

A Flexible High-Performance Accelerator Platform for Automotive Sensor Applications

Henning Sahlbach, Sean Whitty and Rolf Ernst
Technische Universität Braunschweig

ABSTRACT

High-performance computer architectures for advanced driver assistance systems have become increasingly important in automotive research in the last several years. In order to achieve an optimal and robust perception of the vehicle's surroundings, current driver assistance applications typically rely on multiple sensor systems that deliver large amounts of incoming data from different sensor types. Such sensors include optical systems, which consist of a multi-camera setup combined with complex preprocessing algorithms. These algorithms exhibit high computation and data transport demands, as real-time image processing of multiple input streams is a mandatory requirement for these systems. At the same time, however, future driver assistance systems must adhere to strict power consumption requirements and automotive cost constraints in order to be considered for integration in series vehicles.

This paper addresses these power and cost problems and presents an FPGA-based high-performance computing platform combined with a flexible, weakly-programmable data flow architecture and an associated high-level prototyping framework, which targets an efficient acceleration of computation-intensive tasks in driver assistance applications. The usability and processing performance of the platform is demonstrated by an advanced Motion Estimation application, which represents a challenging preprocessing step in automotive image processing.

CITATION: Sahlbach, H., Whitty, S. and Ernst, R., "A Flexible High-Performance Accelerator Platform for Automotive Sensor Applications," *SAE Int. J. Passeng. Cars - Electron. Electr. Syst.* 5(1):2012, doi:10.4271/2012-01-0939.

1. INTRODUCTION

Numerous advanced driver assistance systems will support the drivers of future vehicle generations, who will face more complex situations than today due to the ever-increasing density of traffic in growing megacities. At this stage, prototypic systems are already capable of driving autonomously in predefined scenarios, which has been successfully demonstrated during the DARPA Grand and Urban Challenges [29]. These vehicles typically perceive their environment through various complex sensor systems, which are connected to one or more high-performance PCs that execute the computation-intensive assistance applications. Aspects such as power consumption or hardware costs are of minor importance for these prototypes; however, they are crucial for a successful introduction of the demonstrated features in production vehicles.

Consequently, the automotive industry requires high-performance computing platforms for advanced driver assistance systems that are able to satisfy the processing demands at moderate costs and power consumption. Field Programmable Gate Array-based (FPGA) reconfigurable architectures are a potential platform, as they provide

sufficient processing power for complex real-time applications and have already been successfully integrated in safety-critical space research applications [22]. Their flexible peripheral interconnect allows the integration of nearly every relevant sensor setup, and the programmable logic can also be utilized to run in-field updates for bug fixes. Key disadvantages of FPGAs are the time-intensive development process and high single chip costs. For complex applications, large and expensive silicon devices, as well as a lengthy development cycle, are currently inevitable.

In order to overcome these cost and development effort limitations, this paper presents an FPGA-based processing platform combined with a flexible weakly-programmable data flow architecture and a high-level prototyping framework that enables an efficient reuse of FPGA resources at design and execution time. The presented architectural concept enables the utilization of smaller and cheaper devices for complex applications, which allows embedding FPGAs in cost-driven automotive systems. The usability and performance of the target platform are evaluated by an advanced Motion Estimation (ME) application, which is a fundamental processing step for the detection of moving

objects in an incoming front-view image stream. The application achieves a performance of up to 97 GOPS and a frame rate of up to 607 FPS for a commonly used automotive image resolution of 512×384 pixels. Its thorough evaluation demonstrates the cost and power benefits of the platform concept and shows a novel path for efficient automotive FPGA integration.

The rest of the paper is organized as follows. Section 2 gives an overview of related automotive design approaches and previous work. In Section 3, the general platform concept is presented, followed by a brief overview of the target hardware board and the platform's software integration in Section 4. The demonstration application and its evaluation are described in Section 5, and Section 6 concludes the paper.

2. RELATED WORK

For automotive sensor applications, various static and reconfigurable design approaches and accelerator concepts have been presented in the past. First, several software-based architectures provide flexible, distributed execution of multiple applications on multiple shared Electronic Control Units (ECU) [21], which communicate via buses. The AUTOSAR consortium seeks to standardize these software systems and their interactions [4]. A run-time reconfiguration of these systems at the software level is also feasible [2]. However, a software-based approach does not provide sufficient processing performance for high-performance sensor applications.

Some accelerator architectures are based on a Single Instruction Multiple Data (SIMD) multi-core processor network, in which groups of processing elements can operate on a defined data set in a parallel fashion [17]. Others utilize a combination of fixed control processors and reconfigurable coprocessors, which accelerate specific algorithms and can be reprogrammed using partial dynamic reconfiguration [7]. A different reconfigurable architecture, which aims at the design of a low-power cabin control ECU, is presented in [5]. The architecture enables execution of multiple applications on four reconfigurable engines, which are controlled by a dedicated configuration manager. However, the targeted class of cabin functions does not require excessive computation and data transport capabilities.

For automotive vision, other approaches combine a reconfigurable array with optical systems inside the image sensor [36] in order to implement high-performance processing of applications with limited complexity. Some commercial systems combine a dedicated control processor with specialized coprocessors [27], which are able to execute a predefined set of software applications and operations. Other systems use a combination of a Reduced Instruction Set Computer (RISC) core with Digital Signal Processor (DSP) accelerator units [15]. However, such approaches lack the flexibility of an FPGA-based solution, as they are limited by their fixed architecture template, which cannot be adapted according to the application requirements.

Additionally, several non-automotive high-performance architectures exist, which are also suitable for sensor and image processing applications. In the related domain of high-resolution image processing, specialized Application Specific Integrated Circuit (ASIC) approaches [30] are often used in order to implement a given application without offering hardware reuse. Other accelerators [8],[1] offer fields of DSP-like or single-bit processing units, which can be programmed with a specific image processing application. For the presented case study, several similar implementations using ASICs [35] or FPGAs [26] exist. Finally, recent Graphics Processing Units (GPU) [6] are another potential image processing architecture. They achieve high clock frequencies, which results in impressive computational and data transport capabilities, but come at the cost of high power consumption.

Considering all these different accelerators and approaches, FPGAs have been selected as the preferred hardware platform as they provide a suitable mix of high processing performance, modest power consumption, flexible peripheral interconnect and reconfiguration capabilities. Our work is based on the FlexWAFE stream processing architecture, which offers a component-based FPGA design approach combined with weakly-programmable processing elements and fast context switches [19]. This programming concept is similar to specific coprocessors in MPSoCs [9], which also separate fast local instructions inside the coprocessors from the comparatively slow global program execution.

3. PLATFORM CONCEPT

The general system architecture of the accelerator platform can be seen in [Figure 1](#). The platform follows a modular design concept and is built around the aforementioned stream processing architecture, which is utilized for the implementation of accelerated image processing applications. This core component, which is described in more detail in Section 3.1, is enhanced by several essential peripheral components and a platform-independent software interface running on a host PC or ECU. Each peripheral component is connected via a light-weight data streaming protocol, which is based on a Synchronous Data Flow (SDF) model of computation [14] and consists of only three signals. This eases the exchange and integration of new peripheral components, which only have to be adapted to a simple, standardized interface.

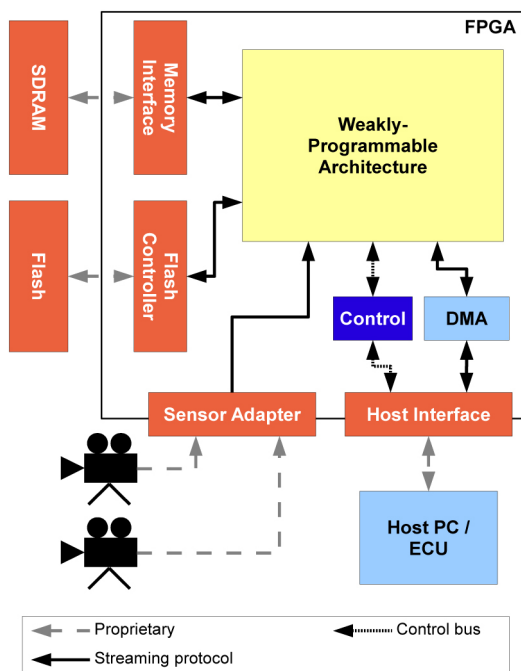


Figure 1. System Architecture

According to SDF semantics, the protocol implements blocking read and blocking write transactions via bounded buffers, which avoids loss of data and allows a data-driven execution of components without requiring activation by a global scheduler. This is beneficial for the design of an automotive system, which consists of various sub-components delivered by different suppliers, and addresses the automotive industry's ambitious efficiency goals.

If components require control information to adapt to certain run-time changes or situations, a dedicated control bus can be used to exchange data with the controlling host PC or ECU. Again, the interface is kept simple and small (*address*, *data*, *read*, *write* signals) to allow efficient component integration. In case synchronization of components is required, the architecture provides modules to register components and to specify synchronization conditions. All these architectural mechanisms ease the integration of new or third-party components, which are often required in application-specific automotive setups.

For the targeted application domain, a suitable sensor integration and flexible components for off-chip and Flash memory integration are of major importance, as these are inevitable for platform reconfiguration and buffering of large data sets. Therefore, Section 3.2 and Section 3.3 present the platform's flexible solutions, which can be tailored to different application scenarios. The developed sensor concept and its implementation are explained by an example sensor setup using CameraLink cameras in Section 3.4, representing a typical automotive sensor type.

3.1. WEAKLY-PROGRAMMABLE ARCHITECTURE

Figure 2 illustrates the basic concept of the weakly-programmable stream processing architecture, which has originally been designed for a high resolution image processing platform. The architecture consists of three component types: Data Processing Units (DPU) contain the computation kernels of an application such as image scaling or a color space conversion, Local Memories with Controllers (LMC) implement data transport mechanisms such as access to external memory or data reordering, and Algorithm Controllers (AC) provide run-time reconfiguration for the other two component types via weak programmability. Large applications can be developed by the composition of an application graph consisting of different independent DPUs and LMCs, which are connected via the aforementioned standardized data streaming protocol. An existing component library, consisting of platform-independent hardware modules, already provides common LMC and DPU components, which significantly reduces the development effort for new applications.

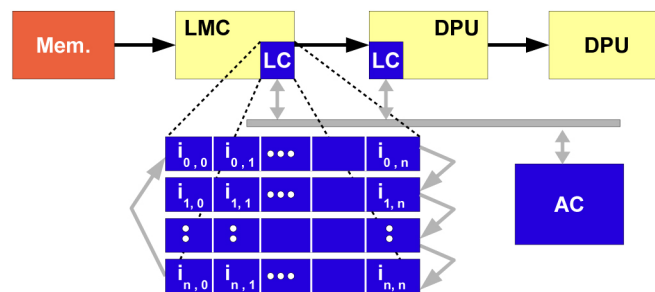


Figure 2. Weakly-Programmable Concept

Programmability is achieved via so-called memory sets, which store component-specific instructions in Local Controllers (LC) inside the components. The memory sets are managed inside the ACs and are transmitted via a dedicated parameter bus as Very Large Instruction Word (VLIW) instructions. After its initial reset phase, a DPU or LMC is triggered by its LC and starts executing the first memory set (instructions $i_{0,0}$ to $i_{0,n}$). As soon as an element-specific break condition is reached (e.g. an LMC reads a complete image from memory), the next memory set ($i_{1,0}$ to $i_{1,n}$) is loaded by the LC and the processing element restarts with new parameters. The context switch between two memory sets is implemented using shadow registers, which results in a context switching time of only one clock cycle. This avoids long pipeline stalls and the need for large intermediate buffers, and allows an efficient execution of several instruction sets using the same hardware resources. As soon as an application completes its last memory set ($i_{n,0}$ to $i_{n,n}$), the LC jumps back to the initial memory set, which enables a looped execution of instruction sets.

This technique of an efficient run-time reuse of components via multiple memory sets offers significant advantages, as it allows the execution of several, similar processing steps using the same hardware resources, which reduces the overall amount of chip area required. Our first adaption of this advanced concept to the automotive application domain results in a measurable reduction in cost and energy, which are both critical factors in the automotive industry, and creates a novel perspective for FPGA introduction in series vehicles. A more detailed presentation of the architecture and its components can be found in [19].

3.2. MEMORY CONTROLLER INTEGRATION

For the targeted application domain of high-performance sensor applications, off-chip memory integration and performance is one key design challenge. Several application parts with different throughput and latency requirements access a shared off-chip memory resource, which quickly becomes the system's performance bottleneck if the selected memory solution does not match the application's requirements. Therefore, the platform provides a generic memory interface solution with standardized application ports and a customizable Synchronous Dynamic Random Access Memory (SDRAM) interface, which simplifies the integration of different memory controllers. This enables an efficient design of tailored memory solutions for diverse application scenarios.

The application interface is shown in Figure 3 and is customized by four parameters at design time. These parameters can be modified for each set of applications, and define the amount of high and standard priority application read and write ports. In case the controller does not support prioritized memory accesses, high priority ports are simply left unconnected. The application ports connect to programmable address generators, which perform burst-oriented memory accesses with a configurable burst size, data word and address width. Memory access is granted by a dedicated *busy* signal, which indicates whether the controller is able to accept new requests. For the write ports, *addr* and the beginning of a *data* burst are sampled by a synchronous *valid* signal. For the read ports, a request-reply mechanism is implemented, which submits the *addr* and *valid* signals and waits for a reply burst (*data_avail*, *data*) by the controller.

The physical SDRAM interface contains technology specific constants and needs to be customized per controller and memory type. The interface covers all relevant signals for double data rate (DDR) and DDR2 SDRAM, which enables an easy adoption to different memory setups. Using this memory template, the platform already provides three integrated memory solutions for different application scenarios, which are described in the next subsections.

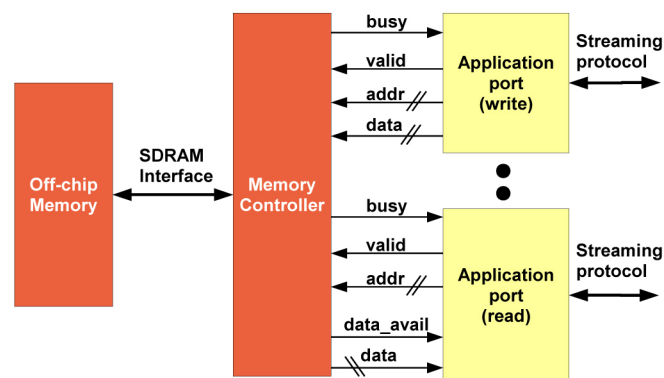


Figure 3. Memory Controller Integration

Xilinx MPMC

Bundled with its design tools, Xilinx ships a Multi-Port Memory Controller (MPMC) IP core [33], which supports various Xilinx FPGA families and memory types. The controller offers up to 8 application ports, but does not support prioritized accesses. Advanced optimization techniques, which enable a significant increase in data throughput at the cost of an enhanced design complexity, are not available with this controller. Therefore, the MPMC is only able to satisfy applications with medium data transport requirements, as it cannot fulfill the demands of a high-performance application with multiple read and write ports.

CMC

A custom-designed Central Memory Controller (CMC) [10] for DDR SDRAM, which provides several optimization techniques, such as bank interleaving or request bundling, has also been integrated in the platform. The controller supports up to 8 application ports and quality of service features via two different priority levels. It reaches up to 70% of the theoretical memory performance, which is achieved by memory access optimizations enabled by a two-stage scheduler and deep request and reply buffers. The controller is able to satisfy significant memory demands by high-performance applications, which has already been demonstrated during the FlexFilm and MORPHEUS projects [18],[32].

BlockRAM-based MC

A memory controller based on the FPGA's internal BlockRAMs completes the available memory solutions. It supports a generic amount of non-prioritized ports and can allocate a generic amount of BlockRAM memory. However, due to the limited amount of on-chip memory available, the controller can only be utilized for applications that require small data buffers. As BlockRAMs are based on SRAM cells, SDRAM optimization techniques have been omitted. The controller is mainly utilized during the testing and verification stage, as it does not come with a lengthy