

Heterogeneous Embedded Architecture for Target Recognition in a Driver Assistant System

Haisheng Liu, Smail Niar, Yassin El-Hillali and Atika Rivenq

Université de valenciennes BP 311 le Mont-Houy, Valenciennes, France

Email: {haisheng.liu, smail.niar, yassin.elhillali, atika.menhaj}@univ-valenciennes.fr

Abstract—This paper presents a new Radar-based recognition system, which is able to identify obstacles during a vehicle movement. Obstacles recognition gives the benefits of avoiding false alarms and allows generating alarms that take into account the identification of the obstacle in front of the vehicle. In this paper, we first identify hotspots in the target recognition application. Then, we propose an optimized version of the multiple target recognition algorithm to respect the real time constraints of the application while simplifying the underlying hardware platform. We also propose a flexible heterogeneous architecture that supports the proposed algorithm. Using a low cost FPGA-based System-on-Chip, our system is able to detect and recognize more than 10 obstacles of different types in a time limit of 25 mSec.

Index Terms—FPGA, embedded systems, driver assistant applications, SoC.

I. INTRODUCTION

Thousands of people around the world lose their lives in road accidents every year. Analyses have shown that most of the accidents are caused by the driver's inattention due to physical and mental fatigues. Automatic early warning onboard system reduces the pressure on the driver. In the literature, such systems are called Driver Assistance Systems (DAS). These DAS's help to establish a confident driving and improve road safety even in stressful driving conditions such as at night or in bad weather. Adaptive cruise control [1], Radar-aided automatic proximity control, and navigation systems, e.g., Global Positioning System (GPS), are examples of well-known range of high-tech DAS's.

Historically, the Application Specific Integrated Circuits (ASICs) [2] occupied the first place in automotive silicon use because of their cost-effective silicon solution. The rising cost of ASIC development and the price drop of CMOS logic gate lead to the end of this ASIC-dominated era. Besides, increasing levels of complexity and computational demands in automotive applications are forcing a move to more powerful yet cost-effective processors [3]. Automotive systems have been designed using 8-bit or 16-bit microcontrollers. As a viable alternative to ASICs, Field Programmable Gate Arrays (FPGAs) have been emerged to implement various automotive subsystems [4]. The application of FPGA technology significantly reduces engineering development time and the cost of multiple silicon iterations. With their inherent support for parallelism, high logic densities, and rich sets of embedded hardware components, FPGAs are very well suited for implementing computationally demanding applications. Their

flexibility, programmability, and fast prototyping times enable system designers to quickly introduce new features or upgrade existing ones in response to changing requirements or new standards [5].

Recent research activities concentrate on the use of DAS in complex environments and scenarios, such as detecting pedestrians, children and under changing weather and lighting conditions. However, existing DAS systems with their limited functionalities and cost constraints for large-scale automotive utilization, can not support these complex applications. Hybrid FPGA-based System-on-Chips give more opportunities to offer good performance, flexibility and cost trade-off designs. In one side, these systems allow the realization of programmable and flexible systems by the use of the multiple integrated hard and soft-cores. On the other side, they make easy the realization of customized hardware processing units by the use of the FPGA's logical elements.

In this paper, a new hybrid programmable FPGA-based architecture is presented to support a multiple obstacle recognition application. This recognition feature categorizes the obstacles and thereby improves the system performance of avoiding false alarms. The alarms are reciprocally generated by taking into account the identification information. Our architecture allows the identification of a big number of obstacle types in a relatively short time. This paper focuses on the architecture design and optimization issues of a Target Recognition System (TRS). Here, both the words target and obstacle have the same signification.

The paper is organized as follows. Section II summarizes the existing projects in the area of DAS. The principles of Radar-based detection system are introduced in Section III. Section IV exposes an overview of the target detection and recognition algorithms. Our heterogeneous embedded FPGA-based architecture is detailed in Section V. Finally, our conclusion and perspectives are presented in section VI.

II. PRESENTATION OF THE EXISTING DAS SYSTEMS

ImapCAR [6] and EyeQ2 [7] systems are two examples of full programmable processors that are dedicated to automotive security applications using vision system. The ImapCar adopts a SIMD (Single Instruction Multiple Data) architecture of 128 processing elements and a 4-way VLIW (Very Long Instruction Word) control processor. The EyeQ2, a single chip with monoscopic embedded vision system, consists of two 64-bit floating-point RISC 34KMIPS processors for scheduling

and controlling the concurrent tasks, five vision computing engines and three vector microcode processors. These two architectures provide support for a specific set of real time data intensive applications. Therefore, these systems are unable either to accommodate new applications or to adapt the hardware to different scenarios. The AutoVision processor [8], is a dynamically reconfigurable MPSoC (Multiple-Processor System On Chip) prototype for video-specific pixel processing. Pixel processing engines offer functions such as object edge detection or luminance segmentation, and are implemented as dedicated hardware accelerators to ensure real time processing.

This paper aims to design a new early warning and collision avoidance application for Intelligent Transportation System (ITS). The system collects data from different sensors onboard a vehicle. These data may contain one or more potential obstacles of interest. Each of the obstacles is then recognized by comparing the received data to a set of predefined obstacle identifiers, called *signatures* in the rest of the paper. Compared with previously mentioned projects and systems, our system uses Radar sensors instead of video camera. The data processing method and the system constraints are thus different. Thanks to its large Field of View (FoV), Radar sensor helps to detect obstacles at longer distances and consequently ensures longer reaction time for vehicle drivers. Moreover, the Radar system behaves better in bad visibility conditions (foggy weather, rain and snow etc.) and has lower computational requirements.

In addition to its ability of intelligent target tracking and obstacle detection, our system is able to recognize obstacles. This is not yet the case in most of the popular DAS in the current market. Let us suppose a pedestrian and a truck have been localized 5 meters far away from the equipped vehicle. Clearly, these two objects do not represent the same danger level. In fact, the identity information, i.e., the target signature, helps the system to determine its corresponding danger level and thus make possible a better alarm generation. To our knowledge, this feature has been rarely investigated in a radar based DAS up to now.

In Reference [5], the authors designed a multiple target tracking DAS that detects and monitors the dynamic behaviour of one or more obstacles in the way of the host vehicle. Their system is also based on an MPSoC architecture. The role of the MTT (Multiple Target Track) is to balance the inaccuracy of the low-cost radar by the utilization of data tracking and filtering processes. Their MTT architecture does not take into account the obstacle identification. As mentioned in the perspective section, our system can be considered as complementary rather than competitors to their works.

III. UWB-RADAR DETECTION SYSTEM

In this section, the basic principles of UWB (Ultra Wide Band) impulse Radar system are introduced. An overview of a simplified UWB-Radar based Target Recognition System (TRS) is given in Fig. 1. The UWB generator sends periodically an impulse signal \tilde{S}_i via the antenna Tx. After the targets has been detected in the Field of View (FoV) of the

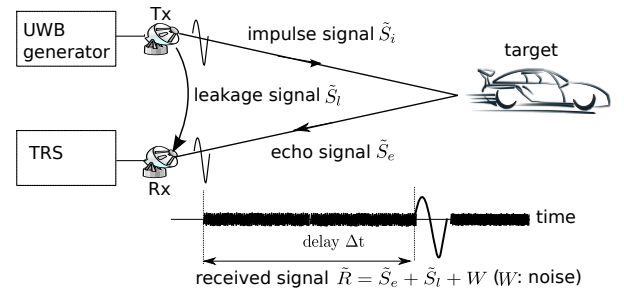


Fig. 1. A simple UWB-Radar based target recognition system

Radar, the echo signal \tilde{S}_e is received at the antenna Rx as well as the leakage \tilde{S}_l . This leakage corresponds to the direct transmission from the antenna Tx to Rx. In addition, a white Gaussian noise W is imposed by the transmission channel. The received signal \tilde{R} is thus the sum of the signals \tilde{S}_e , \tilde{S}_l and W (Fig. 1). The signal processing is performed in real time inside the target recognition system, which is itself connected to the antenna Rx. The measured distance D of a detected target is expressed by the following equation (Equation 1):

$$D = \frac{\Delta t}{2} \times \bar{C} \quad (1)$$

where \bar{C} is the light speed and Δt denotes the impulse transmission delay. Mathematically, the delay Δt is obtained by performing a correlation between the emission and reception signals. With regard to the impulse, different waveforms are possible: monocycle, Gaussian impulse or Gegenbauer function form [9].

IV. TARGET DETECTION AND RECOGNITION ALGORITHM

In this section, the detection algorithm for the UWB impulse Radar system are firstly introduced. Secondly, the target recognition principles are reviewed as well as the associated computational complexity.

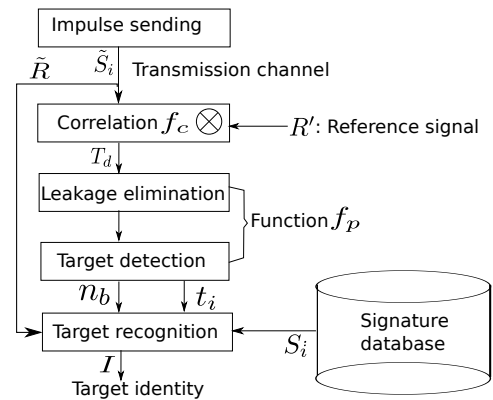


Fig. 2. General outline of data flow processing in TRS

A. Correlation and target detection algorithm

The general outline of data flow processing is presented in Fig. 2. We do not consider the impulse sending step here.

The correlation function f_c between the signals \tilde{R} and R' is first performed and yields the product T_d (see Fig. 2). The reference signal R' corresponds to the received signal when performing an impulse transmission from the antenna Tx to Rx. The correlation function will be reviewed at the next section. Mathematically, the amplitude of the product T_d measures the correlation degree to which the relevant signals resemble each other. Fig. 3 shows an example about the correlation amplitude versus time. The time unit here is equal to 50ps.

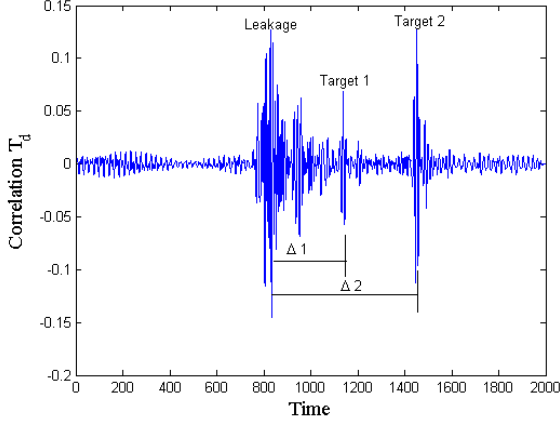


Fig. 3. Correlation Radar detection algorithm using Gaussian impulse form

The next steps consist in finding the peaks, the maximum values in time intervals, which indicate the tracked targets. The corresponding distances are obtained from the associated time values. The adjacent peaks in case of very close targets within a small time interval are combined and considered a single target. In Fig. 3, the first peaks in time interval 775 to 960 correspond to the leakage and must be eliminated, as done in the leakage elimination step. The other peaks show the potential targets. Hence, two targets are respectively detected at times 1138 and 1451 in this case. Both the leakage elimination and the target detection are realized by the function f_c , as indicated in Fig. 2. The number of targets, i.e., $n_b = 2$, and the respective time values t_i are then obtained. The delay Δ_i is the time difference between the leakage and the target, as shown in Fig. 3.

B. Target recognition algorithm

The target recognition is a key feature of the system. It takes into account the number of targets n_b and the time t_i obtained at the previous stage to identify the tracked targets. Actually, each target implies an inherent echo signal form, described also as signature. The recognition algorithm is an iterative detection process through a preliminary built signature database. Let us define a database containing L signatures, denoted by the set $S = \{S_0, S_1, \dots, S_{L-1}\}$. The operations involved at the recognition stage are similar to that at the detection stage.

For each detected target, the received signal \tilde{R} is successively compared to the known signatures. These comparisons

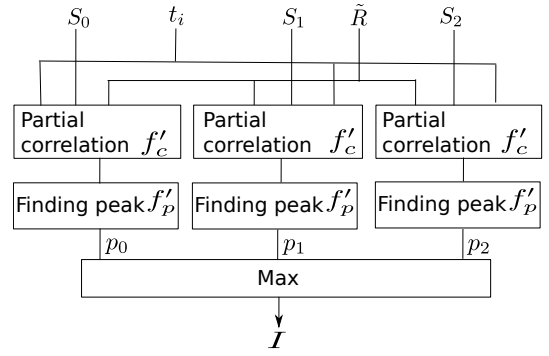


Fig. 5. Diagram for the target recognition with 3 signatures

are realized by the correlation function. Fig. 5 depicts the data processing diagram for a single target with 3 signatures: S_0 , S_1 , S_2 . For a time-known target, it is useless to perform a correlation operation over all the signal samples. That is why a partial correlation, denoted by the function f'_c in Fig. 5, is proposed in this stage. In other words, the correlation is merely performed within a time interval based on the time t_i . Compared with the function f_c , the function f'_c requires a small number of operations. This proposal will be discussed again for complexity issues in the following section. The function f'_p allows to find the individual peaks, marked by the symbols p_0 , p_1 , p_2 in Fig. 5. Finally, the identity I is obtained in such a way that the corresponding peak has the highest value, which is in fact produced by the function Max. The recognition algorithm is executed n_b times for the same number of targets.

Based on the same source of Fig. 3, an illustration of the algorithm is given by Fig. 4. Three signatures are known in the database S : Pedestrian, Metal plate and Car. Firstly, the partial correlation lines are respectively drawn in the figures a), b) and c) (see Fig. 4). Secondly, by running the the function f'_p , three valued peak points are respectively obtained, i.e., $p_1 = p_2 = 0.05$ and $p_0 = 0.06$. Lastly, the target are identified as a pedestrian since the peak p_0 has the maximum value in this case.

C. Computational complexity survey

The processing complexity and memory loads of the proposed system play an important role in system design. These two factors must be reduced to allow a feasible realization. As explained, the complexity of our proposal depends mainly on two key values: the number of targets n_b and the complexity of correlation. Let us now investigate the correlation function, which is in fact a major operation in digital signal process. Mathematically, the correlation function \otimes between two functions f with length M and g with length N ($M > N$) is expressed by the following formula (Equation 2).

$$(f \otimes g)(n) = \sum f(k) \times g(k - n), \quad n \in [0, M] \quad (2)$$

where both n and k are integer values. Both the functions f and g have the following property : $f(k) = 0$ for $k \notin [0, M]$

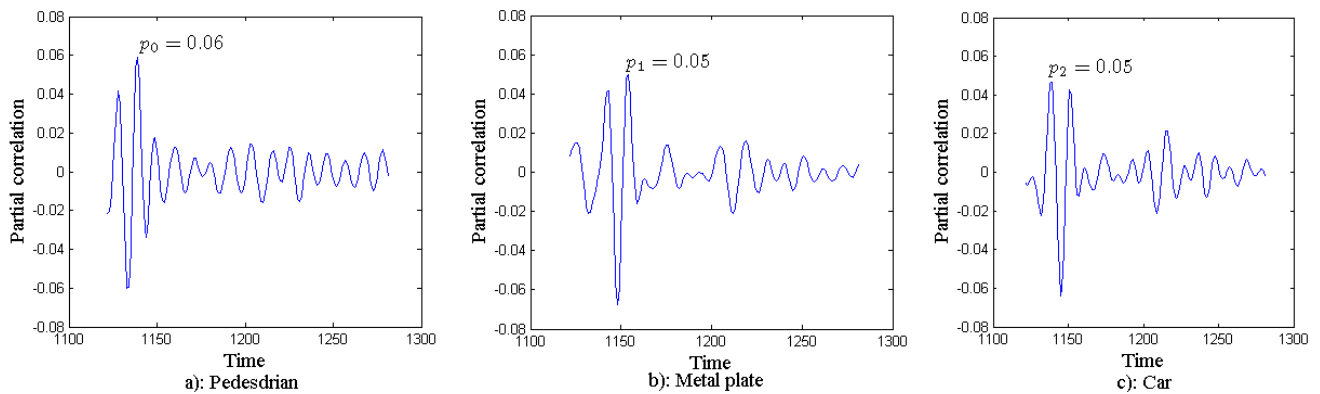


Fig. 4. Target recognition algorithm with a database having 3 signatures

and $g(k) = 0$ for $k \notin [0, N]$. The computational complexity is expressed in number of MAC (Multiplication Addition Cumulation) operations. According to Equation 2, the complexity is equal to the product $(N \times M)$.

For the target recognition stage, it is not necessary to run the correlation function through the overall samples. A partial processing is proposed to avoid redundant data processing. Merely those samples used to identify the targets are selected. Suppose that the signal \hat{R} is composed of M samples (r_0, \dots, r_{M-1}) . If a target is detected at time m , the recognition algorithm is performed over a subset $\{r_{m-L_0}, \dots, r_{m+N+L_0}\}$ of \hat{R} centred around m . Experimentally, the value of L_0 is set around to $0.1N$. Thus, the complexity of the recognition stage is approximatively N^2 .

The complexity of our proposed system is detailed as follows. In our case, there are 2,048 samples for the digital signal \hat{R} and the reference signatures are represented by 128 samples. The correlation of one sample requires 128 memory reads respectively from the signals \hat{R} and R' , 128 MAC operations and 1 memory write back operation to store the result T_d . Hence, for the detection stage, there are in total 526,336 memory loads (Read and Write) and 262,144 MAC operations.

Concerning the complexity of the recognition stage, it depends mainly on the number of targets, the local time interval and the number of signatures. The time interval is bordered by 160 samples in our case. Consequently, the partial correlation requires 20,480 MAC operations and 41,120 memory loads for each signature and each target. According to the Radar characteristics, the scan period is equal to 25ms. In case of 3 signatures and one target for example and to meet the real time constraint, our system must realize 13 MAC operations and 26 memory loads per microsecond.

V. SYSTEM ARCHITECTURE

This section is mainly dedicated to the system architecture. Firstly, the timing profile is performed in Section V-A. Secondly, a configurable heterogeneous architecture is proposed in Section V-B. Thirdly, the hardware component is described in Section V-C. Finally, Section V-D gives the experimental results.

A. Timing profile

The real time constraints imposed by the specific application is the key challenge in designing a microprocessor based embedded system. It is thus important to identify time costs of the functions running in the system. Substantially, the timing profile constitutes an important step to partition tasks between hardware and software components.

With regard to the microprocessor, we use a 32-bit RISC soft core processor, Microblaze, which is developed as soft IP by Xilinx company [10]. It can be easily implemented in FPGA-based system. The profiling architecture will be depicted in the next section. Considering a Microblaze processor running at 100Mhz with 8k cache and 8k data instruction memory, Table I shows the profiling results based on a fixed-point representation of the algorithm.

TABLE I
TIMING PROFILE OF THE RELEVANT FUNCTIONS

Stage	Detection		Recognition	
	f_c	f_p	f'_c	f'_p
Time	1,450	15	93	0.16

unit: ms

As seen in Table I, the correlation function f_c has a significant time-cost and requires more than 1s to complete the process. Additionally, the function f_p performing the tasks of leakage elimination and target detection requires a time-cost of 15ms. Thanks to the partial correlation processing, the function f'_c consumes merely 93ms for the recognition stage. The function f'_p which runs the task of finding peaks has a low time-cost of 0.16ms. This observation shows that the correlation functions (f_c and f'_c) are highly time consuming. Its value is proportional to the correlation length N . As a result, the implementation on a single Microblaze processor does not guaranty the respect of real time requirements.

To consider the real time issue, a hybrid architecture containing hardware and software components is proposed in the next section. The functions having low time-cost (f_p and f'_p) are implemented on the software processor, whereas a hardware accelerator is dedicated to the correlation functions. Eventually, this accelerator can either be individually used or

shared by both the detection and recognition stages.

B. Configurable heterogeneous architecture

In this section, we propose a configurable heterogeneous architecture composed of hardware and software components, as shown in Fig. 6.

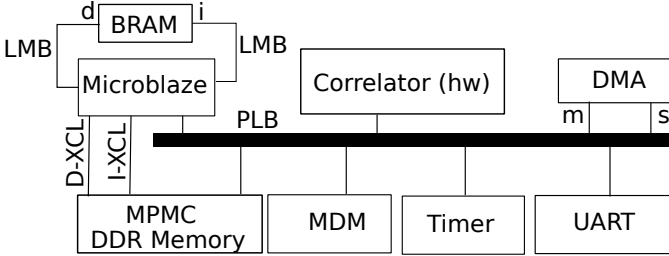


Fig. 6. Configurable heterogeneous system architecture

In this architecture, the key processor components include the Microblaze processor, the hardware correlator and the DMA (Direct Memory Access) controller. The communication between the Microblaze processor and the peripherals is realized through the data transfer bus PLB. The peripherals concern the Timer, the module UART and the external DDR memory. At the same time, the Microblaze processor has its Local Memory Bus (LMB), which connects the private BRAM memory to itself. According to Reference [10], the Microblaze processor features a Harvard memory architecture. It means that this processor has its separate data and instruction bus, as respectively marked by **d** and **i** in Fig. 6.

At the software side, the data and instructions are loaded from the external DDR memory via the peripheral MPMC (Multi-Port Memory Controller). The cache configuration is available for Microblaze to speed up the execution time. Also, we distinguish here the data and instruction cache, marked by **D-XCL** and **I-XCL** respectively.

The hardware correlator handles data processing at a high frequency, up to 370MHz in our application. However, due to the execution delay of Microblaze processor, the data communication through software interface does not adapt to the high speed data processing, as expected by the hardware processor. That is why the DMA technology is proposed here to provide a high rate data communications. This technology automates the movement of large amounts of data without processor control. The DMA controller has two PLB ports: Master and Slave, as marked respectively by **m** and **s** in Fig. 6.

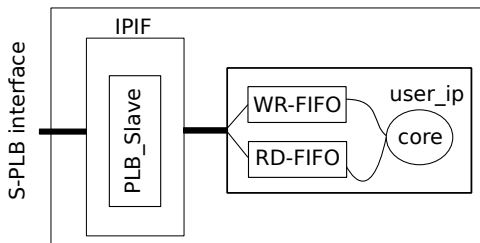


Fig. 7. Hardware accelerator architecture

The hardware accelerator is considered as a peripheral device having a slave PLB interface by the Microblaze processor. Thus, it is necessary to implement this interface to the accelerator. As seen in Fig. 7, the IP module IPIF (IP InterFace) provides a highly adaptable and quick-to-implement interface between the PLB Bus and the hardware core. The basic functionality for PLB Slave operation is provided by the component PLB_Slave. Depending on our requirements, the read and write FIFO buffers, WR-FIFO and RD-FIFO, are optioned in through the use of VHDL files. The corresponding size is respectively equal to 512 words of 32 bits. Thus, the computational core, denoted by `user_ip`, communicates with the external peripherals through a slave PLB interface. Its input and output data is first buffered and then transferred by the DMA controller.

C. Correlation hardware architecture

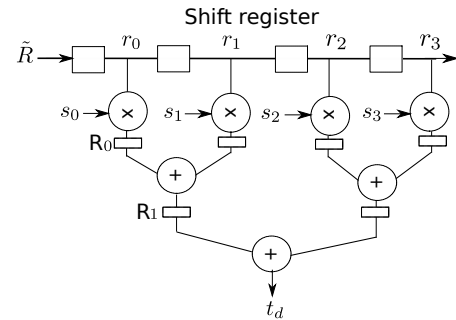


Fig. 8. Correlation architecture for a signature of 4 samples

The hardware correlator (Fig. 7) performs the multiplication and addition computations. For illustration purposes, Fig. 8 depicts an architecture of the correlator with the length $N=4$. This architecture is extended to the length $N = 128$ in our case. The proposed architecture is composed of three major layers. The first layer concerns the shift registers, where the samples r_i are successively stored. The values are updated and shifted right every clock cycle. Then, the samples r_i are respectively multiplied by the signature samples s_i , represented in the second layer. The results are saved in the register file \mathbf{R}_0 and then summed together through a $\log_2 N$ -depth adder tree. The addition delay is then proportional to the value of N . Similarly, the intermediary addition results are buffered in the register file \mathbf{R}_1 before outputting the final result t_d . This architecture has an advantage that the register files guaranty a pipeline execution.

Let us now tackle the time issues. For initialization, it is necessary to fulfil respectively the shift registers r_i and the samples s_i . This initialization consumes 2×2^7 cycles in our case. Furthermore, the throughput delay is equal to 8 cycles (1 cycle for multiplications and 7 cycles for additions). This architecture yields therefore a delay of 2,312 cycles to complete the correlation of 2,048 samples. Suppose that the hardware core runs at the same frequency of the Microblaze processor, i.e., 100Mhz. The delay is then approximatively equal to $24\mu s$.

At the architecture point of view, it is possible to add multiple accelerators, i.e., hardware cores, to speed up data processing. For instance, the partial correlations can be performed in parallel with dedicated correlators. On the other hand, the accelerator can be shared by both the detection and recognition stages. In this case, the samples s_i are dynamically loaded in and out from the external DDR memory depending on the signature. This structure is adopted in the next section.

D. System synthesis and experimental results

The first experiments with the logic synthesis system are attempts to demonstrate the feasibility of our system. The proposed architecture is implemented on the kit board ML605 [11], in which a Xilinx Virtex-6 FPGA circuit [12] is available. The logical synthesis is performed with Xilinx syn-

TABLE II
LOGIC SYNTHESIS RESULTS

Components	Flip-Flop	LUT	BRAM
System	14,049	10,688	23
Correlator	5,608	1,872	1
DMA	557	800	0
Timer	362	291	0
MDM	119	120	0
MPMC	5,052	4,201	11
UART	154	135	0
Microblaze	1,955	2,725	6
PLB	169	476	0
Others	73	68	5

thesis tools. The synthesis results are summarized in Table II, where the component names correspond to the modules mentioned in Fig. 6. The line *Others* contains the synthesis results of the following components: the local memory BRAMs, the LMB buses, the clock management and the input output IPs. According to Table II, the system architecture requires in total 14,049 Flip-Flops, 10,688 LUTs and 23 Block RAMs. The architecture of the proposed correlator needs 996 LUTs and 5,002 Flip-Flops. The Microblaze processor consumes respectively 1,955 Flip-flops, 2,725 LUTs and 6 BRAMs.

TABLE III
TIMING PROFILE OF THE PROPOSED ARCHITECTURE

Stage	Detection		Recognition	
	f_c	f_p	f'_c	f'_p
Time	0.53	15	0.17	0.16

unit: ms

Table III shows the new timing profile of the proposed architecture. In this table, the correlation functions f_c and f'_c requires merely 0.53ms and 0.17ms respectively for the detection and recognition stages. It is important to note that the functions f_p and f'_p are executed by the Microblaze processor. Then, their execution times are not modified compared with Table I. As a result, the total execution time is reduced to 15.53ms for the detection stage. On the other hand, the recognition stage requires 0.33ms for each target and each signature. As an example, considering a database of 3 signatures, the number of targets can attain up to 10 targets with

the same database per Radar scan (25mSec). As a conclusion, the hardware accelerator speeds up significantly the execution time. The proposed hybrid architecture meets the real time requirements.

VI. CONCLUSION

In automotive industries, the Radar-based intelligent system presents a strong detection capacity specially in bad driving conditions. A target recognition application using correlation-based algorithm has been investigated in this paper. As the application requires intensive data processing, respecting real time constraints is considered as a key challenge in the architecture design. Taking into account the real time processing and the limited resource issues, we proposed an hybrid SoC architecture containing both the hardware and software components. The prototyping results show that the hardware accelerator speeds up significantly the execution time. As future investigations, we propose to study the integration of our target recognition system with the multi-tracking system proposed by [5]. Cooperation between these 2 components in a DAS allows to accurately identify a larger number of obstacles with limited hardware resources.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to the French Research Agency and the International Campus on Safety and Intermodality in Transportation for the financial support through the research project Prima-Care.

REFERENCES

- [1] A. Vahidi and A. Eskandarian, "Research advances in intelligent collision avoidance and adaptive cruise control," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 3, 2003, pp. 143–153.
- [2] I. Masaki, "Asic approaches for vision-based vehicle guidance," *IEICE Transactions on Electronics*, vol. E76-C, no. 12, pp. 1735–1743, Dec. 1993.
- [3] M. Valera and S. A. Velastin, "Intelligent distributed surveillance systems: A review," in *IEEE Proceedings - Vision, Image and Signal Processing*, vol. 152, no. 2, 8 April 2005, pp. 192 – 204.
- [4] J. Saad, A. Baghdadi, and F. Bodereau, "Fpga-based radar signal processing for automotive driver assistance system," *IEEE/IFIP International Symposium on Rapid System Prototyping, 2009. RSP '09.*, pp. 196 – 199, 23-26 June 2009.
- [5] J. Khan, S. Niar, A. Menhaj1, Y. Elhillali1, and J. L. Dekeyser, "A mpso architecture for the multiple target tracking application in driver assistance system," *International Conference on Application-Specific Systems, Architectures and Processors, 2008. ASAP 2008.*, pp. 126 – 131, 2-4 July 2008.
- [6] H. Hiroto, "Automotive image recognition processor imapcar," *NEC technical journal ISSN 1880-5884*, vol. 1, no. 5, pp. 24–28, Dec. 2006.
- [7] G. P. Stein, E. Rushinek, G. Hayun, and A. Shashua, "A computer vision system on a chip: a case study from the automotive domain," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPRW'05)*, p. 130, 20- 26 June 2005.
- [8] C. Claus, W. Stechele, and A. Herkersdorf, "Autovision – a run-time reconfigurable mpso architecture for future driver assistance systems," *Information Technology*, vol. 49, no. 3, pp. 181–187, 2007.
- [9] F. Elbahhar, A. Rivenq-Menhaj, and J. M. Rouvaen, "Multi-user ultra wide band communication system based on modified gegenbauer and hermite functions," *Wireless Personal Communications*, vol. 34, no. 3, pp. 255–277, August 2005.
- [10] "Xilinx microblaze processor reference guide, ug081 (v11.0)," *Embedded Development Kit EDK 12.1*, Sep. 14 2000.
- [11] "Xilinx ug534 ml605 hardware user guide ug534 (v1.4)," October 2010.
- [12] "Xilinx virtex-6 fpga configuration user guide ug360 (v3.2)," November 2010.