A Scalable Multi-Core ASIP Virtual Platform For Standard-Compliant Trellis Decoding

Matthias Jung, Christian Brehm, Norbert Wehn

Microelectronic Systems Design Research Group
University of Kaiserslautern, Germany

http://ems.eit.uni-kl.de

ABSTRACT

Multi standard wireless modems are becoming more and more important in industry. The recent move to LTE will aggravate this issue. We present a scalable Multi-Core ASIP virtual platform for trellis based channel decoding in multi-standard wireless modems. The basic building block of the platform is a weakly programmable IP-Core which was designed with the Processor Designer from Synopsys. This core has an implementation efficiency comparable to one of a dedicated architecture, however offers much more flexibility and supports convolutional and turbo code decoding for standards like GSM, EDGE, WiMax, CDMA2000, and LTE. The core was implemented in 90nm and 65nm respectively and is already in use in a commercial product. For a convenient design space exploration and scalability analysis, the multi-core architecture is modelled with Synopsys Platform Architect.

Table of Contents

Table of Figures

Table of Tables

1. Introduction

Today's and future wireless handsets like smart phones and tablet computers have to support multiple standards such as LTE, UMTS, HSPA and GSM (compare Table 1). Flexible modem architectures are needed, to support the seamless services between those different network standards. One essential task in a mobile device is the baseband processing, with the demand for design of flexible, power- and area-efficient solutions.

The channel decoder is one major building block in a wireless modem [1]. This computationally very intensive task requires high throughput in a very limited power budget[1]. The algorithms use non-standard operations and word widths. For this reason it is very inefficient to map these algorithms to standard DSPs. Channel decoding is therefore done either on dedicated decoders which are optimized for one standard, or A*pplication Specific Instruction set Processors* (ASIPs) [2]-[6]. ASIPs can be perfectly tailored according to the required flexibility [7]. Compared to dedicated decoders, they have the advantage of larger flexibility via programmability. This comes at the price of a slightly higher energy consumption per decoded bit.

For many applications, efficient ASIP designs are best derived from standard processor pipelines in a top-down manner. This is done by adding functionality and instructions for the most common kernel operations of the targeted algorithms, such as e.g. an FFT.

However, ASIPs in channel decoding designs often have not much in common with an enhanced standard processor pipeline. Only a limited flexibility is required. Energy- and area-efficiency demand for distributed memory embedded into the pipeline which are typical for many state-of-the-art decoding schemes and the demand for unifying the commonalities of several dedicated architectures. Fully customized deep pipelines with non-standard memory interfaces and instructions tailored to the targeted algorithms are the consequence. A minimal support for flow control operations is added, resulting in a weakly programmable architecture that offers no more than the desired flexibility. We also denote this type of ASIPs as *Weakly Programmable IP* Cores (WPIPs).

Table 1: Wireless Communication Standards and their Channel Codes

Standard	Channel code	Rate	Throughput
LTE data	binary turbo code (bTC)	1/3	150 MBit/s
LTE control	64 state convolutional (CC)	1/3	< 20 MBit/s
HSPA	binary turbo code (bTC)	1/3	42 MBit/s
UMTS	binary turbo code (bTC)	1/3	<1 MBit/s
	256 state convolutional (CC)	1/3	<1 MBit/s
GSM & EDGE	16 & 64 state conv. (CC)	1/2 - 1/4	<1 MBit/s
EDGE Evo	binary turbo code (bTC)	1/3	<1 MBit/s
CDMA2000	binary turbo code (bTC)	1/5	14.7 MBit/s
WiMax	duobinary turbo code (dbTC)	1/2 - 3/4	70 MBit/s

[1] The power budget of a whole handset is usually 2-3W, including display, power amplifier, applications processor, and baseband. The power used by the modem typically should not exceed 500mW.

In this paper we present a scalable multi-core virtual platform for trellis based channel decoding in multi-standard wireless modems. The basic building block of the platform is a weakly programmable and silicon proven[2] IP-Core, called *FlexiTreP* [15], which was designed with *Synopsys Processor Designer* [18]. For a faster design space exploration the multi-core architecture is modelled with *Synopsys Platform Architect* [19].

2. State of the Art

Many other research groups follow the approach of using programmable architectures for multi-standard channel decoding. There are coarse-grained heterogeneous architectures like Montimum [8] or SODA-II [9]. However as they are targeted to perform much more than channel decoding they have serious drawbacks w.r.t. power or throughput.

A much more promising approach deploys ASIPs dedicated to perform channel decoding only. Several publications report from ASIPs for binary (bTC) and duobinary (dbTC) turbo decoding, e.g. [10]. This multi-ASIP architecture targets high throughput for only a few standards and does not support convolutional decoding (CC). With a lower quantization they reach a high throughput and a small footprint, but the communication performance decreases with the lower quantisation.

[11] shows a proof-of-concept of a scalable homogeneous high-throughput architecture for turbo and LDPC decoding implemented in a 45 nm technology. The design is very energy-efficient, but due to a low clock frequency (150 MHz) also quite large. As the static power consumption increases with the area due to leakage, this reduces the energy efficiency during low to medium load situations.

Table 2: State of the Art Wireless Communication SoCs

Type/Publication	Channel Decoding Algorithms	Area [mm^2]	Power [nJ/bit/Iter]	Properties
Heterogeneous MPSoC [9]	CC, bTC	11	39 [3]	Baseband processing DSP
Heterogeneous MPSoC [8]	CC, bTC	0.52	n/a	Baseband processing System
MC-ASIP [10]	bTC, dbTC	1.5	n/a	90 nm, 4-bit input quantization, no UMTS support, shuffled decoding
MC-ASIC [11]	bTC, dbTC, LDPC	0.9	0.15	45 nm, proof-of-concept, 5 bit input quantization
MC-ASIP [12]	bTC, LDPC	0.062	0.32	45 nm, Power & Area only for RAMs & Crossbar
Heterogeneous MC-SoC [14]	bTC, dbTC, CC	3	0.3 (0.69)	65 nm, 6-bit input quantization, LTE Accelerator [13] + 2 ASIPs, Post P&R data

In [12] a scalable and reconfigurable architecture for turbo and LDPC decoding is presented. For the power and area analysis they claim that network and memory dominate the architecture and thus only considered those two components in the power and area estimation.

[2] FlexiTreP is used in a commercial baseband processing SoC.

[3] Power given for 256-states Viterbi Decoding

Except SODA and Montium which are not competitive in terms of power and throughput, all these architectures target only a very small set of all the modes defined in state-of-the-art wireless communication standards. In particular, none of them supports convolutional decoding which is yet deployed in practically every standard. In contrast to that, our ASIP fully supports for example GSM, EDGE, UMTS and LTE.

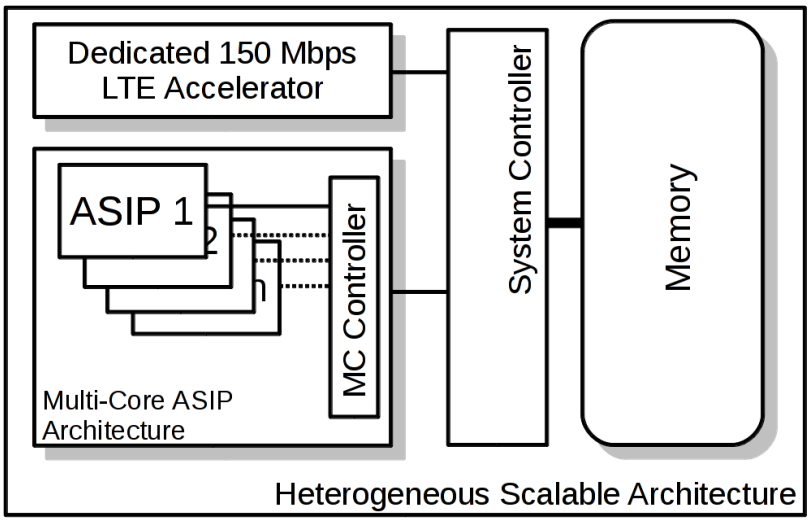

Figure 1: Example of a Heterogeneous Scalable Channel Decoding Architecture [14]

In [14] we presented a heterogeneous FlexiTreP based channel decoding architecture which is scalable and supports the newest high throughput LTE standard. However, mapping LTE onto the multi-ASIP cluster is not very energy and area efficient[4]. Therefore we decided to add dedicated LTE decoder [13] to the system, which requires low flexibility. All the other standards are mapped to the ASIPs. The memories, which represent the largest area in the system (compare Table 3), are shared between the ASIPs and the LTE block (compare Figure 1). The number of ASIPs is configurable to adapt the architecture to the system requirements, for example to the standards to be supported, or the field of application (modem or base-station). Table 2 lists the data of the mentioned architectures.

In this work we modelled the multi ASIP architecture with *Synopsys Platform Architect* as a virtual platform in a realistic and heterogeneous SoC environment to have a basis for further investigations on scalability. In the following we will first introduce the basic building block for our scalable platform and afterwards describe the platform and the model derived with strong support of Synopsys tools itself.

[4] 8 ASIPs would be needed, resulting in an area of $5mm^2$.

3. FlexiTreP - an ASIP modelled with Synopsys Processor Designer

FlexiTreP is a flexible Trellis processing engine. With its capability of decoding binary and duo-binary Turbo and convolutional codes it supports most important wireless communication standards, among others UMTS, LTE, DVB-SH, or WiMax. It comprises of 15 pipeline stages and seven memories that are accessed in different pipeline stages. The pipeline (compare Figure 2) is dynamically reconfigurable in order to react to code changes. The pipeline is implemented in a high-level processor description language, LISA [16]. From this description a cycle accurate C++ simulation model, a SystemC model with TLM2.0 interface as well as a synthesisable RTL model are generated using Synopsys Processor Designer [18]. Beside these models an assembler, a linker and documentation are generated as well. The development tools, together with the extensive profiling capabilities of the *Processor Debugger*, enable rapid analysis and exploration of the ASIP architecture to determine the optimal instruction set for the targeted application domain.

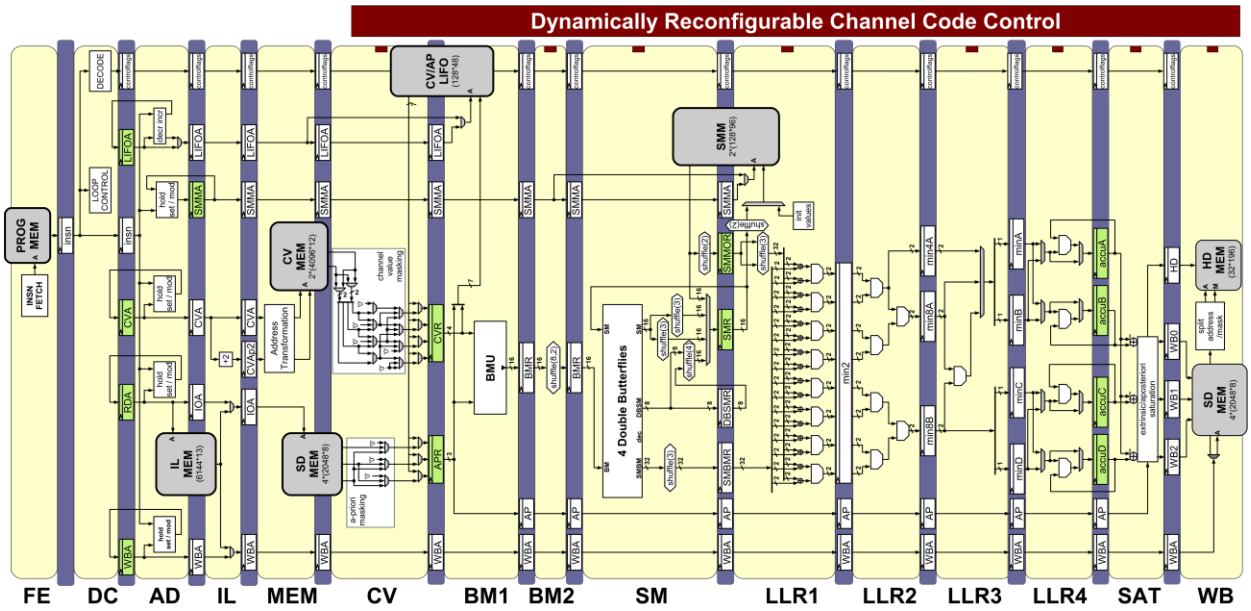

Figure 2: Pipeline Architecture of FlexiTreP

Compared to standard processors, ASIPs like FlexiTreP often have a very complex instruction set architecture due to the tight coupling between the instructions and the optimised micro-architecture. New validation concepts are required to deal with this issue. We developed a validation approach [17] that comprises formal verification methods by property checking as well as simulations and rapid-prototypes.

Table 3: Area and Power for one ASIP Instance

Power	86.5 *mW*
Area Memory	0.25 mm^2
Area Core	0.25 mm^2
Total Area	0.50 mm^2
Throughput (6 Iter.)	21 *Mbit/s*
Energy / Bit / Iter	0.69 *nJ/bit/Iter*

4. Multi-Core ASIP Platform modelled with Synopsys Platform Architect

Scalable Multi-Core ASIP Architecture

Scalability of the channel decoder architecture is an important demand for adaption to new standards and higher throughput requirements. A straight forward way would be to use many ASIPs in parallel, each with its own memory for each code block. This approach has two drawbacks: First, it does not improve the latency of a block, which could lead to problems in latency-critical standards and secondly, it is very inefficient in terms of area, as the memory is the largest piece of silicon in a low to medium throughput decoder. Therefore we have to parallelise the calculation of the block itself.

The calculation consists of a forward and a backward recursion. Both of them have to be calculated sequentially. However, it is possible to split the block into subblocks and distribute it to several ASIP cores to decrease the latency of the block. Dependencies evolve from splitting the block, which can be resolved by exchanging the metrics on the block edges. This is a common practice for dedicated hard core decoders [20], and we adopted it for our architecture in [14].

The more cores are used, the more potential problems may occur. For example the interleaver specified in the HSPA standard is not conflict-free. These memory access conflicts need to be analysed and resolved e.g. by stalling one of the cores. A virtual prototyping platform is perfectly suited for finding smart solutions regarding these conflicts. As a basis for these investigations we created a virtual platform with several ASIPs.

Generic FlexiTreP Multi-Core Virtual Platform

Virtual platforms are high-speed, fully functional software models of physical hardware systems which can include different models like processors, memories, IP blocks and interconnect components. They give the ability to significantly accelerate development cycles, because the software can be developed on the VP before the real hardware is available and they give the ability for a design space exploration on system level.

Synopsys Platform Architect [19] is a SystemC and TLM2.0 based graphical virtual platform environment for designing, exploring, optimising and debugging system architecture. This tool is perfectly suited for the design space exploration of our multi-core channel decoding environment. The platform is assembled and configured in *Platform Creator*. It is possible to choose components from a large library and connect them together with custom components by drag-and-drop or generate scalable platforms with scripts by means of a powerful *Tool Command Language* (TCL) interface. For our work we decided to use the TCL interface because of the convenient way to increase the number of cores for the desired scalability analysis.

Our platforms consist of several FlexiTreP TLM2.0 blocks, which are generated with Processor Designer. For the controlling and the synchronisation of the ASIP cores an additional SystemC ASIP-Controller is used. If the complexity of the platform increases we use an ARM general purpose processor TLM2.0 model for the controlling to guarantee flexibility. There are vari-

ous special memories which are shared between the FlexiTrePs. The communication infrastructure is automatically generated from *Platform Creator* according to a memory map. This was a very helpful feature for us, because writing the whole infrastructure by hand consumes a lot of time and efforts. It is possible to switch between a loosely-timed (LT) simulation, to get high simulation speed, and an approximately-timed (AT) simulation. The LT simulation is used when we analyse conflict-free standards and the AT simulation is used with conflict-prone standards. By way of illustration, Figure 3 shows a small platform with two ASIP instances.

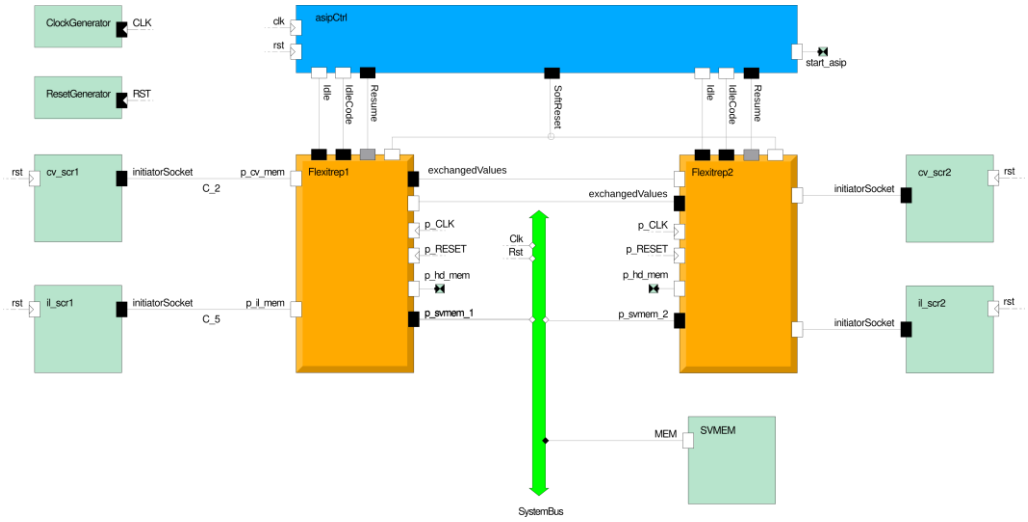

Figure 3: Example Platform with 2 ASIP Cores

After compiling the platform, it can be analysed and debugged with different tools. *Platform Analyzer* serves as a powerful debugging tool. With it, we can analyse the contents of the registers and memories, we are able to step through our assembler programs and we can set break points into the code or on signals. To have a detailed look on the ASIP pipeline it is possible to connect *Processor Debugger* to the platform simulation as well. These debugging capabilities are mandatory for the proper multi-core software development. With *SystemC Explorer* we are able to analyse the bus transactions by means of a graphical transaction tracing that allowed us to identify the main conflicts of this channel decoding architecture.

5. Conclusion

Virtual Platforms offer a new efficient approach for system level design and software development, with the possibility of decreasing time-to-market and achieve a better product quality. In this paper it was shown that the work with virtual prototypes is suitable for the development of complex channel decoding architectures.

Different tools and procedures of virtual prototyping were explained. It was shown that a virtual prototype can be created easily by means of the Synopsys Virtual Platform Architect. A platform can be enhanced with additional components in a convenient way (e.g. to a multi-core system or a heterogeneous platform). This leads to a flexible product development and the possibility to reuse components for future projects. The work with virtual prototypes is a powerful and worthwhile approach for system level design.

6. References

[1] J. Dielissen, N. Engin, S. Sawitzki, and K. van Berkel, "Multistandard FEC Decoders for Wireless Devices," IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 55, no. 3, pp. 284-288, Mar. 2008.

[2] B. Bougard, R. Priewasser, L. V. der Perre, and M. Huemer, "Algorithm- Architecture Co-Design of a Multi-Standard FEC Decoder ASIP," in ICT-MobileSummit 2008 Conference Proceedings, Stockholm, Sweden, Jun. 2008.

[3] S. Kunze, E. Matus, and G. P. Fettweis, "ASIP decoder architecture for convolutional and LDPC codes," in Proc. IEEE International Symposium on Circuits and Systems ISCAS 2009, May 2009, pp. 2457-2460.

[4] F. Naessens, B. Bougard, S. Bressinck, L. Hollevoet, P. Raghavan, L. Van der Perre, and F. Catthoor, "A unified instruction set pro- grammable architecture for multi-standard advanced forward error correction," in Proc. IEEE Workshop on Signal Processing Systems SiPS 2008, Oct. 2008, pp. 31-36.

[5] O. Muller, A. Baghdadi, and M. Jezequel, "From Parallelism Levels to a Multi-ASIP Architecture for Turbo Decoding," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 17, no. 1, pp. 92-102, Jan. 2009.

[6] M. Alles, T. Vogt, and N. Wehn, "FlexiChaP: A Reconfigurable ASIP for Convolutional, Turbo, and LDPC Code Decoding," in Proc. 5th International Symposium on Turbo Codes and Related Topics, Lausanne, Switzerland, Sep. 2008, pp. 84-89.

[7] A. Hoffmann, O. Schliebusch, A. Nohl, G. Braun, O. Wahlen, and H. Meyr, "A methodology for the design of application specific in- struction set processors (ASIP) using the machine description language LISA," in Computer Aided Design, 2001. ICCAD 2001. IEEE/ACM International Conference on, Nov. 2001, pp. 625-630.

[8] G. Rauwerda, G. Smit, C. van Benthem, and P. Heysters, "Reconfig- urable Turbo/Viterbi Channel Decoder in the Coarse-Grained Montium Architecture," in Proceedings of the International Conference on Engi- neering of Reconfigurable Systems and Algorithms (ERSA'06). USA: CSREA Press, June 2006, pp. 110-116.

[9] H. Lee, C. Chakrabarti, and T. Mudge, "A Low-Power DSP for Wireless Communications," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol. 18, no. 9, pp. 1310-1322, 2010.

[10] R. Al-Khayat, P. Murugappa, A. Baghdadi, and M. Jezequel, "Area and throughput optimized ASIP for multi-standard turbo decoding," in Proc. 22nd IEEE Int Rapid System Prototyping (RSP) Symp, 2011, pp. 79-84.

[11] G. Gentile, M. Rovini, and L. Fanucci, "A multi-standard flexible turbo/LDPC decoder via ASIC design," in Proc. 6th Int Turbo Codes and Iterative Information Processing (ISTC) Symp, 2010, pp. 294-298.

[12] F. Naessens, B. Bougard, S. Bressinck, L. Hollevoet, P. Raghavan, L. Van der Perre, and F. Catthoor, "A unified instruction set pro- grammable architecture for multi-standard advanced forward error cor- rection," in Proc. IEEE Workshop on Signal Processing Systems SiPS 2008, Oct. 2008, pp. 31-36.

[13] M. May, T. Ilnseher, N. Wehn, and W. Raab, "A 150Mbit/s 3GPP LTE Turbo Code Decoder," in Proc. Design, Automation and Test in Europe, 2010 (DATE '10), Mar. 2010, pp. 1420-1425.

[14] Brehm, C.; Ilnseher, T.; Wehn, N.; , "A scalable multi-ASIP architecture for standard compliant trellis decoding," SoC Design Conference (ISOCC), 2011, pp.349-352

[15] Vogt, T.; Wehn, N.; , "A Reconfigurable ASIP for Convolutional and Turbo Decoding in an SDR Environment," Very Large Scale Integration (VLSI) Systems, IEEE Transactions on , vol.16, no.10, pp.1309-1320, Oct. 2008

[16] A. Hoffmann, O. Schliebusch, A. Nohl, G. Braun, O. Wahlen, and H. Meyr, "A methodology for the design of application specific in- struction set processors (ASIP) using the machine description language LISA," in Computer Aided Design, 2001. ICCAD 2001. IEEE/ACM International Conference on, Nov. 2001, pp. 625-630.

[17] Brehm, C.; Wehn, N.; Loitz, S.; Kunz, W.; , "Validation of channel decoding ASIPs a case study," Rapid System Prototyping (RSP), 2011 22nd IEEE International Symposium on , pp.74-78, 24-27, May 2011

[18] "Synopsys Processor Designer", February 2012. [Online]. Available: http://www.synopsys.com/Systems/BlockDesign/ProcessorDev/

[19] "Synopsys Platform Architect", February 2012. [Online]. Available: http://www.synopsys.com/Systems/VirtualPrototyping

[20] Thul, M.J.; Gilbert, F.; Vogt, T.; Kreiselmaier, G.; Wehn, N.; , "A scalable system architecture for high-throughput turbo-decoders," Signal Processing Systems, 2002. (SIPS '02). IEEE Workshop on , vol., no., pp. 152- 158, 16-18 Oct. 2002