VOL. 10, NO. 20, NOVEMBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences


www.arpnjournals.com

DESIGN AND IMPLEMENTATION OF A GAS IDENTIFICATION SYSTEM ON ZYNQ SOC PLATFORM

Amine Ait Si Ali[1], Abbes Amira[1], Faycal Bensaali[1], Mohieddine Benammar[1], Muhammad Ali Akbar[1], Muhammad Hassan[2] and Amine Bermak[2]
[1] College of Engineering, Qatar University, Doha, Qatar
[2] School of Engineering, Hong Kong University of Science and Technology, Hong Kong
E-Mail: amine.aitsiali@qu.edu.qa

ABSTRACT

The Zynq-7000 based platforms are increasingly being used in different applications including image and signal processing. The Zynq system on chip (SoC) architecture combines a processing system based on a dual core ARM Cortex processor with a programmable logic (PL) based on a Xilinx 7 series field programmable gate arrays (FPGAs). Using the Zynq platform, real-time hardware acceleration can be performed on the programmable logic and controlled by a software running on the ARM-based processing system (PS). In this paper, a design and implementation of a gas identification system on the Zynq platform which shows promising results is presented. The gas identification system is based on a 16-Array SnO_2 gas sensor, principal component analysis (PCA) for dimensionality reduction and decision tree (DT) for classification. The main part of the system that will be executed on the PL for hardware acceleration takes the form of an IP developed in C and synthesized using Vivado High Level Synthesis for the conversion from C to register transfer level, a hardware design for the entire system that allows the execution of the IP on the PL and the remaining parts of the identification system on the PS is developed in Vivado using IP Integrator. The communication between the processing system and programmable logic is performed using advanced extensible interface protocol (AXI). A software application is developed and executed on the ARM processor to control the hardware acceleration on the programmable logic of the previously designed IP core and the board is programmed using Software Development Kit. The maximum accuracy achieved by the system to classify three types of gases CO, C_2H_6O and H_2 is 96.66%.

Keywords: zynq soc, hw/sw co-design, high level synthesis, vivado, gas identification system.

INTRODUCTION

The Platforms based on the Zynq system-on-chip (SoC) are being used in various applications, mainly for image and signal processing. An image filter evolution is used in (Dobai and Sekanina, 2013) as an example to evaluate the suitability of the Zynq Platform for evolvable hardware (EHW) implementation using virtual reconfigurable circuits and dynamic partial reconfiguration. In (Russell and Fischaber, 2013) a hardware-software (HW/SW) co-design approach is applied to implement a road sign recognition algorithm on the Zynq SoC. The system is designed to detect blue and red signs. Pre-processing and image filtering of high definition (HD) quality images is performed on the programmable logic (PL) while the recognition is executed on the processing system (PS) using OpenCV. A traffic sign recognition system was also implemented on the Zynq SoC in (Han and Oruklu, 2014). The presented solution is similar to the one in (Russell and Fischaber, 2013) where pre-processing and image detection is performed in the PL and the recognition part is executed in the PS. The solution has been compared to an implementation on a Virtex 5 field programmable gate array (FPGA). The solution presented in (Monson *et al.*, 2013) is concerned with the implementation of a complex optical-flow algorithm on both PS and PL. It is first implemented on the ARM Cortex A9 based PS and compared with an implementation on an Intel i7 Core 2.8GHz based desktop running windows 7. The obtained results show that the execution on the ARM processor was eight times slower than it counterpart on the core i7. However, power consumption is 19 times less. The algorithm was then implemented in PL. The hardware implementation is 3.6 times faster than the ARM one while consuming 3.2 times less energy. It was shown that the hardware acceleration of the optical flow algorithm on the PL is competitive with the execution on the Core i7 while consuming 7.4 less energy. A summary of the previously described HW/SW implementations on the Zynq is described in Table-1.

Gas monitoring in general and gas identification in particular is a critical task since the consequences of leaks or dangerous mixtures in production, distribution or domestic environment can be devastating. The use of hardware to accelerate algorithms used in gas identification is highly recommended to improve the response time.

The aim of this paper is to present an innovative architecture for the implementation of a gas identification system on the Zynq SoC using HW/SW co-design approach as well as high level synthesis (HLS).

The rest of the paper is organized as follows. A literature review related to gas identification is presented in section II. In section III, a brief description of the Zynq SoC is given as well as details about the HW/SW co-design approach when targeting the Zynq SoC. In section IV, an implementation of a gas identification system on the Zynq SoC is presented. Section V concludes the paper.

Table-1. Software and hardware implementations on the Zynq SoC.

Papers	Zynq SoC	Tools	Applications	PL	PS
(Dobai and Sekanina, 2013)	XC7Z020	Not mentioned	Image filter evolution for EHW	Evolution algorithm	Fitness evaluation
(Russell and Fischaber, 2013)	XC7Z020 (Zedboard)	PlanAhead, XPS and SDK	Road sign recognition	Image filtering	Image recognition
(Han and Oriole, 2014)	XC7Z020 (Zedboard)	EDK (XPS and SDK)	Traffic sign recognition	Image detection	Image recognition
(Monson et al., 2013)	XC7Z020 (Zedboard)	Vivado HLS and EDK (XPS and SDK)	Optical flow algorithm	Optical flow algorithm	Optical flow algorithm

Table-2. Summary of some gas identification systems based on sensor array.

Papers	No. Sensors	Target gases	Pre-processing	Classification	Implementation
(Shi et al., 2008)	8	CO, H_2, CH_4, CO-H_2 & CO-CH_4	EN & PCA	Committee machine: KNNs, MLP, RBF, GMM & PPCA	FPGA Celoxica RC203
(Benrekia et al., 2013)	8	CO, H_2, CH_4, CO-H_2 & CO-CH_4	EN	MLP	FPGA APS-X208
(Far et al., 2009)	16	CO, H_2, CH_4 & C_2H_5OH	SOM	IM & LDA	PC
(Kim et al., 2012)	8	O_3, LPG/LNG, NO_X, Alcohol, Smoke, VOC, CO& NH_3	SMA , Normalization between 0 and 1 & PCA	GA & ANN	Laptop, Zigbee & phone
(Brahim Belhouari et al., 2004)	8	CO, H_2, CH_4, CO-H_2 & CO-CH_4	PCA vs LDA vs NS	Density models: KNNs, GMM & GTM Discriminant functions: RBF, MLP and GLM	PC
(Ng et al., 2008, 2009a-b)	16	CO, H_2 & C_2H_6O	LSTE	RO	VLSI
(Li and Bermak, 2011)	16	CO, H_2 & C_2H_6O	With and without PCA	DT	FPGA (Xilinx Virtex II)+ ASIC

GAS IDENTIFICATION BACKGROUD

Gas identification systems are mainly made of three main building blocks, the sensing part, the preprocessing part and the classification part (Gutierrez-Osuna, 2002). The sensing part takes the form of the gas sensor or a set of sensors and the data acquisition mechanism. Researchers are continually using an array of sensors instead of a single sensor to overcome the non-selectivity problem. In the preprocessing data part, the collected raw data from the sensors is manipulated, it can be normalized, the dimensionality of the input vector can be reduced and the features for classification are extracted. The last part, which is the classification, is where the gas is actually identified using a specific or a set of classification algorithms. It can also take the form of a regression problem in the case of concentration estimation. A summary of various gas identification systems present in the literature is presented in Table 2. The preprocessing algorithms and classification algorithms used are the following: Euclidean normalization (EN), principal component analysis (PCA), linear discriminant analysis (LDA), neuroscale (NS), self-organized map (SOM), smoothed moving average (SMA), logarithmic spike timing encoding (LSTE), rank order (RO), k-nearest neighbors (KNNs), multilayer perceptron (MLP), radial basis function (RBF), Gaussian mixture model (GMM), probabilistic principal component analysis (PPCA), image moment (IM), genetic algorithm (GA), artificial neural network (ANN), generative topographic mapping (GTM), binary decision tree (DT) classifier and general linear model (GLM). The types of implementations used are mainly personal computer (PC), FPGA, very-large-scale integration (VLSI) and application-specific integrated circuit (ASIC). PCA and LDA are the most common dimensionality reduction algorithms used for gas identification. In this paper, PCA is selected to reduce the dimensionality of the input vector containing the responses generated by the sensor array while DT is used as a classifier for its simplicity for hardware implementation yet with high identification accuracy.

HARDWARE SOFTWARE CO-DESIGN ON ZYNQ

The Xilinx Zynq-7000 all programmable SoC combines a traditional FPGA based on Xilinx 7-series

forming the PL with a dual core ARM Cortex-A9 processor forming the PS. The PL is based on Artix-7 or Kintex-7 with different variants. The two parts are using high bandwidth industry standard advanced extensible interfaces (AXI). There are three AXI bus protocol for the interconnection: AXI4, AXI4-Lite and AXI4-Stream. Both AXI4 and AXI4-Lite are memory mapped links. AXI provides the highest performance since it allows the transfer of up to 256 data word in one connection while AXI4 allows the transfer of a single word in one connection. AXI Stream is suitable for high speed streaming data where an address mechanism is not needed. Details about the Zynq-7000 all programmable SoC can be found in (Xilinx UG585, 2014).

The combination of the PS and the PL inside the same chip makes the platforms based on the Zync SoC suitable for HW/SW co-design approach. Especially when associated with Xilinx Vivado high level synthesis (HLS) tool, Vivado IP Integrator and software development kit (SDK) which will allow a high level of abstraction with benefits in terms of performance, cost and power compared to a conventional FPGA or processor implementations. The design flow to transform a slow program running on a processor to a faster hardware running on the Zynq SoC platform using the new Vivado integrated development environment (IDE) design suite is shown in Figure-1.

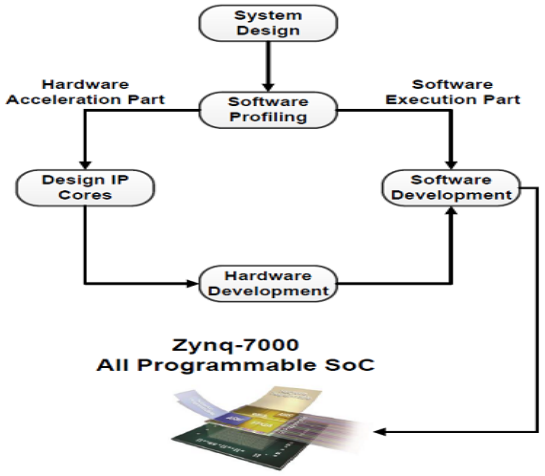

Figure-1. Design flow for HW/SW co-design on Zynq.

It can be seen from Figure-1 that the starting point for a HW/SW co-design on the zynq SoC is the system design where all the desired requirements and specifications of the top-level system and various subsystems are set. The following stage is the software profiling where a software code is executed in a processor and with the help of a profiling tool such as GProf (Fenlason and Stallman, 1997), computationally intensive parts of the program can be identified. The designer can then decide to execute computationally intensive parts on hardware (i.e. PL) and the remaining on the ARM processor (i.e. PS). The third step consists of the design of IP Cores for hardware acceleration of the computationally intensive parts. IPs can be designed using various approaches. One approach is the use of Vivado HLS where the IPs are created and simulated using C, C++ or System C and then exported. Another approach is the use of Xilinx System Generator. The IPs can also be designed using a hardware description language (HDL) such as VHDL or Verilog directly. In all cases the IPs are exported and added to Vivado IP Catalog. A hardware system is then created in Vivado using IP Integrator where all blocks are interconnected including Zynq SoC and the previously created IPs. The following stage is to export the hardware design to SDK, here a software code is to be written and executed on PS. The software corresponds to the parts of the design not implemented on PL and also to manage the IPs implemented on the PL. The PL is programed from SDK.

GAS IDENTIFICATION SYSTEM ON THE ZYNQ

The gas identification system is based on a 16-Array SnO_2 gas sensor, PCA as a dimensionality reduction technique and binary DT as a classifier. The gas sensor array is an in-house fabricated component (Guo *et al.*, 2007). First the gas identification system is designed, simulated, evaluated and validated in MATLAB. Then Vivado HLS is used to create the corresponding register transfer level (RTL) design for the system algorithm described in C. The next step is the hardware development performed in Vivado using IP Integrator. The final stage is the software development and FPGA programing realized in SDK. All versions of HLS, Vivado and SDK used for the design and implementation are the latest at the time of writing which is the 2014.4 one. The prototyping board being used is the Zynq ZC702. The gas identification system architecture is shown in Figure-2, data organization and the decision parts are executed in the PS while PCA and DT are implemented on the PL. Both parts are interconnected using the AXI-Lite Interface.

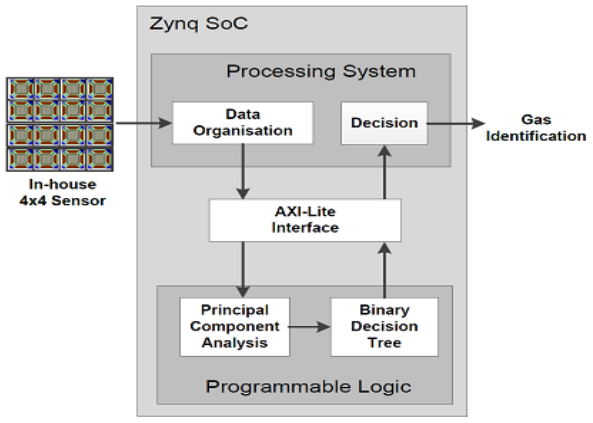

Figure-2. Gas identification system architecture.

Gas identification system

The building blocks of the gas identification system for the learning and testing phases are shown in

Figure-3. The aim is to design a gas identification system to classify three types of gases CO, C2H6O and H2. The system gets 16 floating point values from a 16-Array SnO2 based gas sensor. Those values represent the steady states of the sensor when exposed to a given gas at a given concentration. The first step is to reduce the dimensionality of the input data by decreasing the number of values from 16 to 3 principal components (PCA1, PCA2 and PCA3) using PCA. To perform the dimensionality reduction, a mean normalization is applied to the 16 steady states then a projection is realized to compute the 3 principal components. The third and final stage is to classify the gas using a binary DT. Both parameters for PCA (Vector of means and Eigen vectors) and DT (DT model) are saved from a training performed in MATLAB. For training purposes, the sensor array is exposed to the three types of gases (CO, C2H6O and H2) at ten different concentrations (20, 40, 60, 80, 100, 120, 140, 160, 180 and 200ppm). The experiment is repeated twice to collect 30 patterns for training and 30 patterns for testing. The accuracy of the system is 90%. It can be improved to 96.66% when drift compensation technique is applied and 4 PCAs are used. In this compensation technique the delta between the baseline and the steady state is considered as a feature instead of the steady state only.

PCA and DT IP deign using HLS

The C source code corresponding to the previously described gas identification system is written. The code consists of a function called "Predict". The input of the function is a vector of 16 floating points while the output is an integer ("1" for CO, "2" for C2H6O and "3" for H2). Within the function the vector of 16 means and 3 Eigen Vectors are declared and initialized. The vector of means is used for normalization. The normalization consists in the subtraction of the each mean from the corresponding value of the input vector. The projection consists in the multiplication of the normalized vectors by the 3 Eigen vectors of same size, the resulting 3 floating-points values are the 3 principal components used by the DT to classify the input. The DT takes the form of a succession of "If" and "Else" statements resulting in the output being "1", "2" or "3". A second C file is needed for testing since in Vivado HLS the testbench is also written in C. The C testbench takes the form of the main C function that will execute the "predict function and self-check the results. It is worth mentioning that Vivado HLS allows the user to export the IP to IP Catalog (For Vivado), Pcore (For embedded development kit (EDK)) or system generator. Drivers related to the designed piece of hardware are included in the IP package. They will be used by the software managing the core from the processor. Different optimization directives are applied including loop unrolling, array partitioning and pipelining. A comparison between those optimizations in terms of performances and resources is presented in Table 3 and Table 4. The latency represents the required number of clock cycles taken by the designed system to produce an output while Interval represents the rate in terms of clock cycles at which a new input can be given to the block. The first "Unroll" directive applied to the loop where the mean and projection are computed is very powerful. It allows loops to be executed in parallel having dedicated hardware resources for each loop. Each array in the function can be considered as one entity having limited data ports for data transfer or multiple entities using the "Array Partition" where each entity is having its own data ports. "Array Partition" with factor 4 is applied to the input array of size 16 which resulted in the breakdown of the array into 4 sub arrays. The "Pipeline" directive is applied to the top level function "predict" to allow pipelining of all instructions and sub function existing inside. The last directive which is "AXI Lite" is very important since it will help interconnecting the IP core designed in Vivado HLS for an implementation in the PL with the Zynq PS. The "AXI Lite" directive is applied to the top function "predict" under "Resource" and to the input array along with the output variable under (Interface). Details about Vivado HLS can be found in (Xilinx UG902, 2014).

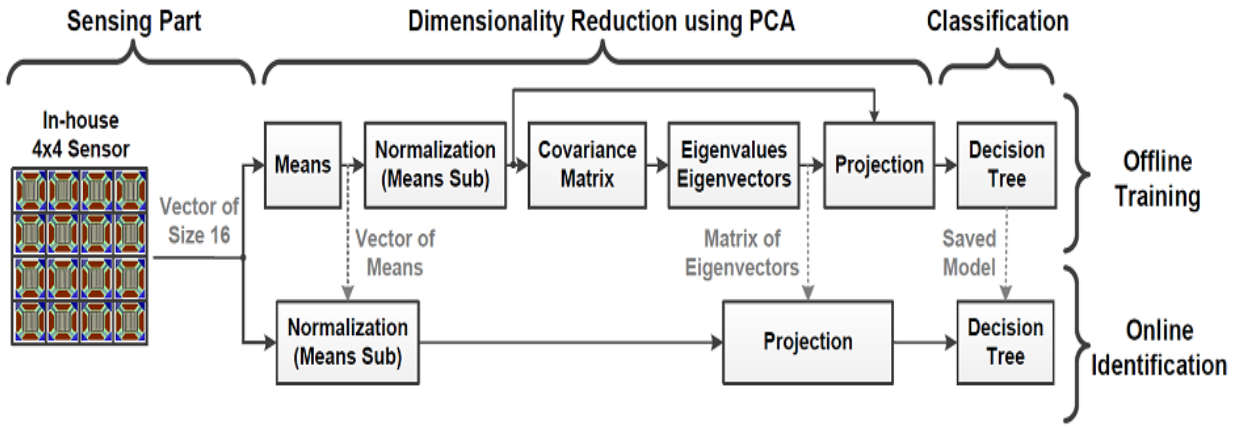

Figure-3. Gas identification building blocks.

Table-3. Implementation results when using the steady states.

Optimization	Without Directives	Unroll Loops	Array Partitioning and Pipelining	AXI Lite Interface
BRAM 18K	0	0	0	8
DSP48E	15	15	136	68
FF	2089	1989	14730	9803
LUT	3925	4830	27208	14676
Clock (ns)	8.20	8.20	8.20	8.20
Latency (clock cycles)	261	95	94	96
Interval (clock cycles)	262	96	2	4

Table-4. Implementation results when using the deltas between the baselines and the steady states.

Optimization	Without Directives	Unroll Loops	Array Partitioning and Pipelining	AXI Lite Interface
BRAM 18K	0	0	0	8
DSP48E	15	5	136	68
FF	1793	1433	14729	9804
LUT	3314	2980	27075	14670
Clock (ns)	8.20	8.20	8.20	8.20
Latency (clock cycles)	261	179	94	95
Interval (clock cycles)	262	180	2	4

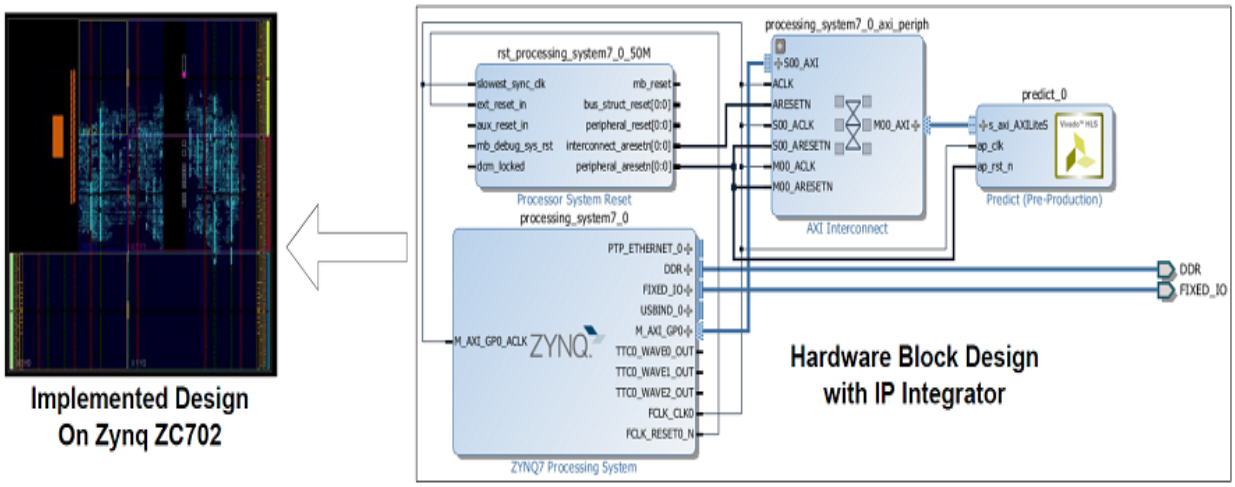

Figure-4. Gas identification hardware design.

Hardware design using IP Integrator

The created design using IP Integrator is shown in Figure-4. It is worth mentioning that when running the block and connection automation in IP integrator, two extra IPs are automatically added to the design (Processor system reset and AXI Interconnect) and all interconnections between different blocks are made. The implemented design can be seen in Figure-3 where all IPs are implemented on the PL and are represented with the blue color while the PS is physically present and represented by the amber color. Details about Vivado IP Integrator can be found in (Xilinx UG994, 2014).

Software design using SDK

The required application is created to get the basic settings and initialization of the platform including the universal asynchronous receiver/transmitter (UART) to print results in the terminal of SDK. The necessary C code is written to read/write data from/to the HLS core implemented on the PL. The communication with the hardware present in the PL is performed by calling some read and write data functions that exist in driver files which were automatically created and exported for various OS including Linux and the lightweight Standalone OS. Details about SDK can be found in (Xilinx UG898, 2014).

CONCLUSIONS

A solution for a HW/SW co-design approach on the Zynq SoC is presented in this paper for an innovative gas identification system based on PCA and DT classifier. Vivado HLS, IP Integrator and SDK are used to create the IP cores for hardware acceleration, to design the hardware and to write the required software respectively. The tools provide various optimization techniques and different implementation solutions for the interconnection between the processing system and the programmable logic of the Zynq such as AXI Stream or AXI Master that should be explored in the future. The gas identification system can be further improved by exploring the implementation of other classifiers or an ensemble of classifier on the Zynq SoC. Using the Zynq platform provide the designer with more flexibility allowing him to control the accelerated part on the hardware using a software running on the ARM processor. The results are promising knowing that resources on the PL have not been fully utilized and only one core of the possessor has been used in the PS. It's planned to use the Zynq SOC to integrate the gas identification system in a larger system that will take the form of a multi sensing platform to monitor gas mixture, temperature and transmit data wirelessly using Radio-frequency identification (RFID) technology.

ACKNOWLEDGEMENTS

This paper was made possible by National Priorities Research Program (NPRP) grant No. 5 – 080 – 2 – 028 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

REFERENCES

[1] Benrekia F., Attari M. and Bouhedda M. 2013. Gas sensors characterization and multilayer perceptron (mlp) hardware implementation for gas identification using a field programmable gate array (fpga). Sensors, Vol. 13, No. 3, pp. 2967–2985.

[2] Brahim Belhouari S., Bermak A., Wei G. and Chan P. 2004. Gas identification algorithms for microelectronic gas sensor. Proceedings of the 21st Instrumentation and Measurement Technology Conference, Vol. 1, pp. 584–587.

[3] Dobai R. and Sekanina L. 2013. Image filter evolution on the xilinx zynq platform. NASA/ESA Conference on Adaptive Hardware and Systems, pp. 164–171.

[4] Far A. B., Flitti F., Guo B. and Bermak A. 2009. A bio-inspired pattern recognition system for tin-oxide gas sensor applications. Sensors Journal, IEEE, Vol. 9, No. 6, pp. 713–722.

[5] Fenlason J. and Stallman R. 1997. GNU gprof: the GNU profiler. Manual, Free Software Foundation Inc.

[6] Guo B., Bermak A., Chan P. C. and Yan G. Z. 2007. An integrated surface micromachined convex microhotplate structure for tin oxide gas sensor array. Sensors Journal, IEEE, Vol. 7, No. 12, pp. 1720–1726.

[7] Gutierrez-Osuna R. 2002. Pattern analysis for machine olfaction: a review. Sensors Journal, IEEE, Vol. 2, No. 3, pp. 189–202.

[8] Han Y. and Oruklu E. 2014. Real-time traffic sign recognition based on zynq fpga and arm socs. International Conference on Electro/Information Technology, IEEE, pp. 373–376.

[9] Kim E., Lee S., Kim J. H., Kim C., Byun Y. T., Kim H. S. and Lee T. 2012. Pattern recognition for selective odor detection with gas sensor arrays. Sensors, Vol. 12, No. 12, pp. 16 262–16 273.

[10] Li Q. and Bermak A. 2011. A low-power hardware-friendly binary decision tree classifier for gas identification. Journal of Low Power Electronics and Applications, Vol. 1, No. 1, pp. 45–58.

[11] Monson J., Wirthlin M. and Hutchings B. L. 2013. Implementing high-performance, low-power fpga-based optical flow accelerators in c. 24th International Conference on Application-Specific Systems, Architectures and Processors, IEEE, pp. 363–369.

[12] Ng K. T., Chen H. T., Boussaid F., Bermak A. and Martinez D. 2009. A robust spike-based gas identification technique for SnO2 gas sensors. International Symposium on Circuits and Systems, IEEE, pp. 553–556.

[13] Ng K. T., Guo B., Bermak A., Martinez D. and Boussaid F. 2009. Characterization of a logarithmic spike timing encoding scheme for a 4×4 tin oxide gas sensor array. Sensors, IEEE, pp. 731–734.

[14] Ng K. T., Guo B., Martinez D., Boussaid F. and Bermak A. 2008. A 4×4 tin oxide gas sensor array based on spike sequence matching. 2nd International Conference on Signals, Circuits and Systems, pp. 1–5.

[15] Russell M. and Fischaber S. 2013. Opencv based road sign recognition on zynq. 11th International Conference on Industrial Informatics, IEEE, pp. 596–601.

[16] Shi M., Bermak A., Chandrasekaran S., Amira A. and Brahim-Belhouari S. 2008. A committee machine gas identification system based on dynamically reconfigurable fpga. Sensors Journal, IEEE, Vol. 8, No. 4, pp. 403–414.

[17] Xilinx Inc. 2014. Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator. UG994, v2014.3, October. Available: http://www.xilinx.com/support/documentation/sw_manuals /xilinx2014_3/ug994-vivado-ip-subsystems.pdf [Accessed March 31, 2015]

[18] Xilinx Inc. 2014. Vivado Design Suite User Guide: Embedded Processor Hardware Design. UG898, v2014.3, October. Available: http:// www.xilinx.com/support/documentation/sw_manuals/ xilinx2014_3/ug898-vivado-embedded-design.pdf [Accessed March 31, 2015]

[19] Xilinx Inc. 2014. Vivado Design Suite User Guide: High-Level Synthesis. UG902, v2014.3, October. Available: http://www.xilinx.com/support/documentation/sw_manuals/xilinx2014_3/ug902-vivado-high-level-synthesis.pdf [Accessed March 31, 2015]

[20] Xilinx Inc. 2015. Zynq-7000 All Programmable SoC: Technical Reference Manual. UG585, v1.8.1, October 2014. Available: http://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf [Accessed March 31, 2015]