

Architectural Analysis of a Smart DMA Controller for Protocol Stack Acceleration in LTE Terminals

Sebastian Hessel, David Szczesny, Felix Bruns, Attila Bilgic

Institute for Integrated Systems, Ruhr-Universität Bochum

D-44780 Bochum, Germany

Email: {sebastian.hessel,david.szczesny,felix.brunns,attila.bilgic}@is.rub.de

Josef Hausner

Infineon Technologies AG

D-85579 Neubiberg, Germany

Email: hausner@ieee.org

Abstract—In this paper we present an architectural analysis of a smart DMA (sDMA) controller for protocol stack acceleration in mobile devices supporting 3GPP's Long Term Evolution (LTE). This concept already demonstrated a significant performance benefit over conventional approaches by on-the-fly header decoding and deciphering for the data plane of the LTE protocol stack layer 2 in downlink direction. With a low-level hardware implementation we prove that also from an architectural point of view the sDMA controller is suitable for LTE terminals. Compared to conventional hardware acceleration, chip area and energy consumption are reduced by 10 % and 56 %, respectively. Furthermore, we show that the header decoding has the highest architectural impact on the sDMA controller. By a change of the hardware/software partitioning within the header decoding unit, the chip area of the sDMA controller is decreased by 35 %, while it consumes 39 % less power. The improvement compared to the conventional approach (with the same modification) is then even increased to 17 % (area) and 59 % (energy).

I. INTRODUCTION

Next generation mobile devices supporting 3GPP's Long Term Evolution (LTE) will face another increase of the data rates at simultaneously reduced latencies. Consequently, the processing demand in the radio architecture will also grow with these requirements [1]. Besides the physical layer, the data plane of higher protocol layers has to be exhaustively investigated regarding a suitable hardware/software partitioning, since up to now only cryptographic algorithms are offloaded to hardware in mobile platforms. Performance analyses, however, have already shown [2] that a hardware support will be needed for even more protocol stack functionality in LTE systems. Additionally, the concepts for this hardware acceleration have to be chosen carefully in order to identify a suitable system architecture that offers the required performance while keeping the processor clock frequency and thus the energy budget at a reasonable level.

A promising solution of this problem is given by the smart DMA (sDMA) controller presented in [3]. This peripheral contains a conventional Direct Memory Access (DMA) engine expanded by hardware accelerators for data plane processing of the LTE protocol stack layer 2 (L2) in downlink direction. By performing on-the-fly header decoding and deciphering, conventional hardware accelerator concepts are outperformed by up to 80 %. This is shown by cycle approximate system level simulations using a Virtual System Prototype (VSP) [4]–[6] of an LTE based mobile phone platform. For a mobile device,

however, with its limited resources like chip area and battery lifetime, it is also important to prove the suitability of such a concept regarding a hardware implementation. We therefore present an architectural exploration of the sDMA controller on register transfer level according to [6]. By VHDL model synthesis we are able to give accurate area and power estimations for the sDMA controller itself as well as a comparison to conventional hardware acceleration.

This paper is organized as follows: Section II provides an introduction of the sDMA controller, followed by its architectural analysis and the comparison with a conventional hardware accelerator in Section III. Finally, the conclusion is given in Section IV.

II. THE SMART DMA (sDMA) CONTROLLER

The structure of the implemented sDMA controller is shown in Fig. 1. It includes two bus interfaces for the communication with other devices: an ARM AMBA Advanced Peripheral Bus (APB) and an ARM AMBA Advanced High-performance Bus (AHB) interface. While the APB is used to read or write the 32-bit configuration registers within the sDMA controller, the transfer of data (i.e. transport blocks) to or from other peripherals is done via the AHB interface. The smart DMA engine

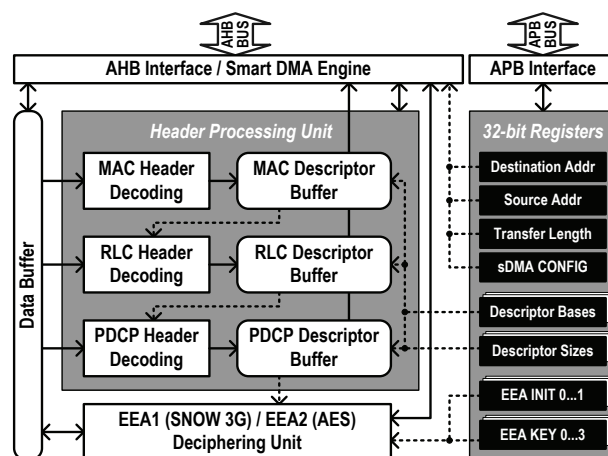


Fig. 1. Structure of the smart DMA (sDMA) controller. The smart DMA engine includes a conventional DMA engine extended by additional logic to control the hardware accelerator units.

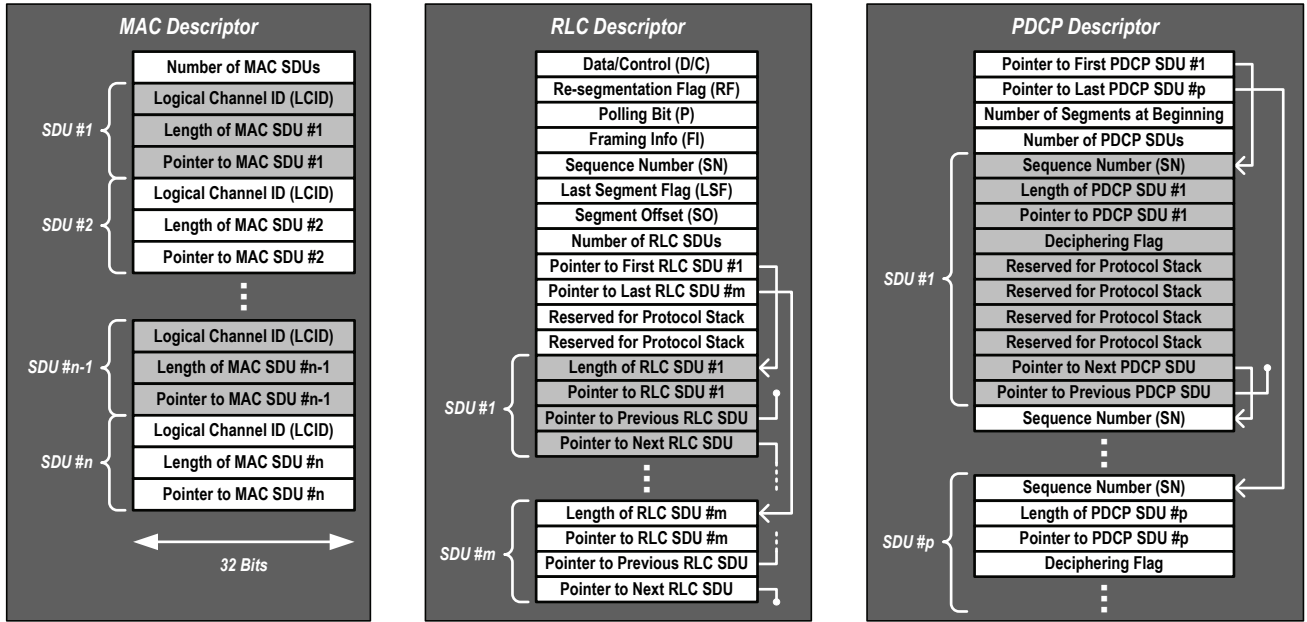


Fig. 2. Structure of the MAC (left), RLC (middle) and PDCP descriptor. All descriptors include static as well as dynamic header information. The latter depends on the number of the respective SDUs during transmission. Furthermore, linked lists are used in the RLC and PDCP descriptor.

consists of a conventional DMA engine with one data transfer channel that enables copy operations to or from the internal data buffer. The source and destination addresses as well as the transfer length (in bytes) are set by the corresponding registers. The DMA engine supports word-aligned 32-bit data transfers at a maximum burst size of 64 words. It is activated by writing the configuration register (sDMA CONFIG). Additional logic in the smart DMA engine controls the processing flow of the hardware accelerator units (header processing and deciphering) and their read and write accesses to the data buffer.

The sDMA controller has three different clock domains. The hardware accelerator units operate with the fastest clock (CLK), while slower clocks are adopted for the APB interface (PCLK) and the AHB interface/smart DMA engine (HCLK).

A. Header Processing Unit

The header processing unit provides on-the-fly header decoding for the LTE L2 data plane. Consequently, it contains decoding units for the Medium Access Control (MAC), the Dedicated Traffic Channel (DTCH) in the Radio Link Control (RLC) and the Packet Data Convergence Protocol (PDCP) sublayers. Their implementations are compliant with the 3GPP Release 8 specifications [7]–[9]. Each unit has a descriptor buffer storing the header information of the corresponding sublayer. After a transport block is processed, the descriptor buffers are written to a defined memory region accessible by the protocol stack. The base addresses as well as the size of the allocated memory region are given by the corresponding registers for each descriptor buffer. A detailed overview of the descriptors is shown in Fig. 2. They contain static and dynamic parts, where the latter depend on the number of the respective Service Data Units (SDUs) during transmission. Moreover,

linked lists are generated in the RLC and PDCP descriptor to improve the descriptor integration in the protocol stack.

B. Deciphering Unit

The deciphering unit includes the two EPS (Evolved Packet System) Encryption Algorithms (EEA) which are specified by the 3GPP for user data confidentiality in LTE [10]: the SNOW 3G algorithm and the Advanced Encryption Standard (AES). The AES is thereby used in the Counter (CTR) mode of operation [11]. The parameters needed in the deciphering unit are shown in Table I. The EEA INIT registers assign the values for BEARER, LENGTH and COUNT, where the latter is maintained by the protocol stack during transport block processing. Additionally, both algorithms apply an initial 128-bit key (KEY) that is provided by the Radio Resource Control (RRC) via the EEA KEY registers. The output of the AES is a 128-bit keystream block whose generation needs at least 11 round transformations (and the same number of

Parameter	Size (bits)	Description
KEY	128	Initial cipher key
BEARER	5	Radio bearer identity
DIRECTION	1	Transmission direction (0 for uplink, 1 for downlink)
LENGTH	16	Number of bits to be processed
COUNT	32	Counter value*

*composed of the RRC hyper frame number and the PDCP sequence number (SN)

TABLE I
PARAMETERS USED FOR USER DATA CONFIDENTIALITY IN LTE [9]. THE LOWER FOUR VALUES ARE PROVIDED BY THE EEA INIT REGISTERS.

clock cycles) depending on the data path widths [12], while the SNOW 3G generates a 32-bit keystream block every clock cycle after an initialization phase of 32 clock cycles [13]. With both algorithms, the keystream block is then adopted for the decryption of the ciphertext by an XOR operation [10]. In LTE, user data decryption is located in the PDCP sublayer and is applied to the data part in the PDCP Protocol Data Units (PDUs). Therefore, the deciphering unit is activated after a PDCP header is decoded. Its completion is indicated by setting the deciphering flag in the corresponding field of the PDCP descriptor (see Fig. 2). It should be noted that on-the-fly deciphering within the sDMA controller is carried out only for complete PDCP SDUs. PDCP PDU segments, however, are decrypted separately after they are reassembled in the protocol stack. A maximum number of two segments is possible per transport block.

III. ARCHITECTURAL ANALYSIS

For the architectural exploration of the sDMA controller, hardware characteristics are obtained from a VHDL model synthesis with Synopsys' DesignVision™ Version B-2008.09. We adopt a standard cell library of Faraday's 90 nm CMOS technology with a core power supply of 1.0V. In the illustrations of the area efforts, the values are given in thousands of gate equivalents (kGE). The number of gate equivalents is calculated from the total area divided by the size of a 2-input AND ($5\mu\text{m}^2$) that is taken from the technology library. For the estimations of the average power consumption, activity data from netlist simulations is used in order to get realistic power values. A transport block of 12500 bytes (given from VSP simulations) is processed during the simulations. This amount of data corresponds to a slightly increased payload compared to the maximum data rate of 100Mbit/s over the air interface specified for the LTE downlink. With an IP packet size of 1000 bytes, 12 complete and two segmented PDCP PDUs are located in one transport block. Consequently, 12 PDCP SDUs have to be deciphered on-the-fly. Additionally, dummy data for other logical channels as well as control data is included in the transport block in order to have a more realistic decoding effort in the MAC sublayer (nine MAC headers in total). Unless stated differently, we therefore choose a {MAC,RLC,PDCP} descriptor size of two to the power of {6,7,8} for the architectural analysis.

The test environment for the simulations is depicted in Fig. 3. The testbench copies the transport block from a file to memory #0. Afterwards, the sDMA controller is configured and activated by setting its registers. Processed data as well as the descriptors are written to memory #1 during simulation. After completion, the data in memory #1 is moved to an output file for verification with the VSP results. Alternatively, a conventional hardware accelerator concept, composed of a DMA controller and a stand-alone hardware acceleration unit, can be attached to the testbench for architectural comparisons with the sDMA controller. In this setup, the stand-alone hardware accelerator contains the same functionality as the sDMA controller except for the DMA engine that is replaced

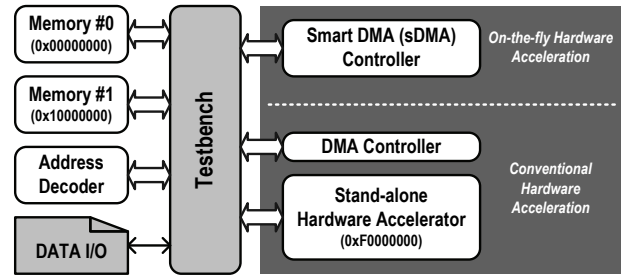


Fig. 3. Structure of the simulation environment for the sDMA analysis. Alternatively, a conventional DMA controller combined with a stand-alone hardware accelerator can be attached to the testbench for comparison purposes.

by an AHB slave interface and realized separately with the DMA controller.

In the following, the area and power results are plotted against the fastest clock (CLK). The two other clocks, HCLK and PCLK, are always set to be five times and ten times, respectively, slower than CLK.

A. Header Processing Unit

The sizes of the descriptor buffers in the header processing unit depend on the number of SDUs in the corresponding sublayer (see Fig. 4). The more SDUs have to be stored, the bigger the buffer size is. The PDCP descriptor buffer has the steepest increase in size due to the highest number of descriptor elements per SDU (see Fig. 2). The influence of the descriptor sizes on the total area of the header processing unit can be seen in Fig. 5. With doubling the descriptor sizes, the area is also almost doubled. An alternative to reduce the area effort is given by calculating only the relative offset values within the descriptors instead of the full pointer values adding the 32-bit descriptor base addresses. With this approach, the bit width of the corresponding logic (mainly registers and adders) can be nearly halved (assuming a maximum transport block length of 12500 bytes) resulting in an area reduction of approximately one third for the analyzed descriptor sizes.

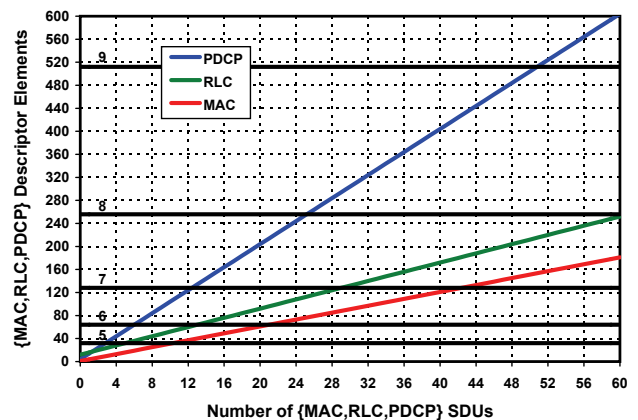


Fig. 4. Descriptor size against the number of Service Data Units (SDUs) for the three sublayers MAC, RLC and PDCP. The horizontal black lines show the threshold for descriptor sizes to the power of two.

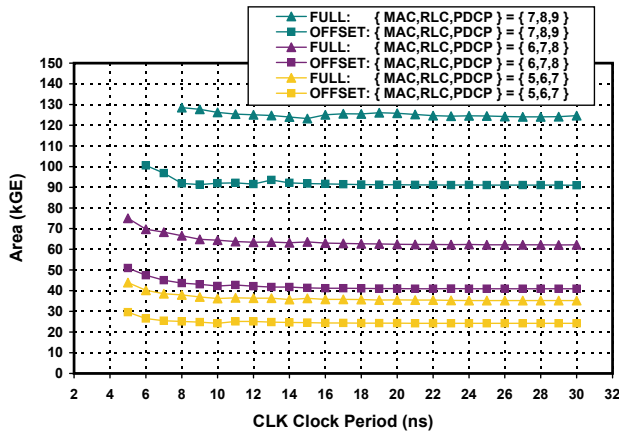


Fig. 5. Area effort of the header processing unit for different descriptor sizes: 'FULL' denotes a calculation of the full pointer values (including the base addresses) and 'OFFSET' denotes a calculation of the offset values only.

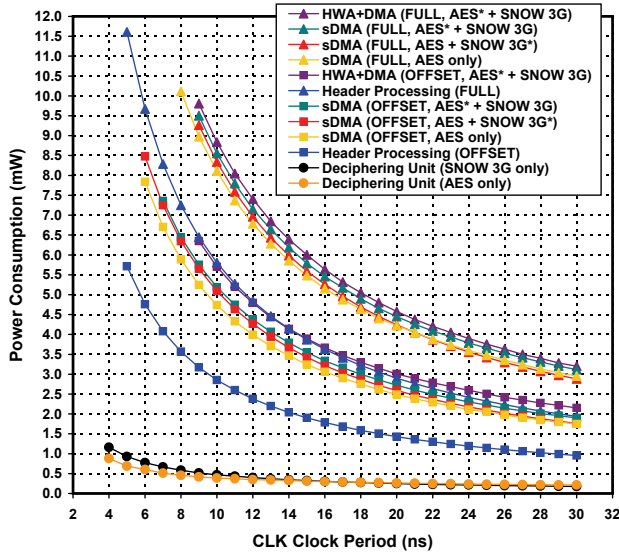


Fig. 6. Power consumption of the sDMA controller and its sub-units for descriptor sizes of {6,7,8} for {MAC,RLC,PDCP}. Additionally, the results for a conventional hardware accelerator approach (HWA+DMA) are shown. The active deciphering algorithm is marked with (*).

Looking at the power consumption of the header processing unit (see blue curves in Fig. 6), a decrease of almost 50% can be observed. On the other hand, the descriptor base addresses have then to be added to the offset values in the protocol stack and thus by the processor. A check of the performance in the VSP according to [3], however, shows that the execution time of the protocol stack is almost equal in both cases (see Fig. 7). Nevertheless, additional power will then be consumed by the processor subsystem which relativizes this power benefit of the sDMA controller in a mobile phone platform. Finally, the contribution of the header processing unit to the sDMA controller regarding the area is nearly 40% (33%) and up to 68% (55%) from a power perspective. The second value in brackets thereby (as well as in the following) denotes the value for the offset calculation only.

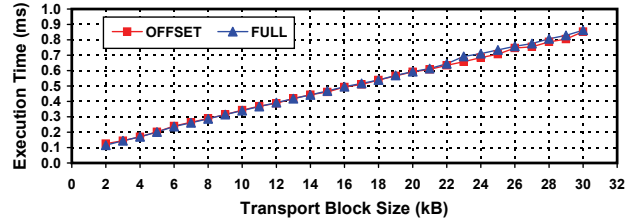


Fig. 7. Execution time of the protocol stack against the transport block size comparing the full pointer calculation with the offset mode.

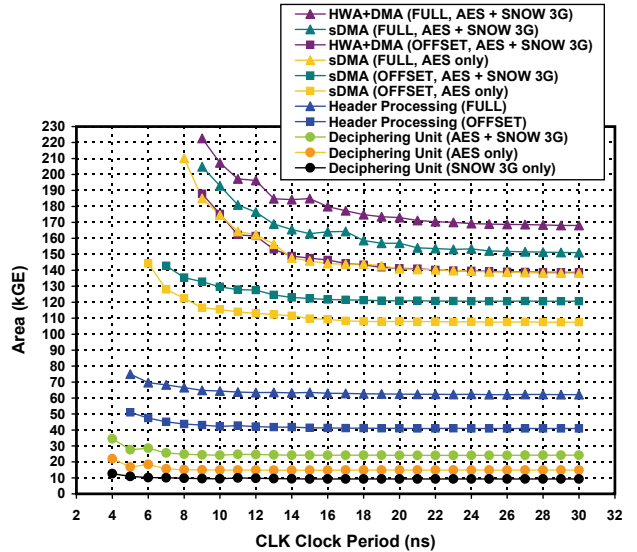


Fig. 8. Area effort of the sDMA controller and its sub-units for descriptor sizes of {6,7,8} for {MAC,RLC,PDCP}. Additionally, the results for a conventional hardware accelerator approach (HWA+DMA) are shown.

B. Deciphering Unit

The main focus in the design of the deciphering algorithms is the identification of suitable cryptographic substitution box (S-Box) implementations for both algorithms as well as a suitable data path width for the AES in order to achieve a reasonable trade-off between power consumption and area effort. We therefore implemented a one-hot encoder and decoder with a suitable switching block in between for the S-Boxes [14] and a 128-bit data path for the AES (resulting in 11 round transformations). This setup is already identified in [15] as the most promising solution for deciphering in LTE terminals. The area values in Fig. 8 show that the complete deciphering unit needs only approximately 15% (20%) of the total area of the sDMA controller. The area contribution of the AES to the deciphering unit is slightly bigger (60%) than the one of the SNOW 3G algorithm. Regarding the power consumption, the AES consumes up to almost 7% (11%) of the total energy budget, whereas deciphering with the SNOW 3G algorithm needs 6% (10%).

C. sDMA Controller

With the calculation of the offset values only in the header processing unit, the total area of the sDMA controller can

be decreased between 20 % and 35 % (see Fig. 8). As a side effect, a higher clock frequency is achieved because of a shorter critical path. At the same time, the power consumption is reduced by approximately 39 % (see Fig. 6) adopting the AES. In general, the difference of the power consumption between the decryption with the AES and the decryption with the SNOW 3G algorithm is negligible, although an active AES results in a slightly higher power consumption due to a more complex control logic in the smart DMA engine. In case only the AES is implemented in the deciphering unit, the overall area is reduced by nearly 9 % (11 %). It is clear that there is only a minor benefit with respect to the power consumption compared to an architecture with both decryption algorithms, since only one algorithm is active at a time.

The comparison between the conventional hardware accelerator and the sDMA concept shows that the area of the conventional approach is 10 % (17 %) bigger. This is mainly caused by an additional data buffer (since a data buffer is needed in the DMA controller as well as in the stand-alone hardware accelerator) and the additional AHB slave interface in the stand-alone hardware accelerator. Looking at the power values, the increase is about 3 % (11 %). In order to have a fair comparison with respect to the energy budget, it is necessary to check the energy consumption of the two concepts as well, since the processing time is significantly reduced with the sDMA controller due to less data copy operations [6]. Instead of 15456 clock cycles adopting the conventional approach, only 7000 clock cycles are needed for the transport block processing with the sDMA concept. Regarding the investigated CLK clock periods and the respective values of the power consumption (see Fig. 6), this results in an average reduction of the energy consumption of about 56 % (59 %).

IV. CONCLUSION

In this paper we present an architectural exploration of a smart DMA (sDMA) controller for protocol stack acceleration in Long Term Evolution (LTE) terminals. Besides conventional DMA functionality, the sDMA controller enables on-the-fly header decoding and deciphering for the data plane of the protocol stack layer 2 in downlink direction. Cycle approximate system level simulations have already shown maximum speedups of up to 80 % compared to a conventional hardware accelerator concept [3]. We prove that also an architectural benefit is given by this concept: its implementation is 10 % smaller and consumes 56 % less energy. With a closer look at the sDMA controller, the header processing unit comes up with the highest contribution regarding the area effort (40 %) and the average power consumption (up to 68 %). On the other hand, the deciphering unit (including the AES and the SNOW 3G) needs only 15 % of the total area and consumes 7 % of the power budget when the data is decrypted with the AES (6 % with the SNOW 3G). Finally, we show that a change in the header processing unit leads to a further architectural improvement of the sDMA concept. By the calculation of only the offset values instead of the full pointers in the descriptors, area and power consumption can be reduced by 35 % and

39 %, respectively, with a negligible penalty on the sDMA controller performance within the mobile phone platform. The architectural benefit compared to the conventional concept with the same modification is then given by 17 % (area) and 59 % (energy) which again strengthens the overall suitability of the sDMA controller in LTE terminals.

ACKNOWLEDGMENT

The authors acknowledge the excellent cooperation with all project partners within the EASY-C project and the support by the German Federal Ministry of Science and Education (BMBF). Further information is available on the project website: <http://www.easy-c.de>.

REFERENCES

- [1] C.H. van Berkel, "Multi-Core for Mobile Phones", in *Design, Automation & Test in Europe Conference (DATE '09)*, Nice, France, pp. 1260-1265, Apr. 2009.
- [2] D. Szczesny, A. Showk, S. Hessel, U. Hildebrand, V. Frascolla and A. Bilgic, "Performance Analysis of LTE Protocol Processing on an ARM based Mobile Platform", in *11th International Symposium on System-on-Chip (SoC 2009)*, Tampere, Finland, pp. 56-63, Oct. 2009.
- [3] D. Szczesny, S. Hessel, F. Bruns and A. Bilgic, "On-the-fly Hardware Acceleration for Protocol Stack Processing in Next Generation Mobile Devices", in *7th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2009)*, Grenoble, France, pp. 155-162, Oct. 2009.
- [4] T. Eckart and M. Schnieringer, "Development and Verification of Embedded Firmware using Virtual System Prototypes", in *International Symposium on System-on-Chip (SoC 2006)*, Tampere, Finland, pp. 1-1, Nov. 2006.
- [5] M. Brandenburg, A. Schollhorn, S. Heinen, J. Eckmüller and T. Eckart, "From Algorithm to First 3.5G Call in Record Time - A Novel System Design Approach Based on Virtual Prototyping and its Consequences for Interdisciplinary System Design Teams", in *Design, Automation and Test in Europe (DATE 2007)*, Nice, France, pp. 1-3, Apr. 2007.
- [6] S. Hessel, D. Szczesny, S. Traboulsi, A. Bilgic and J. Hausner, "On the Design of a Suitable Hardware Platform for Protocol Stack Processing in LTE Terminals", in *7th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC-09)*, Vancouver, Canada, pp. 1-8, Aug. 2009.
- [7] *Evolved Terrestrial Radio Access (E-UTRA): Medium Access Control (MAC) Protocol Specification*, 3GPP Std. TS 36.321, Rev. 8.7.0, Sep. 2009.
- [8] *Evolved Universal Terrestrial Radio Access (E-UTRA): Radio Link Control (RLC) Protocol Specification*, 3GPP Std. TS 36.322, Rev. 8.7.0, Sep. 2009.
- [9] *Evolved Universal Terrestrial Radio Access (E-UTRA): Packet Data Convergence Protocol (PDCP) Specification*, 3GPP Std. TS 36.323, Rev. 8.6.0, June 2009.
- [10] *3GPP System Architecture Evolution (SAE): Security Architecture*, 3GPP Std. TS 33.401, Rev. 8.2.1, Dec. 2008.
- [11] National Institute of Standards and Technology (NIST), "NIST Special Publication 800-38A: Recommendation for Block Cipher Modes of Operation - Methods and Techniques", Dec. 2001, available: <http://csrc.nist.gov/publications/PubsSPs.html>.
- [12] National Institute of Standards and Technology (NIST), "FIPS Publication 197: Advanced Encryption Standard (AES)", Nov. 2001, available: <http://csrc.nist.gov/publications/PubsFIPS.html>.
- [13] *Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 & UIA2. Document 2: SNOW 3G Specification*, ETSI/SAGE Specification, Version 1.1, Sep. 2006.
- [14] G. Bertoni, M. Macchetti, L. Negri and P. Fragneto, "Power-efficient ASIC Synthesis of Cryptographic Sboxes", in *14th Great Lakes Symposium on VLSI (GLSVLSI 2004)*, Boston, USA, pp. 277-281, Apr. 2004.
- [15] S. Hessel, D. Szczesny, N. Lohmann, A. Bilgic and J. Hausner, "Implementation and Benchmarking of Hardware Accelerators for Ciphering in LTE Terminals", in *52nd IEEE Global Communications Conference (GLOBECOM 2009)*, Honolulu, Hawaii, USA, pp. 1-7, Dec. 2009.