# Resource Efficiency of Scalable Processor Architectures for SDR-based Applications

Thorsten Jungeblut[1], Johannes Ax[2], Gregor Sievers[2],
Boris Hübener[2], Mario Porrmann[2], and Ulrich Rückert[1]

[1]Cognitive Interaction Technology
Center of Excellence
Bielefeld University, Germany

[2]Heinz Nixdorf Institute
University of Paderborn, Germany

## Abstract

*In this work we discuss the resource efficiency of scalable processor architectures for software-defined radio (SDR) based applications. The development of resource efficient processor architectures is based on a two-staged design flow using a high level processor specification as a reference. This design-flow has been used to perform a comprehensive design-space exploration of algorithms from various application scenarios. As a result the 4-issue VLIW-architecture* CoreVA *has been implemented. The fine-grained parallelism of this architecture allows for performance gains of up to a factor of three to four for the analyzed application scenarios. From the design-space exploration on system level dedicated hardware accelerators have been derived. The hardware accelerators proposed in this work improve the energy efficiency by a factor of 6 to 43. To further improve the performance of the system, the single-core based approach is extended to network-on-chip (NoC) level. As an example, an IEEE 802.11b application has been mapped to a cluster of four processor cores reducing the processing time of the algorithm by up to 60 %.*

## 1 Introduction

Wireless communication finds it's way in our daily life. Complex transmission methods (WLAN[1], UMTS[2], LTE[3] provide increasing data rates at low latencies. This allows for new applications, like HDTV[4], video conferences, or 3D online gaming. Basically, a mobile phone has to support multiple standards. Up to now, heterogenous hardware platforms with dedicated accelerator devices were used. Whereas in practice, none or only few algorithms are used simultaneously, all accelerators contribute to the area (and therefore the costs) and the power consumption of the mobile device.

Therefore, modern approaches rely on flexible architectures, which are based on high performance and universal processors [10, 7, 17, 18, 16]. This principle is called *software-defined radio* and allows for the exchange of the communication methods during run-time. In addition, new algorithms can be ported to the architecture. High-level programming languages simplifies the development of applications and are mainly platform independent.

The decreasing feature size of modern fabrication processes allows for the integration of more and more logic or memory in embedded systems. Parallel architectures offer a high performance at a reasonable low clock frequency. Compared to superscalar architectures, *very long instruction word (VLIW)* processors allow for a fine-grained scaling of the computational throughput at low resource requirements. Examples for current commercial VLIW architectures are *NXP TriMedia* [25] and the *Texas Instruments TMS320C6X* [24]. Scalable Network-on-Chips (NoCs) with multiple processor cores can further enhance the available throughput dependent on the application-requirements. Current architectures are, e.g., *Tilera Tile64* [1, 2], *Intel TeraFLOPS* [26], *XPipes* [3], *AEtheral* [23], or *FAUST* [15].

In this work we discuss the potential of scalable processor architectures and NoCs for the execution of SDR-based applications. In Section 2 we propose a two-stages design flow for the design-space exploration of resource efficient processor architectures. Section 3 introduces our modular VLIW-architecture *CoreVA*. Section 4 presents the system architecture and proposes a flexible environment for the in-

[1]Wireless Local Area Network
[2]Universal Mobile Telecommunications System
[3]Long Term Evolution
[4]High Definition Television

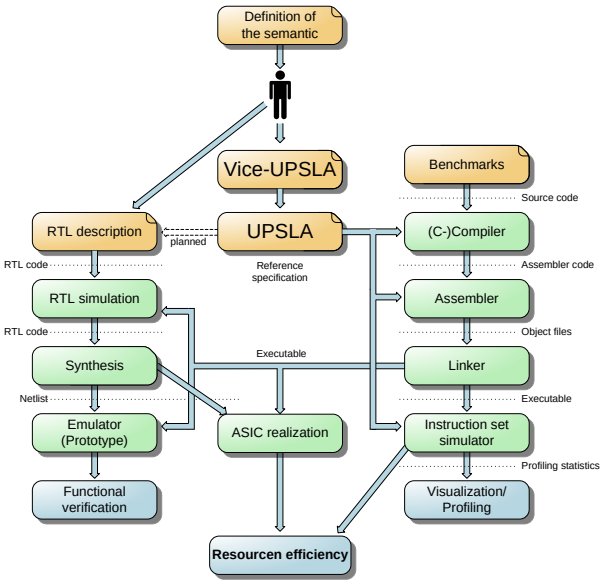


**Figure 1. Two-stage design-flow for the design space exploration of processor architectures**

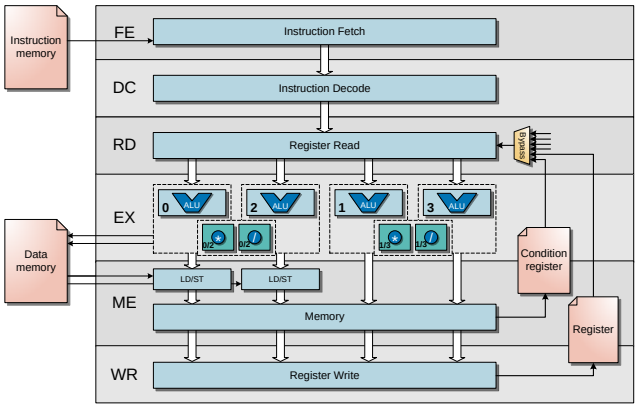


**Figure 2. 6-staged pipeline of the CoreVA processor**

tegration of hardware accelerators. In Section 5 the extension of the CoreVA system to NoC level is introduced. Section 6 concludes this paper and gives an outlook on future work.

## 2 Processor Design Based on UPSLA as a Reference Specification

For the development of resource efficient processor architectures, a two-stage design-flow is used, which consists of the automatic generation of a complete C-compiler-based toolchain from a reference specification in the *Unified Processor Specification Language (UPSLA) (cf. Figure 1, [14, 11]*).

The development toolchain encompasses a C-compiler, an assembler, a linker, a cycle-accurate instruction set simulator (ISS) and various debugging and profiling tools. Starting from the application, the source code is compiled and can be profiled, e.g., in terms of clock cycles or functional units utilization. The hardware design flow is based on the register-transfer level (RTL) description of the architecture. The executables of the application can be simulated using RTL-simulation or verified functional using our modular rapid prototyping environment RAPTOR [21]. Combining the synthesis results (wrt. the resource requirements) and the profiling results, the *resource efficiency* is derived. To verify the consistency of the software and the hardware domains, the simulation by validation approach of [5] is used.

## 3 The modular CoreVA VLIW-architecture

Using this design flow we implemented the resource efficient VLIW-architecture called *CoreVA*. The RTL description of the architecture is widely configurable: The number of VLIW-slots (even single slot), arithmetic-logical units (ALUs), dedicated multiply-accumulate (MLA) or division step (DVS) units can be specified. Load/store interfaces can be assigned the each VLIW-slot. Using a Greedy-based approach, optimized pipeline-bypass configurations [4] can be derived by the systematic deactivation of rarely used bypass paths.

From design-space exploration of a heterogeneous set of algorithms of various application scenarios (synthetic benchmarks, coding, baseband processing, error correction, cryptography, and image processing) the configuration of the first implementation of the CoreVA architecture has been derived [12, 6, 10, 7]. The CoreVA architecture embeds four VLIW architectures with four ALUs, two dedicated MLA- and DVS-units (cf. Figure 2). The level-1 caches for instructions and data (two-port) with a size of 16 kByte each interface to external memory [13]. For applications with low memory requirements the caches can be configured to a scratchpad mode at run-time to omit high latencies on cache misses. The fine-grained parallelism of the VLIW architecture allows for performance gains of up to factor three to four for the analyzed application scenarios. Figure 3 combines the layout with a chip photo of the CoreVA architecture. Maximum frequency of the ASIC prototype is 400 MHz (1.2 V, 25 °C) at a power consumption of 100 mW. Area requirements are 2.7 $mm^2$ in am 65 nm standard cell technology from STMicroelectronics.

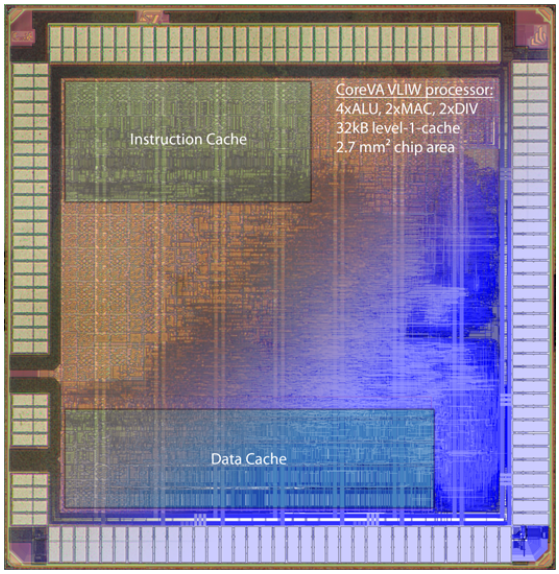


**Figure 3. Chip foto (upper left) and layout (lower right) of the CoreVA architecture**

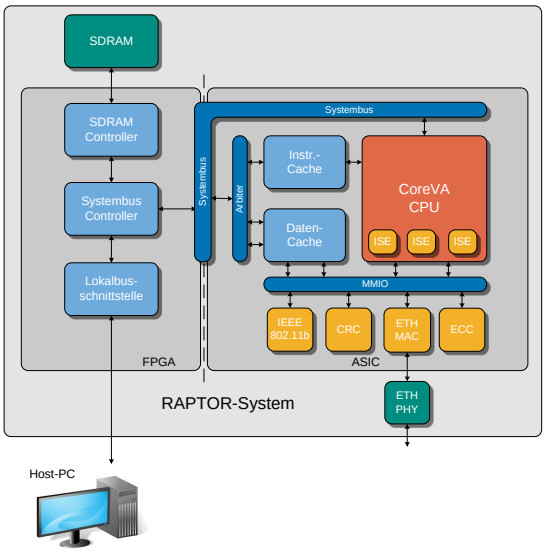


**Figure 4. System architecture of the CoreVA processor**

## 4 Hardware accelerators

Figure 4 depicts the system architecture of the CoreVA processor. Beside the instruction and data caches, the processor core can access dedicated hardware accelerators via *memory-mapped I/O (MMIO)*. Hardware extensions are mapped to the logical memory address space. Dependent on the memory address, an address decoder forwards accesses either to the physical memory or to an extension. Application-specific hardware accelerators allow for high performance gains but require with a considerable additional hardware effort. Nevertheless, due to a large decrease of the processing time, overall energy efficiency can be increased by orders of magnitudes. Table 1 shows the results for the resource efficiency for four hardware accelerators (*cyclic redundancy check (CRC)*, *elliptic curves cryptography (ECC)* [12], *IEEE 802.11b* [7], and *advanced encryption standard (AES)*). To limit area requirements and costs it may not be possible to integrate all dedicated hardware extensions available on the processor DIE. Therefore, we implemented a generic interface to either integrate hardware extensions tightly coupled in the processor core, or map them loosely coupled to a dedicated FPGA (cf. Figure 4). The reconfigurability of the FPGA supports the dynamic exchange of hardware extensions during run-time.

## 5 The CoreVANoC

To further enhance the performance of the system, the single-core based approach can be extended to a multi-core architecture [8, 9]. Whereas the fine-grained parallelism of VLIW architectures can improve the throughput for applications with a high *instruction level parallelism (ILP)*, NoCs can enhance the efficiency of the system by applying software-pipelining or exploiting data concurrency. The *CoreVANoC* is based on the GigaNoC [22, 20, 19] and represents a hierarchical NoC that is especially suitable for scalable multiprocessor architectures. The CoreVANoC features packet-switched wormhole routing with a link bandwidth of up to 24 GBit/s. Backbone of the NoC is a parameterizable *Switch-Box (SB)*. Each SB interfaces to a dedicated CoreVA processor. Area requirements are about 0.5 $mm^2$ per SB at a maximum clock frequency of 750 MHz.

As an example, the IEEE 802.11b application of [7] has been mapped to the CoreVANoC (cf. Figure 5. The algorithm is partitioned to four processor nodes. The first node performs *scrambling*, *differential encoding*, and symbol mapping. The FIR-filter is split into inphase (I) and quadrature (Q) components, each mapped to a dedicated processor node. A fourth CoreVA is required for the synchronization and post-processing of the I/Q-data. By using the CoreVANoC, the IEEE 802.11b algorithm could be sped up by about 60 % (cf. Figure 6).

## 6 Conclusion

In this paper we discussed the resource efficiency of scalable processor architectures and NoCs for SDR-based applications. We introduced a two-staged design-flow used

| Hardware accelerator | Processing time (speedup) | Area requirements | Power consumption | Energy efficiency |
|---|---|---|---|---|
| CRC | -87 % (×8) | +0.7 % | +0.6 % | × **6.8** |
| ECC | -93 % (×14) | +30 % | +30 % | × **11.0** |
| 802.11b | -88 % (×8) | +40 % | +19 % | × **6.0** |
| AES | -99 % (×66) | +39 % | +54 % | × **43.0** |

**Table 1. Resource efficiency of hardware accelerators**

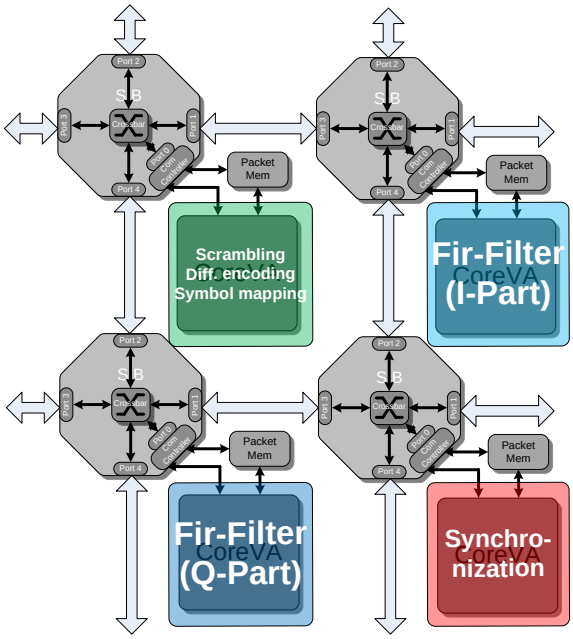


**Figure 5. Mapping of the IEEE 802.11b algorithm to the CoreVANoC**

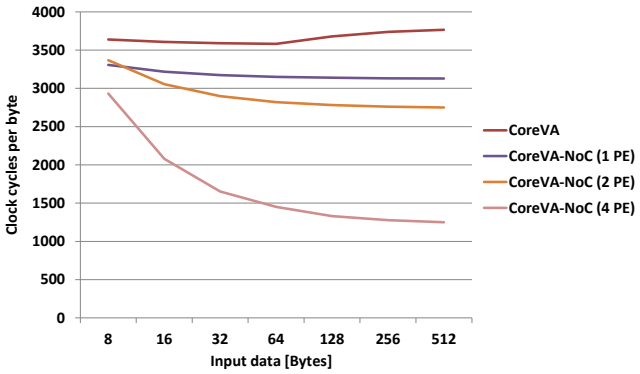


**Figure 6. Execution time of the IEEE 802.11b algorithm for different mapping strategies**

for the design-space exploration of resource efficient processor architectures. The design flow is based on a central processor specification in the UPSLA language. The design flow is highly automated to shorten the iteration cycles of the design-space exploration. As a result from a comprehensive profiling of a large set of applications from SDR-scenarios, the CoreVA VLIW architecture has been derived. The fine-grained parallelism of the architecture allows for performance improvements of the selected applications of a factor of three to four.

By extending the design-space exploration to system level, several hardware accelerators have been implemented. This application-specific extensions improve the energy efficiency by a factor of 6 to 43.

To further improve the performance, the single-core approach can be extended to NoC level. The CoreVANoC is based on the GigaNoC and integrates the CoreVA VLIW architecture as a processor node. As an example an IEEE 802.11b algorithm has been mapped to a cluster of four CoreVA processors. Different mapping strategies have been compared, leading to a reduction of the processing time of up to 60 %.

## References


[1] A. Agarwal. The Tile Processor: A 64-core Multicore for Embedded Processing. In *Proc. of HPEC Workshop*, 2007.

[2] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, L. Bao, J. Brown, et al. Tile64-processor: A 64-core SoC with Mesh Interconnect. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pages 88–598. IEEE, 2008.

[3] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. D. Micheli. NoC Synthesis Flow for Customized Domain Specific Multiprocessor Systems-on-Chip. *IEEE*, 16, 2005.

[4] R. Dreesen, T. Jungeblut, M. Thies, and U. Kastens. Dependence Analysis of VLIW Code for Non-Interlocked Pipelines. In *Proceedings of the 8th Workshop on Optimizations for DSP and Embedded Systems (ODES-8)*, Apr. 2010.

[5] R. Dreesen, T. Jungeblut, M. Thies, M. Porrmann, U. Kastens, and U. Rückert. A Synchronization Method for Register Traces of Pipelined Processors. In *Analysis, Architectures and Modelling of Embedded Systems*, pages 207–217. Springer Boston, 2009.

[6] T. Jungeblut, R. Dreesen, M. Porrmann, U. Rückert, and U. Hachmann. Design Space Exploration for Resource Efficient VLIW-Processors. In *University Booth of the Design, Automation and Test in Europe (DATE) conference*, 2008.

[7] T. Jungeblut, R. Dreesen, M. Porrmann, M. Thies, U. Rückert, and U. Kastens. A Framework for the Design Space Exploration of Software-Defined Radio Applications. In *The 2nd International ICST Conference on Mobile Lightweight Wireless Systems*, May 2010.

[8] T. Jungeblut, M. Grünewald, M. Porrmann, and U. Rückert. Real-time multiprocessor soc for mobile ad hoc networks. In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '07) – University Booth, 2007*, 16 - 20 Apr. 2007.

[9] T. Jungeblut, M. Grünewald, M. Porrmann, and U. Rückert. Realtime Multiprocessor for Mobile Ad Hoc Networks. *Advances in Radio Science*, 6:239–243, 2008.

[10] T. Jungeblut, D. Klassen, R. Dreesen, M. Porrmann, M. Thies, U. Rückert, and U. Kastens. Design space exploration for next generation wireless technologies. In *Proc. of the Electrical and Electronic Engineering for Communication Conference (EEEfCOM) 2009*, 2009.

[11] T. Jungeblut, S. Ltkemeier, G. Sievers, M. Porrmann, and U. Rckert. A Modular Design Flow for Very Large Design Space Explorations. In *Proceedings of the CDNLive! EMEA 2010, Munich, Germany, 2010*, May 2010.

[12] T. Jungeblut, C. Puttmann, R. Dreesen, M. Porrmann, M. Thies, U. Rückert, and U. Kastens. Resource Efficiency of Hardware Extensions of a 4-issue VLIW Processor for Elliptic Curve Cryptography. *Advances in Radio Science*, 2010.

[13] T. Jungeblut, G. Sievers, M. Porrmann, and U. Rückert. Design Space Exploration for Memory Subsystems of VLIW Architectures. In *5th IEEE International Conference on Networking, Architecture, and Storage (NAS 2010)*, July 2010.

[14] U. Kastens, D. K. Le, A. Slowik, and M. Thies. Feedback Driven Instruction-Set Extension. *ACM SIGPLAN Notices*, 39(7):135, 2004.

[15] D. Lattard. A Reconfigurable Baseband Platform Based on an Asynchronous Network-on-Chip. *IEEE J. Solid-State Circuits*, 43, 2008.

[16] D. Liu, A. Nilsson, E. Tell, D. Wu, and J. Eilert. Bridging Dream and Reality: Programmable Baseband Processors for Software-defined Radio. *Communications Magazine, IEEE*, 47(9):134–140, 2009.

[17] S. Mamidi, E. Blem, M. Schulte, J. Glossner, D. Iancu, A. Iancu, M. Moudgill, and S. Jinturkar. Instruction Set Extensions for Software Defined Radio on a Multithreaded Processor. In *Proceedings of the 2005 international conference on Compilers, architectures and synthesis for embedded systems*, pages 266–273. ACM, 2005.

[18] S. Mamidi, E. Blem, M. Schulte, J. Glossner, D. Iancu, A. Iancu, M. Moudgill, and S. Jinturkar. Instruction Set Extensions for Software Defined Radio. *Microprocessors and Microsystems*, 33(4):260–272, 2009.

[19] J.-C. Niemann. *Ressourceneffiziente Schaltungstechnik eingebetteter Parallelrechner – GigaNetIC*. 2009.

[20] J.-C. Niemann, C. Puttmann, M. Porrmann, and U. Rückert. Resource efficiency of the GigaNetIC chip multiprocessor architecture. *ScienceDirect, Systems Architecture*, 2006.

[21] M. Porrmann, J. Hagemeyer, C. Pohl, J. Romoth, and M. Strugholtz. *Parallel Computing: From Multicores and GPU's to Petascale, Advances in Parallel Computing*, volume 19, chapter RAPTOR – A Scalable Platform for Rapid Prototyping and FPGA-based Cluster Computing, pages 592–599. IOS press, 2010.

[22] C. Puttmann, J.-C. Niemann, M. Porrmann, and U. Rückert. Giganoc – A Hierarchical Network-on-Chip for Scalable Chip-Multiprocessors. 2007.

[23] E. Rijpkema. Trade offs in the design of a router with both guaranteed and best-effort services for network on chip. *IEEE Proc. Computers and Digital Techniques*, 150, 2003.

[24] TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide. Technical report, Texas Instruments, 2008.

[25] J.-W. van de Waerdt, S. Vassiliadis, S. Das, S. Mirolo, C. Yen, B. Zhong, C. Basto, J.-P. van Itegem, D. Amirtharaj, K. Kalra, P. Rodriguez, and H. van Antwerpen. The TM3270 Media-Processor. In *MICRO 38: Proceedings of the 38th annual IEEE/ACM International Symposium on Microarchitecture*, pages 331–342, Washington, DC, USA, 2005. IEEE Computer Society.

[26] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, et al. An 80-tile sub-100-W Teraflops Processor in 65-nm CMOS, 2008.