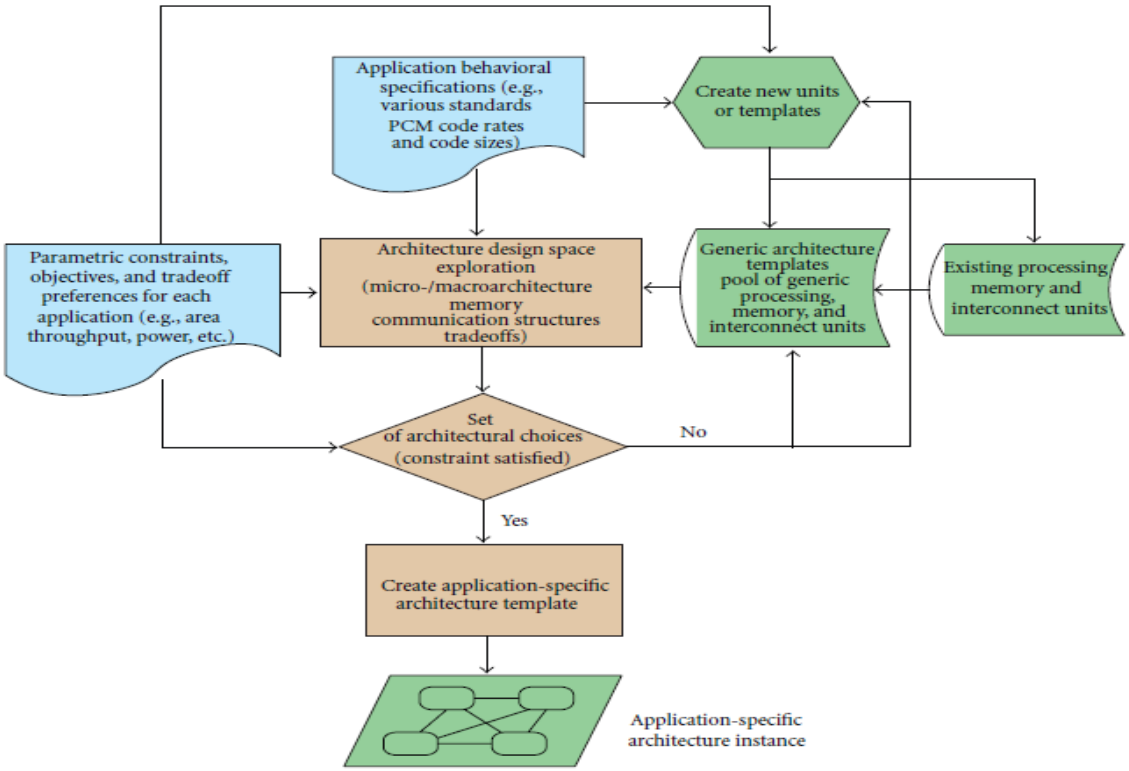

Figure 3. Architecture exploration framework of the accelerator design methodology.

(1) Decision of the processor's micro- and macro architectures (processing parallelism) for each different data parallel task,
(2) Decision of the memory and communication architecture for selected micro-/macro architecture combinations,
(3) Selection and actual composition of the final complete accelerator architecture.
Since the throughput and clock speed are the hard constraints and their satisfaction mainly depends on the processing parallelism, and in turn, the required processing parallelism decides to a high degree the memory and communication architecture, the architecture exploration starts with the decision of the processing parallelism (stage 1). In this stage, two major aspects of the accelerator design, being its micro architecture and macro 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. The data distribution and data mapping approach are described below using an example of two heterogeneous data-parallel tasks sharing multiple memories.
Let us assume a set of m data-parallel tasks $Ti = \{T1, \ldots, Tm\}$ and another set of n data-parallel tasks $Tj = \{T1, \ldots, Tn\}$. Let $|Pi(P\text{mic}, P\text{mac})|$ and $|Pj(P\text{mic}, P\text{mac})|$ be the number of processing tiles allocated to the tasks Ti and Tj, respectively, where Pmic represents the micro architecture parallelism, and Pmac represents the macro architecture parallelism of each processing tile. Let $|Mi, j| = Pi(P\text{mic}) \times Pi(P\text{mac})$ and $|Mj,i| = Pj(P\text{mic}) \times Pj(P\text{mac})$ be the number of memory tiles shared among the processing tiles $|Pi|$ and $|Pj|$. Further, we assume that $|Pi|$ reads data from $|Mi, j|$ and writes to $|Mj,i|$ and vice versa for $|Pj|$. For data distribution, we propose an interleaved (cyclic) data distribution scheme. This approach regularly and uniformly distributes data in memories, which enables us to use our communication strategies. Further, this approach has the additional benefit that it minimizes the complexity of the addressing logic. We perform data distribution based on interleaving in two stages, to resolve the read and write access conflicts, respectively. This data distribution (distribution-L1) will resolve all the memory read conflicts for processors

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

|*Pi*| that will be in the case if no memory partitioning is done and all data is stored in a single memory (single port), as shown in Figure 8. On the other hand, when the processor tiles |*Pj* | write to the share memory tiles |*Mi*, *j* |, it might result in write conflicts because of the order imposed by the |*Pi*| processor tiles for conflict-free read on the data tiles, as given in (1). Therefore, we use another level of data interleaving so that the processor tiles |*Pj* | write their data without any conflict, while ensuring that |*Pi*| read accesses will not be effected. Unlike the interleaving which is at the level of a data tile, we perform this rather at the block level. All the data tiles distributed in the partitioned memories |*Mi*, *j* | for the task |*Ti*| are first divided into sets of equal size blocks (each block consists of a set of data tiles), then the data tiles of each block are skewed (interleaved) by a certain value. The data blocks are formed by taking into account the information about the set of tasks *Tj* and their relevant data tiles *Si*, *j* that are scheduled simultaneously on the processor tiles *Pj* . The block is formed in such a way that each block contains a single data tile *Si*, *j* from the scheduled subsets of tasks *Tj*, and to avoid the conflicts, data tiles are then interleaved (skewed) in each block by some value. This way the processor tiles |*Pj* | can write to the shared memory tiles |*Mi*, *j* | without any conflict, when ensuring that the corresponding read will not be effected. We can determine the block-level data distribution using the equation below:

$Bn(x) = n$, where $0 \leq n \leq |Bn|$, (2)

where *B*(*x*) represents the block index number, *n* represents the value of the interleaving (skew factor), and |*Bn*| represents the total number of blocks. The same conflict-free read/write access order is valid for |*Mj*,*i*| shared memory tiles except that the read/write access order is just reversed. It is equally possible that during data distribution for resolving the read conflicts, it might also resolve the write conflicts.

In such scenario, the second level data distribution would not be needed. Further, the shared partitioned memories can be implemented using flip-flop- (FF-) based registers or embedded SRAM memories. We integrated into our DSE framework, the HP CACTI, a cache, and SRAM compiler, for memory characterization with different configurations required during DSE. We will further explain the above discussed communication and memory design approach and its strategies using as a representative test case the design of LDPC decoders for the future communication system standards.

The performance gain for all the accelerator instances as high as *5 times* on the cost of a very small area penalty of less than *1*% is compared to the non partitioned two-level hierarchical communication network. Nevertheless, the communication network area still dominates the area of processing elements and memories.

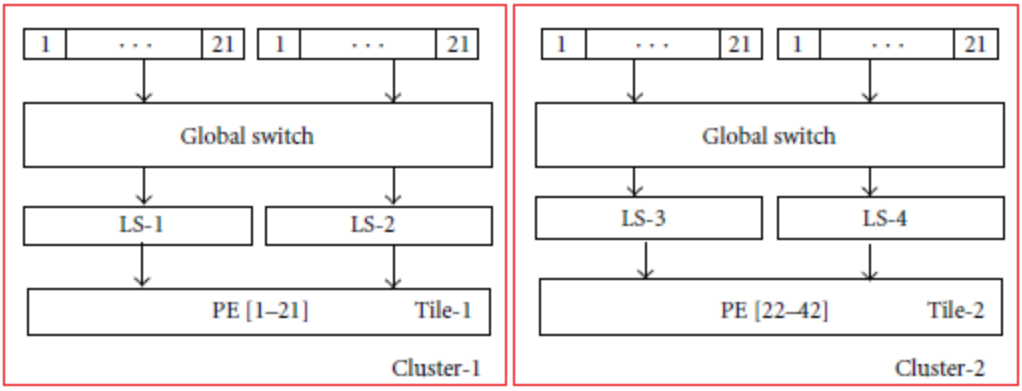

Figure 4. Data-distribution-based communication partitioning.

Therefore, we introduce one more technique to further reduce the communication network area, while preserving the performance.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

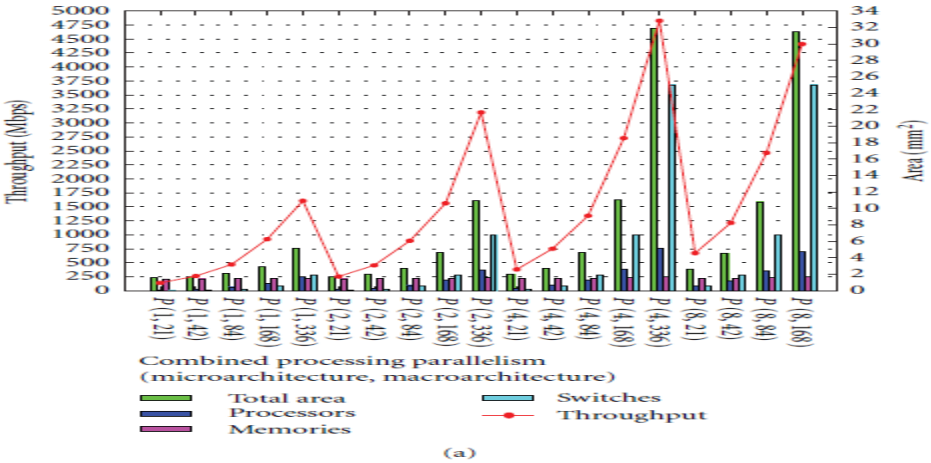

Figure 5. Area/performance tradeoffs for the data-identification-based communication partitioning shown at (a), and the same partitioned network when realized using multistage switches shown at (b).

IV. EXPERIMENTAL RESULTS DISCUSSION AND COMPARISON

We conducted a series of experiments for IEEE 802.15.3c LDPC codes with different micro-/macro architecture parallelism combinations to explore the numerous complex design tradeoffs, specifically, in relation to the memory and communication architecture. We performed our experiments for the 1/2 rate IEEE 802.15.3c LDPC code with code length 672, assuming a high data precision of 8 bits for communication and 10 iterations per frame. Most of the more specific results of our experiments are discussed in the previous sections to illustrate the memory and communication design approaches. However, the overall view and general conclusions are not less important. The multi-stage/single-stage switch combination approach results in a low performance penalty of less than 1% at most, while offering a significant area savings as high as *4 times*. An interesting observation is that independent of the two-level hierarchical communication network partitioned or not, the application of different combinations of single stage and multi-stage realization to local and global switches always results in a superior scalability in one design dimension or in the other design dimension.

Our experiments on different switch combinations show that the highest performance is achieved for the single-stage global switch and the multi-stage local switches, while for the lowest area, the multi-stage realizations are required for both the global and local switches. The same sort of behavior can be observed for the two-level non partitioned communication network, when the single and multi-stage switch combinations are applied. For high-performance applications, a state-of-the-art partially parallel LDPC architecture is proposed by Liu and Shi [36]. The architecture exploits partially parallel micro architecture for CNP and fully parallel micro architecture for VNP with the macro architecture parallelism of 84 for each kind of processors (CNP and VNP). To make a fair comparison of this architecture to architectures constructed by our method, we assumed a standard set of parameters (IEEE 802.15.3c code, code length (672 bits), code rate (7/8), number of iterations (10), and data width (8 bits)). We then implemented the architecture proposed in [36] and our architectures in the same technology (TSMC 90 nm LPHP standard Cell Library) using our DSE and synthesis tool called multiprocessor accelerator explorer (MPA-explorer). The throughput achieved by the architecture proposed in [36] is limited to 1Gbps due to the long delays of the flat multiplexer- and de multiplexer-based interconnects scheme exploited in [36]. Our architecture with the same processing parallelism implemented using our MPA-explorer achieves *1.94 times* higher throughput, *1.43 times* lower area, and almost the same power consumption compared to the architecture proposed in [6].

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

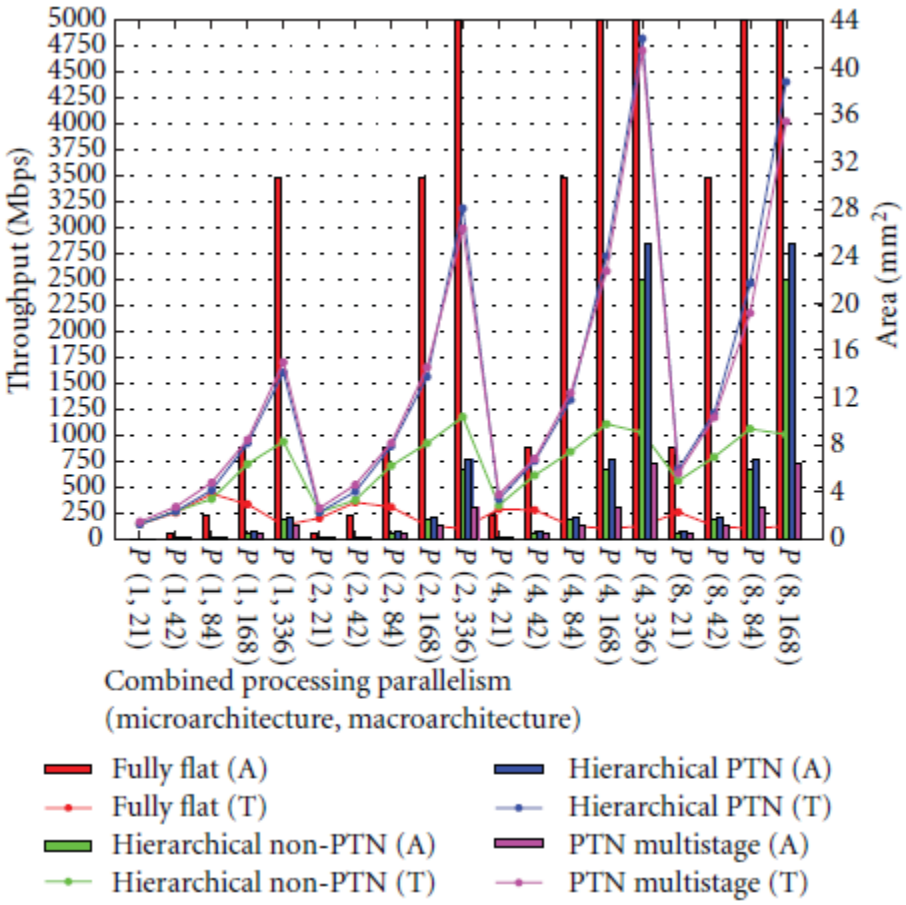

Figure 6. Combined results representing interconnect parameters for various processing parallelism.

This is the main reason for almost the twice higher throughput by almost 50% lower area of the LDPC decoder architecture generated by our MPA-explorer. summing up, the above-discussed memory and communication architecture design approaches, when applied in combination, ensure scalability of the memory and communication architecture for the massively parallel multi-processor hardware accelerators and exploration of the complex design tradeoffs. This reconfirms once more the fact that without having a quality-driven model-based design approach, such complex design tradeoffs for massively parallel multiprocessors cannot be adequately explored in a reasonable time.

V. CONCLUSION

In the former sections, we discussed the communication and memory architecture design issues of the massively parallel multi-processor accelerators necessary to realize the required ultrahigh throughput of the highly demanding modern applications. Our discussion was focused on the communication and memory bandwidth and scalability issues. We demonstrated that in the massively parallel hardware multiprocessors, the memory and communication influence on both the throughput and circuit area dominates the processors influence, and the communication and memory design are strictly interrelated. Therefore, communication and memory architecture design is one of the major aspects of our new accelerator design methodology. We VLSI Design 19 demonstrated that the traditionally used simple flat communication architectures and multi-port memories do not scale well with increase of the accelerator parallelism. In consequence, they are not adequate for the massively parallel accelerators. We proposed to design the application-

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 7, July 2016

specific hierarchical partitioned organizations of the communication architectures and vectorized memories exploiting the regularity and hierarchy of the actual information flows of a given application. This drastically improves the scalability. In particular, we demonstrated that for the moderate parallelism levels, the two-level architectures with several small global communication-free clusters or a single global cluster perform well, with performance gains as high as *12 times* and area savings as high as *25 times* compared to the flat communication scheme. However, for the high parallelism levels, only the partitioned hierarchical approach ensures the scalability regarding performance with gains as high as *5 times* and a small area penalty of less than 1% compared to the non partitioned two-level hierarchical communication network. To further increase the performance or eliminate the area penalty, the multi-stage and single stage switch combinations can be employed for local and global switches of the two-level partitioned communication network, resulting in area saving as high as *4 times*. Regarding the memory design, the partitioned vectorized shared (single port)memories seem to be the most promising for the massively parallel hardware multiprocessors. To guarantee the required memory and communication bandwidth and achieve the communication and memory scalability in the whole considered performance range, we incorporated all the discussed in this paper strategies of communication architecture synthesis, as well as of memory partitioning, data distribution, and related data mapping into our multi-processor accelerator design method and related automated architecture exploration framework. Using this framework we performed a large set of experiments including all the experiments referred to in this paper. The experiments demonstrated that our quality-driven model-based accelerator design method, and specifically its memory and communication synthesis techniques discussed in this paper, is adequate for the hardware multi-processor design for modern highly demanding applications.

REFERENCES

[1] W. Wolf, A. A. Jerraya, and G. Martin, "Multiprocessor system-on-chip (MPSoC) technology," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 10, pp. 1701–1713, 2008.
[2] W. Wolf, "The future of multiprocessor systems-on-chips," in*Proceedings of the 41st Design Automation Conference*, pp. 681– 685, ACM, New York, NY, USA, June 2004.
[3] G. Martin, "Overview of the MPSoC design challenge," in *Proceedings of the 43rd ACM/IEEE Design Automation Conference*, pp. 274–279, 2006.
[4] R. Joost and R. Salomon, "Advantages of FPGA-based mul- tiprocessor systems in industrial applications," in *Proceedings of the 31st Annual Conference of the IEEE Industrial Electronics Society (IECON '05)*, pp. 4451–4506, November 2005.
[5] C. Wright and M. Arens, "Fpga-based system-on-module approach cuts time to market, avoids obsolescence," *FPGA and Programmable Logic Journal*, vol. 6, no. 6, 2005.
[6] S. Raje, "Catching the FPGA productivity wave," *Electronic Design*, vol. 52, no. 24, p. 20, 2004.
[7] K. Ravindran, N. Satish, Y. Jin, and K. Keutzer, "An FPGA- based soft multiprocessor system for IPV4 packet forwarding," in *Proceedings of the International Conference on Field Programmable Logic and Applications (FPL '05)*, pp. 487–492, August 2005.
[8] Y. Jin, N. Satish, K. Ravindran, and K. Keutzer, "An automated exploration framework for FPGA-based soft multiprocessor systems," in *Proceedings of the 3rd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and Systems Synthesis (CODES+ISSS '05)*, pp. 273–278, ACM, Jersey City, NJ, USA, September 2005.
[9] P. Huerta, J. Castillo, J. I. Ma´ rtinez, and V. Lo´ pez, "A microblaze based multiprocessor SoC," *WSEAS Transactions on Circuits and Systems*, vol. 4, no. 5, pp. 423–430, 2005.
[10] J. Dykes, P. Chan, G. Chapman, and L. Shannon, "A multiprocessor system-on-chip implementation of a laser- based transparency meter on an FPGA," in *Proceedings of the International Conference on Field Programmable Technology (ICFPT '07)*, pp. 373–376, December 2007.
[11] O. Lehtoranta, E. Salminen, A. Kulmala, M. Ha¨nnika¨inen, and T. D. Ha¨ma¨la¨inen, "A parallel MPEG-4 encoder for FPGA based multiprocessor SOC," in *Proceedings of the International Conference on Field Programmable Logic and Applications (FPL '05)*, pp. 380–385, August 2005.
[12] R. K. Karanam, A. Ravindran, and A. Mukherjee, "A stream chipmultiprocessor for bioinformatics," *SIGARCH Computer Architecture News*, vol. 36, no. 2, pp. 2–9, 2008.
[13] A. Tumeo, M. Branca, L. Camerini et al., "A dual-priority real-time multiprocessor system on FPGA for automotive applications," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '08)*, pp. 1039–1044, ACM, New York, NY, USA, March 2008.
[14] J. Khan, S. Niar, A. Menhaj, Y. Elhillali, and J. L. Dekeyser, "An MPSoC architecture for the multiple target tracking application in driver assistant system," in *Proceedings of the 19th IEEE International Conference on Application-Specific Systems, Architectures and Processors (ASAP '08)*, pp. 126–131, July 2008.
[15] S. Ben Othman, A. K. Ben Salem, and S. Ben Saoud, "MPSoC design of RT control applications based on FPGA SoftCore processors," in *Proceedings of the 15th IEEE International Conference on Electronics, Circuits and Systems (ICECS '08)*, pp. 404–409, September 2008.