Development Platform for Safe and Efficient Drive

Standard interfaces definition – 2nd Release

Deliverable n.	D25.4 – Standard interfaces definition (Second Release)		
Sub Project	SP2	ADAS development platform	
Workpackage	WP2.5	Platform System Architecture	
Tasks	T2.5.3	Standard interfaces	
Authors	Frank Badstübner Ralf Ködel Wilhelm Maurer Martin Kunert Joshué Pérez David Daurenjou F. Giesemann, G. Paya Vaya, H. Blume	Infineon Technologies AG Robert Bosch GmbH INRIA Continental IMS	
File name	D25.4_Standard_interfaces_definition_-_Second_Release_IFAG_Final		
Status	Final, v0.5		
Distribution	Restricted (RE)		
Issue date	31/08/2014	Creation date	05/10/2014
Project start and duration	1st of September, 2012 – 36 months		

REVISION CHART AND HISTORY LOG

Ver	DATE	AUTHOR	REASON
0.1	2014-05-28	Frank Badstübner	Table of contents and document structure created, content of first version included
0.2	2014-07-18	Pier Claudio Antonello	Added trends on CAN and Ethernet interfaces
0.3	2014-09-03	Frank Badstübner	IMS contribution included; consolidated version generated
0.4	2014-09-24	Frank Badstübner	Final version generated
0.5	2014-10-05	Frank Badstübner	Reviewer comments and recommendations integrated. Final version generated for sub-mission.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

None

LIST OF ABBREVIATIONS

ABBREVIATION	DESCRIPTION
ADAS	Advanced Driver Assistance System
ADC	Analog-to-Digital Converter
ADTF	Automotive Data and Time-Triggered Framework
ASAM	Association for Standardisation of Automation and Measuring Systems (ASAM e.V.)
AUTOSAR	AUTomotive Open System ARchitecture
BBB	Backbone bus
CAN	Controller Area Network
CIF	Camera Interface (please note the difference to the Common Intermediate Format, introduced by ITU standard H.261)
CMOS	Complementary metal-oxide-semiconductor
DC	Direct Current
ECL	Emitter Coupled Logic
ECU	Electronic Control Unit
EMI	Electro-Magnetic Incompatibility
FPGA	Field Programmable Gate Array

HW	Hardware
ISO	International Organization for Standardization
JPEG	Joint Photographic Experts Group
iJPEG	Independent JPEG Group
LIN	Local Interconnect Network
LVDS	Low Voltage Differential Signaling
MPEG	Moving Picture Experts Group
MOST	Media-Oriented Systems Transport
OSI	Open Systems Interconnection (Model)
PCI	Peripheral Component Interconnect
RF	Radio Frequency
RIF	(Serial) RADAR Interface
RTPGE	Reduced Twisted Pair Gigabit Ethernet
SAE	Society of Automotive Engineers
SCI	Scalable Coherent Interface
SW	Software
TCP / IP	Transmission Control Protocol / Internet Protocol
UDP	User Datagram Protocol

USB	Universal Serial Bus
UTP	Unshielded Twisted Pair
VAN	Vehicle Area Network

EXECUTIVE SUMMARY

The basic idea and intention of the DESERVE project is to standardize the interfaces between the three different development levels as far as possible:

- Level 1: PC-based development framework
- Level 2: Embedded controller development framework
- Level 3: Dedicated hardware (e.g. ASIC / accelerator) development framework

To follow this idea, the importance of well-defined and standardized interfaces between the particular components and modules is striking. Special interface hardware boxes for connecting proprietary (sensor) interfaces to already standardized communication protocols are needed to ensure high data rate, low latency communication.

In order to be able to integrate existing modules – mostly communicating utilizing today's automotive bus systems like LIN, CAN, FlexRay – DESERVE needs to consider these bus systems. However, as these are by far not sufficient for a transfer of raw data (e.g. radar signal sample data or image pixel information) additional advanced (so called next generation interfaces) are needed for the DESERVE platform as well to ensure communication between sensors, the above mentioned 3 levels of processing platforms and the actuators.

Consequently, the deliverable starts with a review of existing ADAS interfaces (including interface standards and communication protocols). Chapter 3 presents the interface architecture, while chapter 4 defines next generation interfaces, e.g. parallel camera interface (CIF) and serial radar interface (RIF). Suitable interface candidates are LVDS and Gbit-Ethernet. The latter ist proposed as generic interface for the communication between ADTF project (level 1) and FPGA based hardware (level 3).

It is to be noted that several functions of the embedded application software (regardless on which platform it is running) can be calibrated (or tuned) for similar applications without the need to change the embedded application software itself. This step is a very essential one, as it allows the desired reuse of "standardized software functions".

1. Introduction

1.1. Objective and scope of this document

The objective of this deliverable is to describe the second release of the standardized interfaces to be implemented into the platform system architecture. D25.4 is the second of a series of deliverables documenting the work performed to define the second release of the platform system architecture:

- D25.2 Platform system architecture – 2nd release
- D25.4 Standard interfaces definition – 2nd release
- D25.6 Guidelines for applications – 2nd release and
- D25.8 DESERVE platform – 2nd release

1.2. Structure of the deliverable

D25.4 is dedicated to the definition of the standardized interfaces in the DESERVE platform. Work performed refers to task 2.5.3 of the DESERVE project. This is a very critical task of the overall DESERVE platform development. Several of the innovative and beneficial aspects of the DESERVE platform depend on the definition of standard interfaces between sensors, control units and actuators. This task identifies general rules to be satisfied by these interfaces and will then, at least for the most diffused and important sensors/actuators/control modules, define the standard interface.

2. Review of existing ADAS interfaces

2.1. Communication protocols in existing ADAS

All electronic embedded systems used to control vehicle functions (specifically ADAS) need communications networks and protocols to manage all the process information. The modules receive input information from a network of sensors (e.g. for engine speed, lasers, cameras, etc.) and send commands to the control stage (Application platform in DESERVE), and finally to the actuators or warning systems that execute the commands (IWI platform) [2].

Due to the increasing complexity of modern ADAS applications, point-to-point wiring has been replaced by multiple networks and communications protocols. These protocols use different physical media to provide safe connection among components on the vehicle. These include single wires, twisted wire pairs, optical fiber cables, and communication over the vehicle's power lines. Some of the automotive manufacturers are seeking a standard protocol. It is not part of the objectives of DESERVE to provide a universal solution on this low level communication, because four different platforms will be used. However, a definition and developing of standard HW/SW interfaces, which allows a large reusability of SW and HW components, will be provided for communication between functions and modules developed in the project.

Some of the most known and used communication protocols and standards used in nowadays vehicles are explained [1]:

- **CAN (controller area network):** it is a cheap (and most used) low-speed multi-master broadcase serial bus standard (developed by Bosch). The CAN bus is designed to allow electronic control units (ECUs), microcontrollers and devices to communicate with each other within a vehicle without a host computer. It's a message-based protocol, specifically designed for automotive applications but

now also used in other areas such as aerospace, industrial automation and medical equipment.

- **VAN (vehicle area network):** it is similar to CAN but not widely used. It was developed by PSA Peugeot Citroën and Renault.
- **FlexRay:** it is a protocol used for general-purpose. It has high-speed protocol to support time triggered architecture and fault-tolerance property.
- **LIN (local interconnect network):** it is a low-cost in-vehicle sub-network, used mainly by some European vehicle manufacturers. Some features are: Guaranteed latency times, variable length of data frame (2, 4 and 8 byte), configuration flexibility, etc.
- **SAE-J1939 and ISO 11783:** an adaptation of CAN for agricultural and commercial vehicles. It is also used for diagnostics among vehicle components.
- **MOST (Media-Oriented Systems Transport):** it is a high-speed multimedia that supports user applications such as GPS, radios, and video players. It can be used for multimedia applications inside or outside the car.
- **Keyword Protocol 2000 (KWP2000):** it is a protocol for automotive diagnostic devices. It can run in serial line or over CAN.

Other protocols used are: DC-BUS, IDB-1394, SMARTwireX, SAE-J1850, SAE-J1708, SAE-J1587 and ISO-9141-I/-II.

Recent vehicles have installed multiple networks (with different protocols) to communicate among electronic control units (ECU) onboard. The networks are isolated from one another for several reasons, including bandwidth and integration concerns.

2.2. Review of existing interface standards

Current ADAS systems (Figure 1, for example, presents an overview of the Continental ADAS system) are designed and built to provide a dedicated answer to specific functionalities as: Lane Departure Warning, Pedestrian Detection, Blind Spot Detection, Emergency Braking,... .

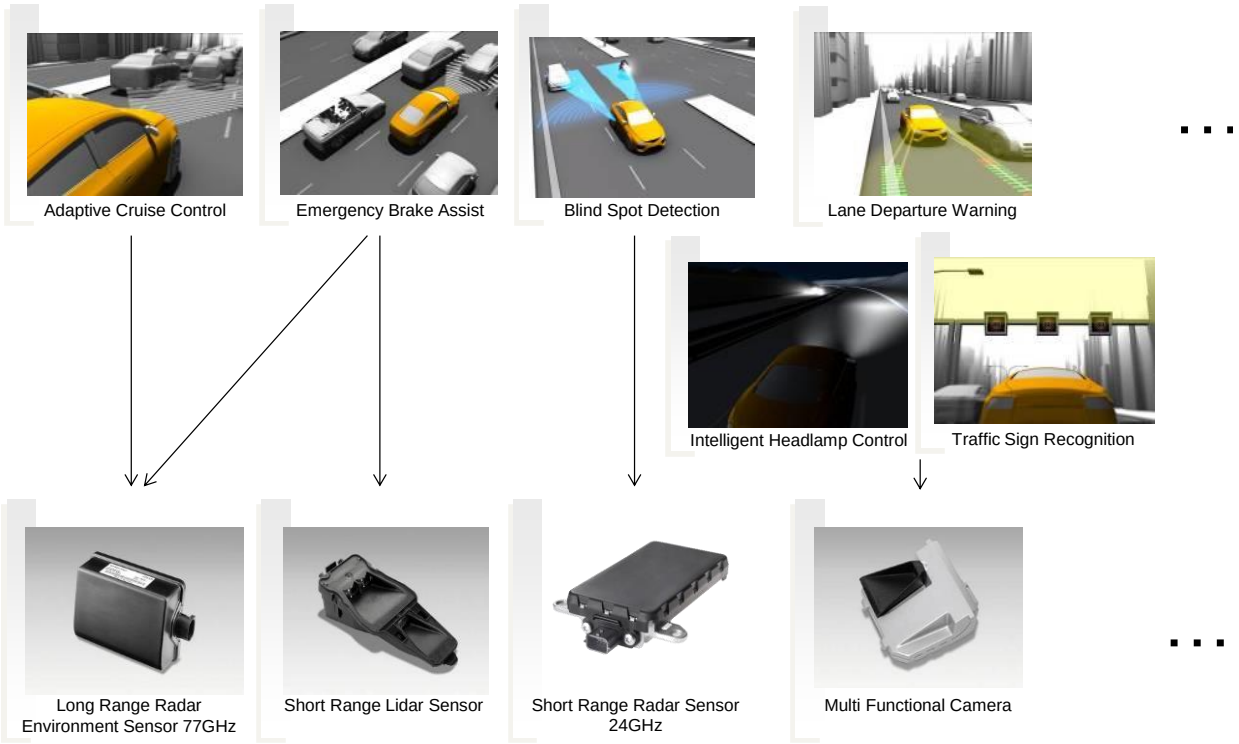

Figure 1: Continental ADAS system

Most ADAS are including in the same box the sensor itself and the processing unit (this is typical for Continental products, for example). So, the raw data provided by the sensor (camera, radar) are directly loaded inside the ECU unit and processed. Only high level (processed) information is available on the communication buses. Raw data (e.g. pixel information of images) is not available.

The ADAS modules are dedicated products which communicate mainly within the same hardware unit. Nevertheless, to adjust the algorithms in function of the vehicle status, it's necessary to provide the ADAS modules with some vehicle information as: speed, yaw rate, direction indicator status...

To manage the vehicle information acquisition and sending of the outputs, various communication interfaces are available, depending on the product:

- CAN communication interface
- FlexRay communication interface

The CAN Bus interface uses an asynchronous transmission scheme controlled by start and stop bits at the beginning and the end of each character. This interface is used, employing serial binary interchange. Information is passed from transmitters to receivers in a data frame. Different data rates are defined and the cable length depends on the data rate used. The maximum data rate is 1Mbps (cable length max: 40 m), the lower rate is 10kbps (cable length max: 1 km)

Figure 2: CAN Message Frame

FlexRay is also an automotive network communications protocol; it is designed to provide high–speed deterministic distributed control for advanced automotive applications, to govern on-board automotive computing. It is faster and more reliable compared to CAN, but also more expensive (see comparison of LIN, CAN, FlexRay in Figure 3).

FlexRay's dual channel architecture offers system–wide redundancy that supports the reliability requirements of safety related systems. With a data rate of 10 Mbit/s per channel, the FlexRay system can also be employed as a vehicle–wide network backbone, working in conjunction with other communication systems like CAN and LIN. It can be used to reduce the number of parallel CAN networks used to solve bandwidth bottlenecks.

Bus	**LIN**	**CAN**	**FlexRay**
Max Speed	40 kbit/s	1 Mbit/s	10 Mbit/s
Cost	€	€€	€€€
Wires	1	2	2 or 4
Typical Applications	Body Electronics (Mirrors, Power Seats, Accessories)	Powertrain (Engine, Transmission, ABS)	High-Performance Powertrain, Safety (Adaptive cruise control, Drive-by-wire, Active suspension, Pedestrian Detection)

Figure 3: Bandwidth comparison

Nevertheless, during the development step even the FlexRay bandwidths aren't sufficient. It is necessary to record a lot of data useful to develop, improve, test and to validate for instance ADAS functionalities: the sensor raw data (image of the camera), vehicle information data, some intermediate data (optical flow, disparity map, ...) and of course the results data. All those data must be synchronized and recorded in real time, the bandwidth is really more important than the FlexRay capabilities.

For this reason, specific electronics (based on the Ethernet communication) and software (similar as ADTF, RTMaps ...) have been developed and integrated to record, in real time, these data. On some products up to 4 Ethernet connections are integrated to transfer the data until the recorder unit (dedicated PC). As an example, Figure 4 shows the Continental ADAS architecture.

The communication bandwidth requirements increase more and more with more and more complex applications, the existing network are not specified to cover the increasing demands for bandwidth, and the Ethernet price. The Ethernet seems to be an alternative to the existing communication hardware.

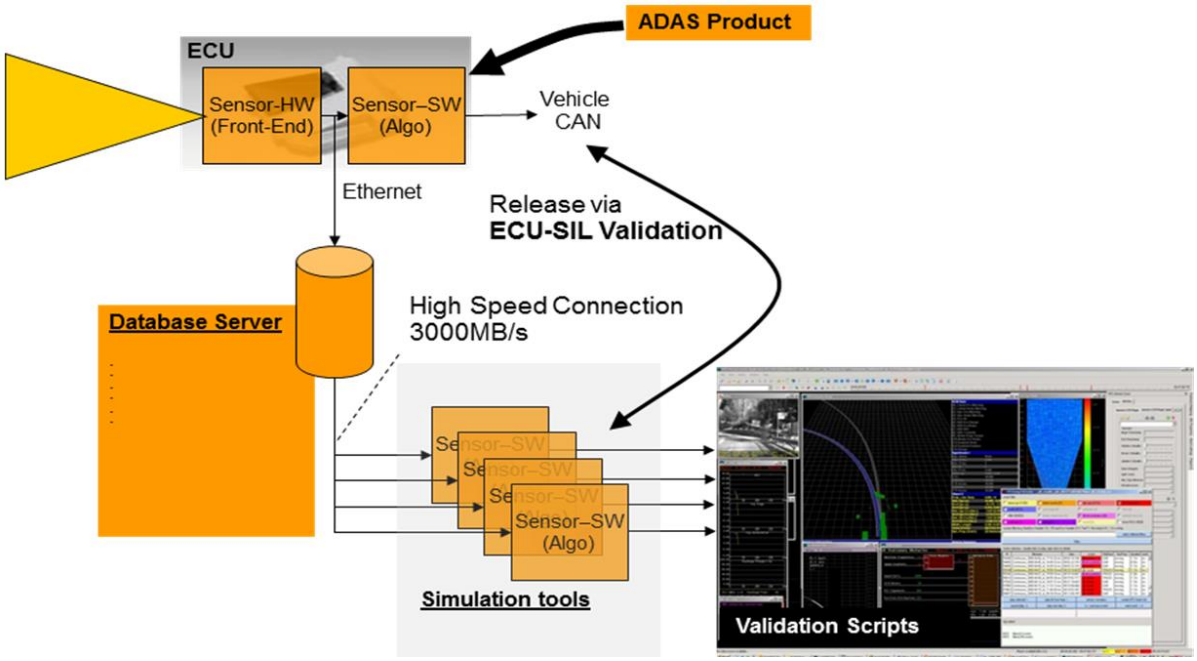

Figure 4: Continental ADAS architecture

2.3. Trends for future ADAS interfaces

The current CAN specification is the ISO 11898. It is named CAN 2.0B with 11- 29 bit message identifier and typical bitrate of 500Kbps or lower (could be used at up to 1Mbps).

A new specification, known as CAN-FD, has been submitted to ISO for extension of ISO 11898. The main differences to CAN 2.0 are the extended user data from 8 up to 64 bytes, and the ability to send user data with higher data rates (up to 2-5 Mbps in accordance to network topology). In this way, the requirements for higher bandwidth networks in the automotive industry are fulfilled, while profiting from the experiences in CAN development.

CAN FD shares the physical layer, with the CAN protocol as defined in the CAN Specification 2.0. The frame format, however, is different. There are two new control bits in the

CAN FD frame, the first enabling the new frame format with different data length coding and the second optionally switching to a faster bit rate after the arbitration is decided. New CRC polynomials are introduced to secure the longer CAN FD frames with the same Hamming distance as in the proven CAN protocol.

The introduction of CAN FD for busload critical situations seems to be in near future the solution to achieve higher network capability without large changes in the existing systems.

However, the more promising communication interface candidate to replace other existing protocols (MOST, LVDS) and alternative to FlexRay, seems to be Automotive Ethernet.

This is not yet a mature standard, but what is going to change from Standard Ethernet is the physical layer only.

The physical layer will be adapted to automotive requirements. The other communication layers will still be Standard Ethernet, with all advantage of larger market technology.

Currently, this technology offers, by means of low cost single or double unshielded twisted pair (UTP) cables, 100Mbps full duplex of bandwidth; the roadmap of the technology foresee speeds up to 1 Gbps or higher on single UTP cable (Reduced Twisted Pair Gigabit Ethernet -RTPGE).

The protocols over physical layer offer many opportunities to future development; the more relevant one is that Ethernet can manage different protocols over the same network: for example UDP is connectionless (best for loss-less connections) and TCP connection oriented (with additional overhead).

But over a unified vehicle network would be possible to have at the same time deterministic communication (with defined latency) and best effort traffic.

Other interesting features, that probably will be available, are Power over Ethernet (possibility to supply energy on the communication cable) and Partial Networking (reducing power consumption switching-off ECUs or sensors when unused).

Automotive Ethernet is the ideal solution for high demanding vehicle networks. Expected application domains are: ADAS, Infotainment and Diagnostics.

3. Definition of the interface architecture

The definition of the interface architecture plays a central role for the communication and data exchange between the different DESERVE platform modules and sensor components. In the DESERVE deliverable D2.2.1 [4] the abstracted interface descriptors are already defined on a content-based hierarchical level. With standardized information data flow between the numerous platform modules both the development time and the extension in performance and scope of the encapsulated modules can be realized very efficiently and in a well-structured way.

The architecture of the interface has to be defined individually for each of the existing OSI layers, starting from the physical layer up to the application layer. Taking the OSI model for computer networks as starting point, the following DESERVE layers can be defined:

1) Physical data layer
   In this layer the physical structure of the sensor component or communication unit is concerned. Often proprietary data structures for connecting the sensors or actuators exist and need to be taken into consideration. Depending on whether raw signal streams will be transferred or already condensed and preprocessed data, the transfer rate may vary from a few hundred bits up to several Gigabits per second. A back-channel for sensor calibration and setup is almost always required and has to be foreseen.

2) Communication layer
   The DESERVE communication layer includes the data link, network and transport layer and contains all the necessary information on how the data will be sent over the link between the platform modules and components. Package based protocols like TCP/IP or UDP are the favorite protocol types to be used for the communication. However, also proprietrary or recently developed bus standards like Thunderbolt (PCI-Express based active interface link) or the USB-3.0 serial bus may be used in addition to the Ethernet bus protocol.

3) Application service layer

This is the highest level that establishes the abstracted data link to the user applications. Similar to the TCP/IP protocol the session layer (responsible for synchronization and logical port mapping) and the presentation layer (syntax layer for data conversion and encryption) are merged (or better omitted, if not needed) in the application layer. Bi-directional point-to-point connections between the upper-level modules and the sensor units occur with a data flow and format as depicted in Figure 5.

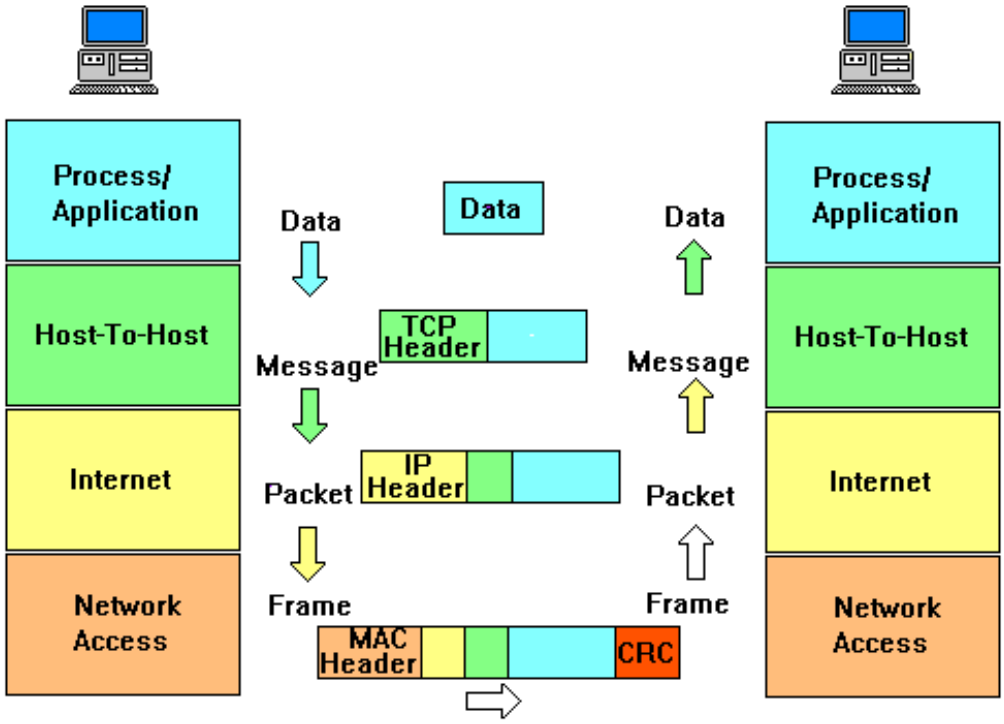

Figure 5: Data flow for a peer-to-peer connection over a physical link

For modules that only communicate within the same hardware unit the physical data and communication layer are no longer needed. Instead, a message box oriented data trans-

fer link is proposed for usage in the DESERVE project. The data to be transmitted is written in a predefined message box descriptor field and message flags trigger the synchronization and data updates in the concerned modules. The message box principle is sketched in Figure 6.

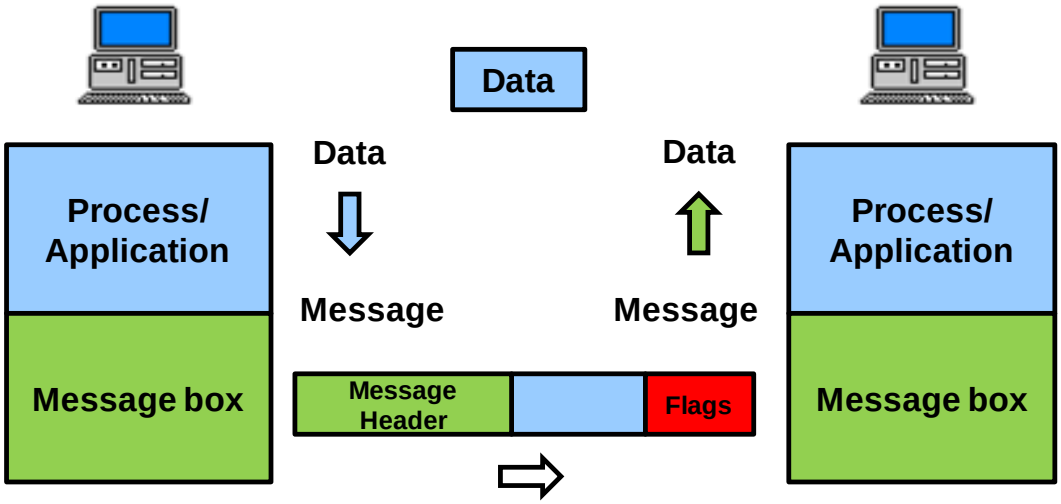

Figure 6: Message box principle for intra-unit communication

Finally, the interfacing concept of the AUTOSAR standard is considered and incorporated in the DESERVE platform where useful and appropriate. The AUTOSAR mode of operation, as depicted in Figure 7, fits already quite well with the general DESERVE approach proposed in this document.

Note: Being a research project, the development work conducted in DESERVE is discharged from being fully compliant with the AUTOSAR standard. Where possible and easy to implement, inputs from AUTOSAR will be considered, of course. A mandatory request for AUTOSAR compliance is, however, not up for discussion.

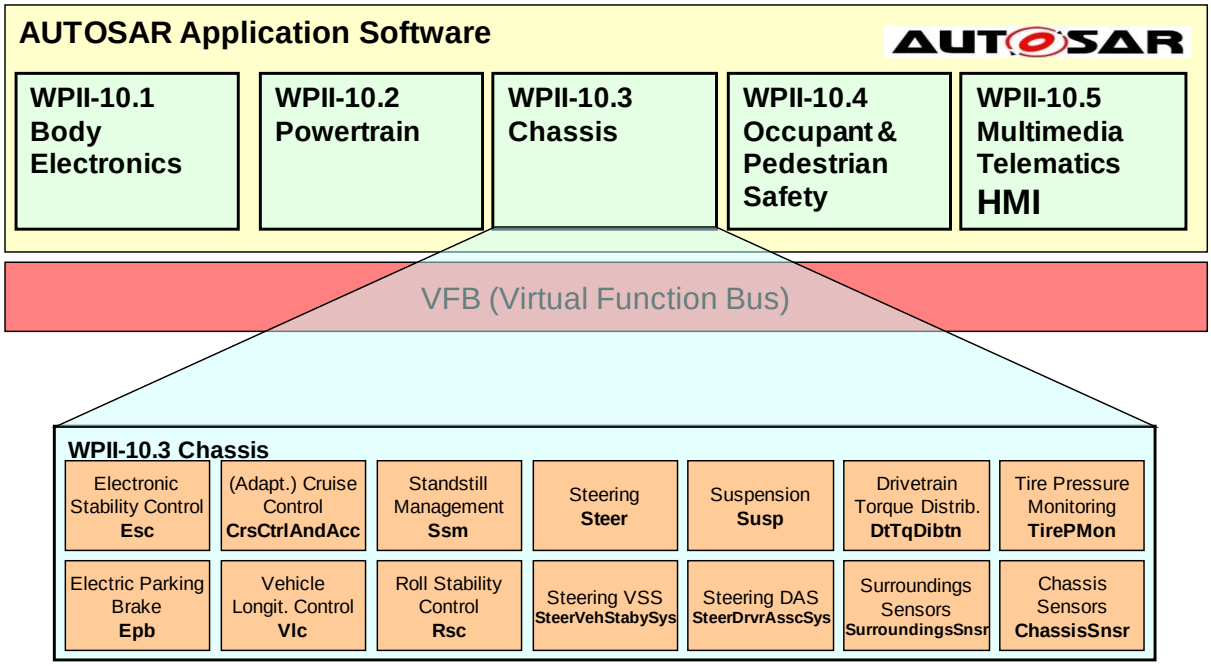

Figure 7: AUTOSAR Application software concept

In order to achieve a good reusability of embedded software functions, it has proven to be efficient in the industry to separate the "function software" from parameters defining the behavior of the software (= calibration data). This allows generating embedded systems with generic software functionalities by "embedded systems suppliers" (e.g. Continental, Bosch or others). Such systems are bought by OEMs for building their ADAS systems. The OEM can adapt the generic function to the individual behavior significant for his customers "just by calibration". In this process via an application system (market leader is INCA for example), the calibration data can be changed while the embedded system is running - regardless if simulated on a PC or running already on the target hardware.

In engine control units, for instance, it is used the ASAM standard ASAP2 [5] in order to describe the calibration parameters. This allows the access of the application system to access and to change such defined calibration parameters while the software is running in order to reduce the adaptation work to different hardware or application tasks (e.g.:

other camera, other vehicle, other driver demands) just by changing the input maps, input curves or values of the function software. This split of function development work (e.g. done by BOSCH) and calibration work (e.g. done by the OEM) has proven to be highly efficient also for other embedded mechatronic systems (e.g. for transmission control units and meanwhile also for active suspension control units) in order to reduce software versions or releases and to reuse function code.

The separation of calibration date and function software is also allowed according to the AUTOSAR concept, but not really specified there. Within DESERVE the separation shall be foreseen in the software design following the ASAM ASAP 2 standard.

4. Definition of next generation interfaces

4.1. Next generation high speed sensor interfaces

The definition of next generation high speed sensor interfaces is the key to enable the improvement for next generation driver assistant systems. An optimized interface leads to optimized dataflow and system performance. For each sensor family (Camera/RADAR) there is a dedicated interfacing needed.

4.1.1. Parallel camera interface (CIF)

The Camera Interface (CIF) represents a complete video and still picture input interface transferring data from an image sensor into video memory. Furthermore, several hard-ware blocks - performing image processing operations on the incoming data - are provided.

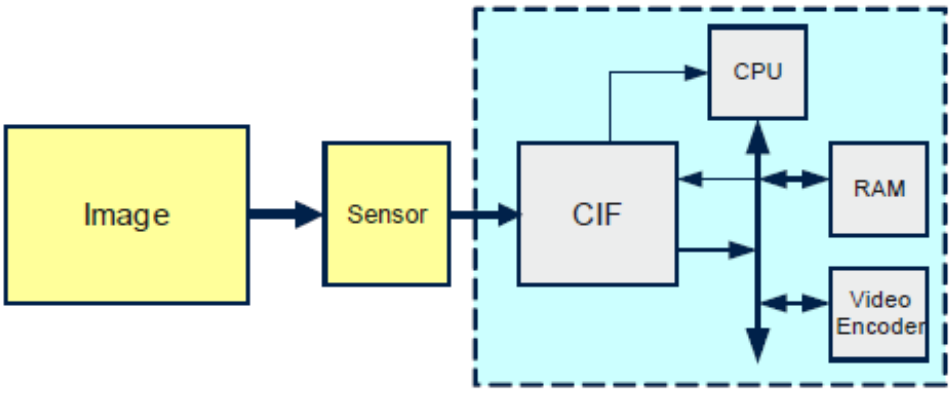

Figure 8: Camera Interface (CIF) overview

Camera Interface Functional Overview

Apart from providing the physical interfacing to various types of camera sensor modules, the CIF block implements image processing and encoding functionalities. The integrated image processing unit supports image sensors with integrated YCbCr processing. Additionally, the CIF also supports the transfer of RAW (e.g. Bayer Pattern) images and non-frame synchronized data packets. The CIF block features a 16 bit parallel interface. All output data are transmitted via the memory interface to a BBB (Back Bone Bus) system using the master interface. Programming of the CIF is done by register read/write transactions using a BBB slave interface.

The CIF provides a sensor/camera interface for a wide variety of video applications and it is optimized for high speed data transmission under terms of low power consumption. This module is designed to be used for the following use cases:

- Video capturing/encoding
- Still image capturing in YCbCr with on-the-fly JPEG encoding
- RAW frame data capturing

The CIF requires fast system memory for image storage in either planar, semi-planar or interleaved YCbCr or RAW planar format or as JPEG compressed data. The iJPEG encoding engine should be able to generate a full JFIF 1.02 compliant JPEG file that can be displayed directly by any image viewer. All important YCbCr formats - which are used for video compression (e.g. MPEG4) for instance - are supported. For on-the-fly encoding macro block line interrupts are generated to trigger video encoding. The following list summarizes the targeted CIF's features:

- Throughput up to 96 MPixel/s
- 32-bit Back Bone Bus (BBB) slave programming interface
- BBB master interface
- ITU-R BT 601 compliant video interface supporting YCbCr and RAW data transfer
- ITU-R BT 656 compliant video interface supporting YCbCr and RAW data transfer
- Non line/frame aligned data transfer (data mode)
- 16-bit parallel camera interface

- YCbCr 4:2:2 processing
- Hardware JPEG encoder (supporting images up to a horizontal resolution of 1280 pixel) incl. JFIF1.02 stream generator and programmable quantization and Huffman tables
- Windowing and frame synchronization
- 1 Main and 5 Extra Image Cropping units allowing parallel transfer and position adjustment of up to 6 sub regions
- Path from Main Image Stabilization or from one Extra Image Cropping unit to debug interface including a Metasymbol Generation unit inserting frame start and timing information into the stream
- Programmable watchdog timer to detect abortion/breaks in frame transmission
- Linear downscaling for the first extra path, supporting a mode for RGB Bayer Pattern
- Frame skip support for video (e.g. MPEG-4) encoding
- Macro block line, frame end, capture error, data loss interrupts and synchronization (h_start, v_start) interrupts
- Programmable polarity for synchronization signals
- Luminance/chrominance and chrominance blue/red swapping for YCbCr input signals
- Maximum input resolution of 4095x4095 pixels
- Buffer in the video memory organized as ring-buffer, supporting up to 2x8 Beat Bursts (2x32 Bytes)
- Buffer overflow protection for RAW data transfer and JPEG files
- Asynchronous reset input, software reset for the entire IP and separate software resets for all sub-modules
- Support of planar, semi planar and interleaved storage format (main path)
- Power management by software controlled clock disabling of currently not needed sub-modules

4.1.2. Serial RADAR interface (RIF)

Analog-to-digital converter (ADC) sample rates have been increasing steadily for years to accommodate newer bandwidth-hungry applications in communication, instrumentation, and consumer markets. Coupled with the need to digitize signals early in the signal chain to take advantage of digital signal processing techniques, this has motivated the development of high-speed ADC cores that can digitize at clock rates higher than 100 MHz to 200 MHz with 8 to 12 bit resolution.

In standalone converters, the ADC needs to be able to drive receiving logic and accompanying PCB trace capacitance. Current switching transients due to driving the load can couple back to the ADC analog front end, adversely affecting performance. One approach to minimize this effect has been to provide the output data at one-half the clock rate by multiplexing two output ports, reducing required edge rates, and increasing available settling time between switching instants.

Use of LVDS for ADC high speed data output

A new approach to providing high-speed data outputs while minimizing performance limitations in ADC applications is the use of LVDS (low voltage differential signaling). Infineon is incorporating LVDS output capability in new RF devices ADCs—and will include LVDS input capability in its new micro-controller designs.

LVDS is, as the name says, a low voltage differential signaling scheme. The operative words here are low voltage (~350 mV) and differential. Standards bodies have developed specifications that will be discussed later in this note. Lower voltage signal swings have the intrinsic advantage of shorter switching times as well as reduced EMI concerns (adjacent differential traces tend to cancel each other's EMI).

Figure 9: LVDS Output Levels

Differential signaling also has the well-known common-mode rejection benefit. Noise that is coupled to the signals tends to be common to both signal paths and cancelled out by a well-designed differential receiver. LVDS outputs are current output stages requiring a 100 Ohm terminating resistor at the receiver, differing from CMOS outputs that generally do not require termination. The current output results in a fixed dc load current on the output supplies—avoiding current spikes on the supply that can couple to the sensitive analog front end.

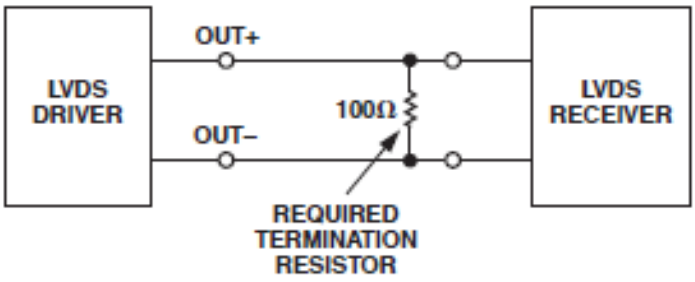

Figure 10: Output termination in LVDS connections

Standards

Two standards have been written to define LVDS. One is the ANSI/TIA/EIA-644 which is titled, ”Electrical Characteristics of Low Voltage Differential Signaling (LVDS) Interface Circuits.” The other is IEEE Standard 1596.3 which is titled, ”IEEE Standard for Low-

Voltage Differential Signals (LVDS) for Scalable Coherent Interface (SCI).” A brief summary of these two standards is provided below.

a) ANSI/TIA/EIA-644

   The ANSI/TIA/EIA Standard was developed under the Telecommunications Industry Association (TIA) Subcommittee TR-30.2 and contains only generic electrical specifications for LVDS. Its purpose was to create a general high-speed interface standard for use in point-to-point connections between data communications equipment. The maximum data signaling rate is 655Mbps. The TIA Subcommittee intended other standards bodies to reference ANSI/TIA/EIA-644 in more complete interface specifications between transmitters and receivers.

b) IEEE Standard 1596.3

   The IEEE Standard 1596.3 was developed as an extension to the 1992 SCI protocol (IEEE Standard 1596-1992). The original SCI protocol was suitable for high-speed packet transmissions in high-end computing and used ECL levels. However, for low-end and power-sensitive applications, a new standard was needed. LVDS signals were chosen because the voltage swing is smaller than that of ECL outputs, allowing for lower power supplies in power-sensitive designs.

4.2. Generic interface to communicate between ADTF project and FPGA based hardware platform

In order to allow an easy and standard communication between an ADTF-Project and the FPGA-based hardware platform, a generic interface is used. The generic interface realizes the communication with different processing elements implemented in the FPGA-based hardware platform transparent to the user. The general communication method is shown in Figure 11.

As a proposal for the link between ADTF and the FPGA-based hardware platform, a Giga-bit Ethernet interface can be used.

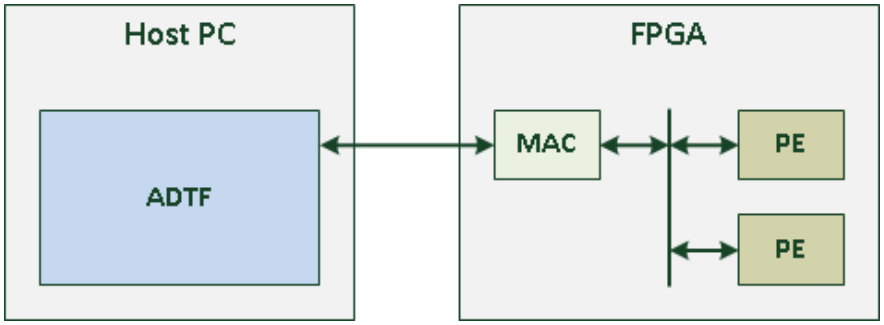

Figure 11: Communication between ADTF on host pc and processing elements implemented on the FPGA-based hardware platform

An exemplary implementation of a Gigabit Ethernet connection between ADTF and an FPGA development board was created and evaluated [3]. The communication scheme, which is depicted in Figure 12, is very similar to the structure shown in Figure 11. A host PC is running the ADTF software and serving as an interface to different sensors. It sends data via an Ethernet link to the FPGA board. Data can be written to the processing elements (PE) directly. Additionally, bigger amounts of data can be written to the FPGA board's memory.

This implementation has been evaluated with a case study (see [3] for details) and it has been shown, that the simulation time of a complex application can be reduced by a factor of 65 in this application. Therefore the algorithmic evaluation and parameter space exploration is accelerated significantly.

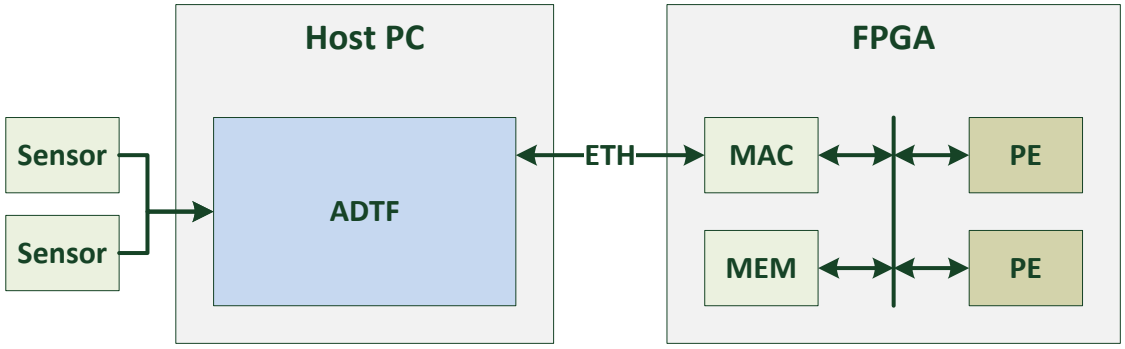

Figure 12: Implemented Gigabit Ethernet Interface between ADTF and an FPGA development board

5. Conclusion

A first proposal for the interfaces in the DESERVE platform was described in this deliverable. It integrates existing automotive bus systems and proposes additional advanced bus systems, namely LVDS (for communication with sensors, in particular the transfer of raw data) and Gbit-Ethernet (for the high speed, low latency communication between the three levels of DESERVE architecture).

It is concluded that the communication concept is well suited to cover all needs of future ADAS systems. Consequently, DESERVE work can continue as planned.

REFERENCES

[1]. Transportation Research Board, special report 308: the safety Promise and challenge of Automotive electronics, Insights from unintended acceleration, National Research Council of the national academies, 2012.

[2]. DESERVE Deliverable D12.1 - Development Platform Requirements, SP1 Requirements and Specifications.

[3]. F. Giesemann, G. Paya-Vaya, H. Blume, M. Limmer, W. Ritter. A Comprehensive ASIC/FPGA Prototyping Environment to Explore Embedded Processing Systems for Advanced Driver Assistance Applications. International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS), 2014

[4]. DESERVE Deliverable D2.2.1 – Perception layer Preliminary Release

[5]. ASAP2 file: http://www.asam.net/