Design and Test of a Gigabit Ethernet MAC for High-Speed HIL-Support

Alexander Rohleder, Steffen Jaeckel, Axel Sikora

***Abstract*—The efficient support of Hardwae-In-the-Loop (HIL) in the design process of hardware-software-co-designed systems is an ongoing challenge. This paper presents a network-based integration of hardware elements into the software-based image processing tool „ADTF“, based on a high-performance Gigabit Ethernet MAC and a highly-efficient TCP/IP-stack. The MAC has been designed in VHDL. It was verified in a SystemC-simulation environment and tested on several Altera FPGAs.**

***Index Terms*—Hardware-Software Co-Simulation, Hardware In the Loop (HIL), Gigabit Ethernet MAC, VHDL, SystemC.**

I. INTRODUCTION

The design of complex hardware-software-co-design systems mostly starts with its behavioural and high-level description in software [1] [2]. The subsequent process of stepwise implementation and verification is an ongoing challenge, as high-level software blocks must be executed in combination with Hardware-In-the-Loop (HIL) elements. Existing solutions often are proprietary, requiring dedicated hardware and software interface, as e.g. [3].

The development of advanced driver assistance systems (ADAS) is a key element for modern and safe vehicles. These systems may use contact- or non-cooperative perception sensor systems for the detection, such as CMOS-cameras, RADAR or LIDAR. The subsequent processing steps might have a very high complexity. This will be especially the case, if the objects, which shall be detected, have a variety of possible classifiers. And this is the case for predictive pedestrian protection systems (PPPS) [4], as they are recently available from some car manufacturers.

However, these systems will find a broad dissemination, if their cost in mass production is low enough, which calls for hardware-assisted solutions. One example was presented in [5].

Alexander Rohleder, alexander.rohleder@stzedn.de, Steffen Jaeckel, steffen.jaeckel@stzedn.de, Axel Sikora, axel.sikora@stzedn.de, Poststr. 35, 79423 Heitersheim, Germany

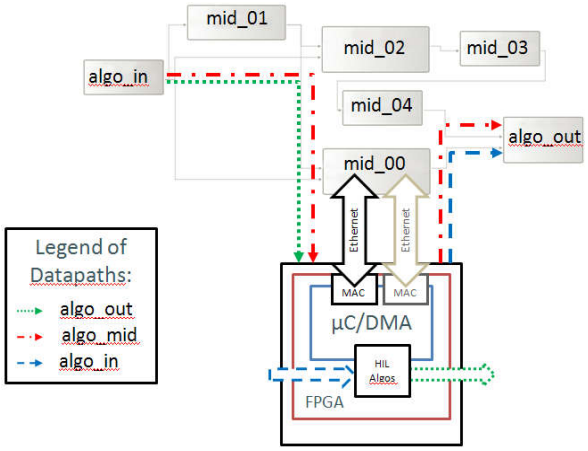

Fig. 1 Integration of hardware modules into ADTF

II. THE HIL APPROACH

A. The Basic Framework

For the development of advanced driver assistance systems (ADAS), common frameworks are used by practically all European OEMs and 1st tier suppliers, which allow an easy combination of processing blocks being described in high-level software, and is a valuable tool for the evaluation of different algorithms and their suitability for real input streams, i.e. video streams from the CMOS cameras.

After the algorithms are evaluated and approved, system developers typically start to migrate the functional blocks successively into firmware or hardware implementations.

The verification of the system is only possible after the process of full migration to the productive system. There exists no standardized possibility for early integration of hardware elements into the framework.

Especially the “Automotive Data and Time triggered Framework” (ADTF) is widely used. The restrictions described above are not limited to the ADTF framework, but are a limitation of many simulation frameworks. Therefore, the proposed approach can be regarded as a general concept.

B. Extension to the Framework

Within the ADTF framework, it is possible to link several ADTF instances via the so called Messagebus protocol with the objective to distribute highly complex implementations on several workstations in order to improve real-time characteristics. The Messagebus protocol is a generic application layer protocol, accessing the network resources over a standard TCP/IP-socket interface. It can potentially be used for the integration of dedicated hardware blocks into the flow.

C. Requirements

The integration of dedicated hardware blocks is possible in three different positions, as shown in Fig. 1:

- algo_in: In this case the dedicated hardware block is positioned as first block in the data path and can provide the link to a camera or a player system,
- algo_out: In this case the dedicated hardware block is positioned as last block in the data path fand can provide the interface to a display or analysis system.
- algo_mid: In this case, the dedicated hardware block is positioned in the middle of the data path. As this case encompasses the two other cases, and represents the typical use case, it is the one to concentrate on.

For the maximum bandwidth requirement, a full-colour (RGB) image sensor with a sensitivity of 12bit, a resolution of 1024 x 512 and a frame rate of 43 frames / s is assumed as algo_in block. This calculates to a net data rate of around 820 Mbit / s, where the exact data rate depends on the control information, which is included into the data stream.
It is anticipated that all subsequent blocks work on images with less information, e.g. edge images, regions of interest, or alike, so that this number is regarded as maximum. A Gigabit Ethernet interface is therefore regarded as most suitable network connectivity, as it fulfils the bandwidth requirements and allows an easy use on the PC side. However, some additional precautions have to be taken to sustain the bandwidth on the hardware system.

III. The MAC Architecture

A. General Architecture

A Gigabit Ethernet MAC has been designed, which follow the legacy partitioning into three main parts, the MAC core, the PHY interface and the bus interface. These three parts can be integrated into existing systems as one instance of the MAC.

As shown in Fig.2 both the PHY and the bus interface are connected via FIFO interfaces, allowing easy replacement by implementations for different PHY, and bus interfaces, respectively.

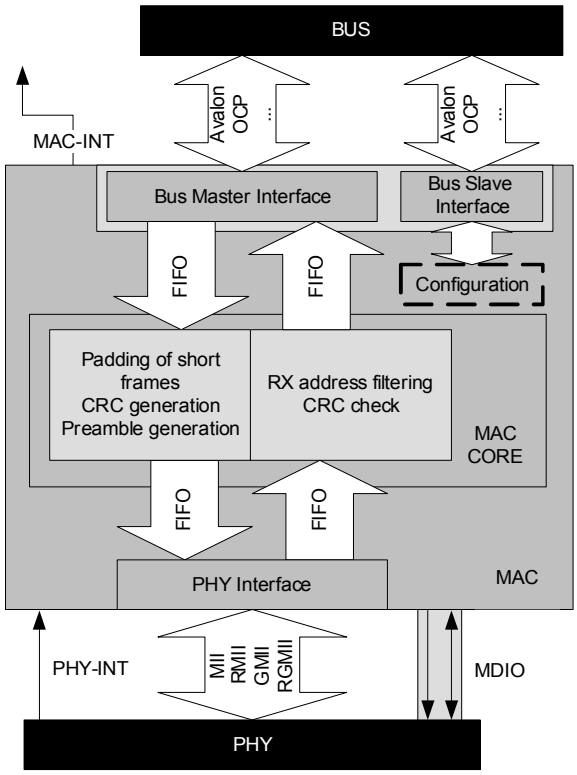

Fig.2 Block diagram of the Gigabit Ethernet MAC

Every single task, e.g. CRC checksum verification, address filtering, etc. is implemented as a single block, which brings the advantage that:

- Every single module is optional. E.g. if the VLAN tag support is not needed, it can just be omitted.
- Easy extensibility allows the integration of special function blocks. An example is the checksum calculation for higher-layer protocols.

The detailed block diagrams of the transmit and the receive paths are shown in Fig. 3 and Fig. 4, respectively.

B. Bus Interfaces

As the designed block shall be used flexibly in different environments, also the bus interfaces shall be interchangeable. I.e. the following bus interfaces shall be available in the first version:

- Avalon is the bus system used on Altera Nios II SoCs, called SOPC (System on programmable Chip). This interface is currently implemented conforming to version 1.3.
- AXI is one of five buses/interfaces defined in the Advanced Microcontroller Bus Architecture (AMBA) by ARM ltd. where AXI stands for Advanced eXtensible Interface. The version that shall be supported is AXI 2.0

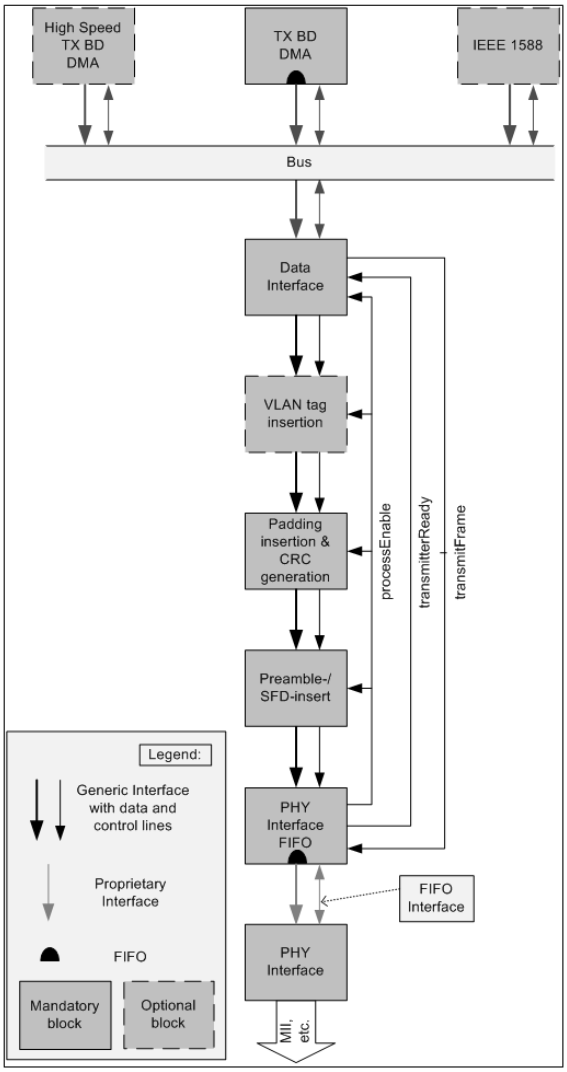

Fig. 3: Diagram of modular block architecture for the transmit data path

- OCP (Open Core Protocol) is a family of bus interfaces that is defined by the OCP International Partnership. It is not proprietary, openly licensable and an extensive and very comprehensive specification. The design will implement a sub-set of OCP version 3.0.

It is planned that all the three bus interfaces will be implemented as Single-Request-Multiple-Data Master and Single-Request-Single-Data Slaves, where both are supporting only In-Order requests and serving In-Order responses.

IV. Design, Test and Implementation

A. Design Principles

The complete design shall be usable both in hard-wired ASICS and programmable logic devices (PLD).

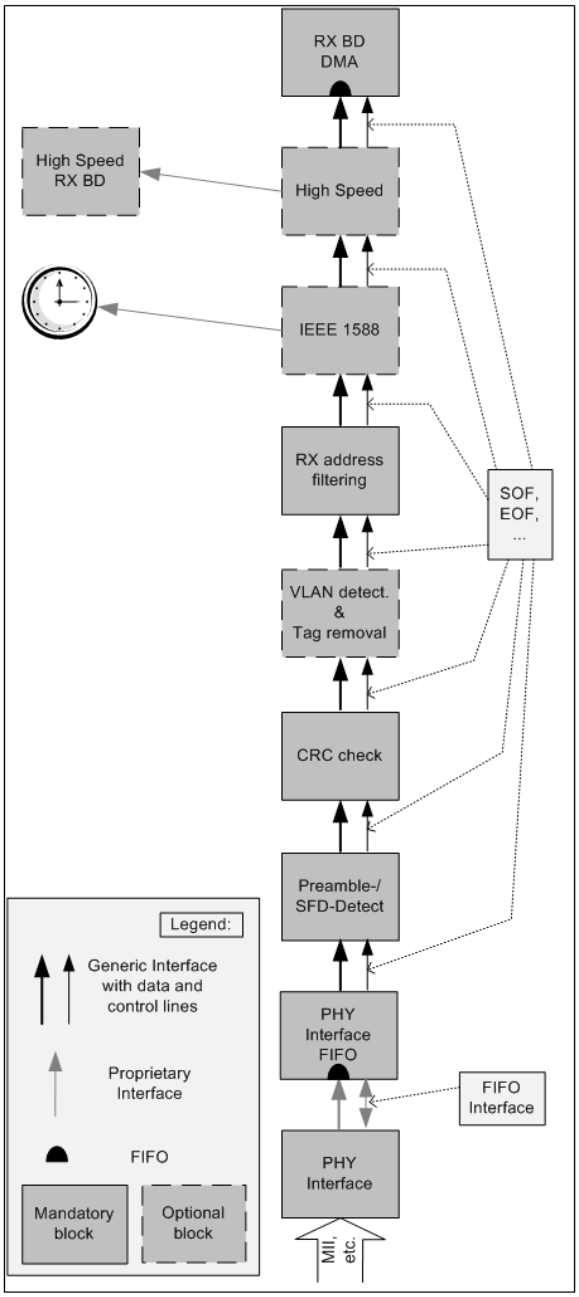

Fig. 4: Diagram of modular block architecture for the receive data path

It is therefore described in finite state machines using VHDL. In order to achieve good portability, the memory blocks are abstracted with a generic “wrapper”.

B. Test Environment

The verification of hardware components is always one of the most time-consuming tasks in the development process. As stated in [6], the full verification of the behaviour of a HDL model may consume as much as 60 to 80 % of the efforts.

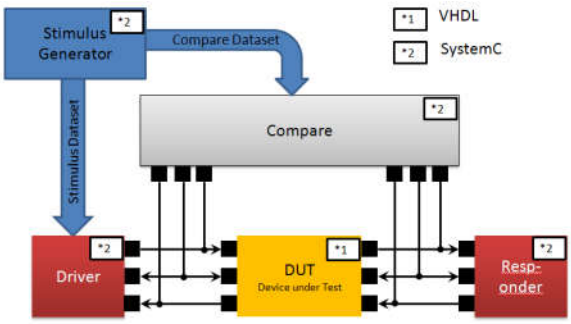

Fig. 5: Simulation Environment I - Module Tests

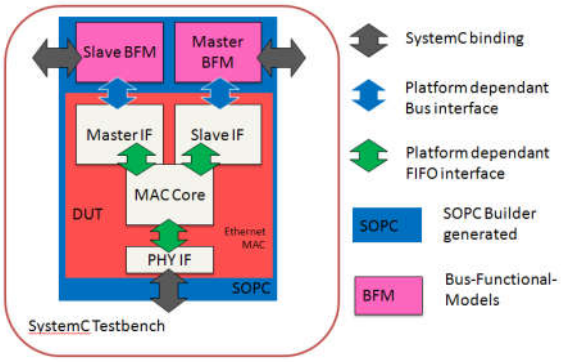

Fig. 6: Simulation Environment II - Full-System Simulation

There exist already several approaches like the "random constraints" included for example in the SystemVerilog language since version 3.1 to easily accomplish the task of full verification [7]. The missing experience with SystemVerilog and Verilog in the design team led to an implementation of a similar framework in SystemC for testing HDL models implementing packet based communication devices. This framework follows the UNH IOL[1] test cases, a testing and functional verification facility for Ethernet networking devices.

Mentor Graphics Questa was used as simulation engine, which allows cross-language simulation between VHDL, Verilog, SystemVerilog and SystemC models.

The simulation environment is shown in Fig. 4 for the module tests, and in Fig. 5 for the full system.

C. Implementation & Verification

The implementation followed a requirements engineering design approach and started with a detailed description of some 65 requirements.

In the second step the complete design and simulation environment was prepared, including the combined use of VHDL in the design itself, and SystemC for the simulation and verification environment.

The subsequent design phase started with the partition, and included the implementation and verification. As a result, around 10k lines of code (loc) for the VHDL design were generated. The verification environment includes around 2k loc of the SystemC framework (C++), and 12k loc for the simulation and verification (10k C++, 2k SystemVerilog).

Apart from the behavioural verification during the design phase, the network block was extensively tested in real networking environments.

D. Results

The design was first implemented in various Altera Cyclone II and III devices. In both devices, the MAC with simple FIFO Interface consumes ~1800 LUTs and ~1400 registers. The MAC with Avalon bus interface and statistical functions requires ~7100 LUTs and ~5100 registers. For verification, a NIOS host processor was used together with an embedded TCP/IP stack from the authors' institute.

V. Summary and Outlook

This project opened the opportunity to improve existing work-flows and to integrate new approaches. The implementation of a requirements engineering-based approach and a distributed development process was a challenge for every contributing engineer and will improve the flow of the future projects at the authors' institute. The design itself could be fruitfully used already also in further projects.

Acknowledgement

The presented work results from the joint project "Propedes" (Predictive Pedestrian Protection at Night), which is part of the project initiative eNOVA and is partially funded by the Federal Ministry of Education and Research (BMBF) under contract number 13N9754.

References

[1] D. Jansen (Ed.), "Handbuch der Electronic Design Automation", Fachbuchverlag Leipzig, 2001.

[2] A. Sikora, R. Drechsler, "Software-Engineering und Hardware-Design: Eine systematische Einführung", *Carl Hanser Verlag,* 2002.

[3] O. Gietelink, J. Ploeg, B. De Schutter, M. Verhaegen, "Development of advanced driver assistance systems with vehicle hardware-in-the-loop simulations", *Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility*, Vol. 44, Issue 7, 2006.

[4] D. Gerónimo, A.M. López, A.D. Sappa, T. Graf, "Survey of Pedestrian Detection for Advanced Driver Assistance Systems", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 2010.

[5] S. Jaeckel, A. Sikora, W. Rülling, "Implementierung eines Single Pass Connected Component Labeling Algorithmus zur Detektion von leuchtenden Objekten in Nachtszenen im Automotive Umfeld", *43. Workshop der Multiprojekt-Chip-Gruppe Baden-Württemberg*, Göppingen, 2010.

[1] UNH IOL = University of New Hampshire InterOperability Laboratory
http://www.iol.unh.edu/services/testing/

[6] J. Bergeron, “Writing Testbenches: Functional verification of HDL models”, Kluwer Academic Publishers, 2000.

[7] Accellera Organization Inc., “SystemVerilog 3.1 - Extensions to IEEE 1364-2001 Verilog”, 2003.

Steffen Jaeckel studied Computer Engineering at the University of Applied Sciences, Furtwangen and received his Dipl.-Inform. degree in April 2009. After that, he continued to work as project engineer at Steinbeis Innovation Center Embedded Design and Networking mainly in the field of embedded wireless and wired communication, embedded security and hardware design for networking protocols.

Alexander Rohleder studied Information Technology at the Baden-Wuerttemberg Cooperative State University Loerrach and received his B.Eng. degree in Sep. 2010. His bachelor thesis was on the protocol implementation of Messagebus protocol in SW and HW. Since then, he works as project engineer at Steinbeis Innovation Center Embedded Design and Networking mainly in the field of embedded wireless communication.

Axel Sikora was awarded the Dipl.-Ing. in electrical engineering in 1993 and the Dipl. Wirtsch.-Ing. in 1995 from RWTH Aachen. He earned his Ph.D. at Fraunhofer Institute IMS at Gerhard-Mercator-University Duisburg. Since 1999, he is Professor at the Baden-Wuert¬temberg Cooperative State University Loerrach, where he leads the Steinbeis Innovation Center Embedded Design and Networking.