

SOC Integration for Video Processing Application

Chan Boon Cheng¹, Asral Bahari Jambek²

^{1,2}School of Microelectronic Engineering, Universiti Malaysia Perlis, Perlis, Malaysia

Article Info

Article history:

Received Oct 17, 2018

Revised Nov 19, 2018

Accepted Dec 14, 2018

Keywords:

Terasic TRDB-D5M

Video Processing

SDRAM Controller

Greyscale

Cyclon II EP2C70 FPGA

ABSTRACT

Video processing is an additional system that can improve the functionality of video surveillance. Integration of a simple video processing system into a complete camera system with a field-programmable gate array (FPGA) is an important step for research, to further improve the tracking process. This paper presents the integration of greyscale conversion into a complete camera system using Nios II software build tools for Eclipse. The camera system architecture is designed using the Nios II soft-core embedded processor from Altera. The proposed greyscale conversion system is designed using the C programming language in Eclipse. Parts of the architecture design in the camera system are important if greyscale conversion is to take place in the processing, such as synchronous dynamic random-access memory (SDRAM) and a video decoder driver. The image or video is captured using a Terasic TRDB-D5M camera and the data are converted to RGB format using the video decoder driver. The converted data are shown in binary format and the greyscale conversion system extracts and processes the data. The processed data are stored in the SDRAM before being sent to a VGA monitor. The camera system and greyscale conversion system were developed using the Altera DE2-70 development platform. The data from the video decoder driver and SDRAM were examined to confirm that the data conversion matched greyscale conversion formulae. The converted data in the SDRAM correctly displayed the greyscale image on a VGA monitor.

*Copyright © 2018 Institute of Advanced Engineering and Science.
All rights reserved.*

Corresponding Author:

Chan Boon Cheng,

School of Microelectronic Engineering,

Universiti Malaysia Perlis,

Perlis, Malaysia.

Email: chanbc91@gmail.com

1. INTRODUCTION

Video processing is an important step for improving the functionality of video surveillance. A highly accurate video processing system is able to track a specific target with minimal error. However, a simple video processing system test is needed in the early stages of video processing system design. The test system is able to confirm that the designed camera system can be integrated into the video processing system. The proposed video processing system is a simple greyscale conversion system using Nios II software build tools for Eclipse on the Altera DE2-70 development platform. Proposed simple greyscale conversion system is able to confirm the functionality of designed architecture before proceed with major video processing algorithm integration. External peripherals, i.e., a TRDB-D5M camera and a VGA monitor, are needed to complete the video processing system [1,3]. The TRDB-D5M camera is needed to capture the image or video and deliver the data to the video decoder driver for further processing. The VGA monitor is used to display the resulting processed data from the SDRAM [2].

The camera system was designed using Quartus II version 13.0 software and implemented on the Altera DE2-70 development platform [4]. An Altera Cyclone II 2C70 FPGA device was chosen as the core

for the system. In order to build a complete architecture, the Quartus II system-on-programmable-chip (SOPC) builder was used to design the camera system architecture [5,6]. The external peripherals included in the camera system design were a TRDB-D5M camera and a VGA monitor [7]. The camera system represents the main step in designing the video processing system. A good camera system reduces the errors occurring in the video processing system. Figure 1 shows the block diagram for the camera system.

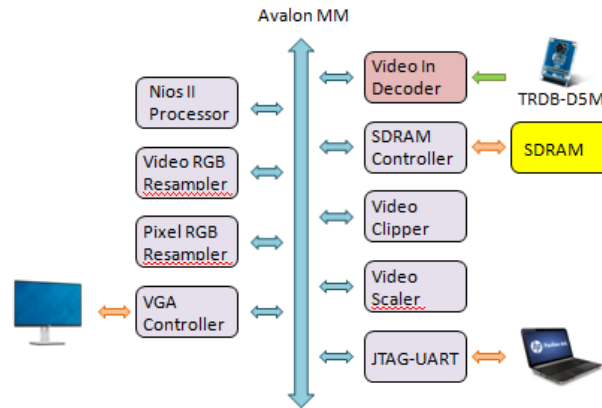


Figure 1. Block diagram for camera system

The video decoder is an important peripheral in the video processing system [8]. The captured image or video data must be in the binary form of the RGB format to enable further processing steps [14]. The first step of the capture process is to convert the captured image or video, in a specific format size and in RGB format, via the video decoder driver, prior to further processing by another driver. A video clipper driver and a video scaler driver are used to convert the captured image or video data into a suitable format size before the main processing takes place. A video RGB resampler converts the data between 24-bit and 16-bit formats [8]. The different formats are suitable for different type of video processing systems. Using a suitable number format increases the processing speed and prevents noise. The converted data are temporarily stored in static random-access memory (SRAM) through the SRAM controller [9], before being stored in SDRAM, displayed on the monitor or submitted for video processing.

The processing algorithm is the most important part of the video processing system, since retrieval of incorrect data will cause noise or corruption of data. The designed algorithm uses the C/C++ language in the Nios II software build tool for Eclipse [10]. The complete algorithm design is downloaded into the Nios II processor through the Joint Test Action Group universal asynchronous receiver/transmitter (JTAG-UART) core. The video data are retrieved from the SRAM, undergo some binary data processing and are stored in the SDRAM before running a VGA monitor display conversion step.

The SDRAM has another important role in storing the converted data from the drivers [4,6,11]. The SDRAM chip on the Altera DE2-70 board is able to store 8 Mbytes of data and the memory is organized in the form $1M * 16 \text{ bits} * 4 \text{ banks}$ [4,6]. The SDRAM controller is included in the architecture design of the camera system in order to access the SDRAM chip [4,6,9]. Signals will be generated from the SDRAM controller to deliver read or write instructions from the Cyclone II 2C70 processor to the SDRAM chip. In the greyscale conversion video processing design, the processing data occupied less than 8 Mbytes. Therefore, only one SDRAM chip was used in this design. Future processing of large quantities of data will make full use of the SDRAM chip on the Altera DE2-70 board, which is in fact two SDRAM chips each storing 16 Mbytes of data [4,6].

The final part of the video processing system is displaying the processing output. The data are retrieved from SDRAM and undergo some format conversion before being displayed on the VGA monitor. The pixel RGB resampler driver plays an important role in RGB format conversion [8]. The data are converted to a suitable RGB format before being sent to the video scaler driver and VGA controller. The video scaler driver changes the resolution of the data into a suitable format size while the VGA controller enables data display by creating the timing signals required by the VGA monitors. A more detailed explanation of the data flow will be given in section III.

In this article, studies using several different video processing designs and application platforms are surveyed. The surveyed papers include: implementation of Laplacian video processing using the Virtex-6 ML605 Evaluation Kit [11], image encryption operations implementation on the Cyclone II EP2C35 FPGA [12], FPGA implementation for image processing algorithms using the Xilinx System Generator [13], image encryption system implementation using the Virtex-5 VC5VLX110T FPGA [14] and optical seam tracking system implementation on an FPGA [15]. In section II, existing video processing design and application

platforms are discussed. Comparisons are made, and advantages and disadvantages are discussed at the end of the section. Section III presents a full description of the proposed video processing method. In section IV, the output of the design and details of the data conversion are described.

2. LITERATURE REVIEW

A simple video processing test on the complete camera system is important to prevent errors occurring in the architecture design. In this paper, a simple greyscale conversion is chosen as the main test for the complete camera system design. The existing greyscale-related usage implementation on FPGA presented in [11] provides a good reference for greyscale-related usage. The author used the greyscale video as a benchmark video to perform Laplacian filtering. The chosen pixel number of the video was $40 * 40$ pixels, and the system was implemented using the Virtex-6 ML605 as FPGA. The designed system frame rate was up to 10,000 frames per second, at an operating frequency of 0.31 MHz. A DDR3 SDRAM was chosen as the main memory for storing the preprocessed and processed videos. Figure 2 shows the block diagram for the system configuration design to the interrupt, with an ML605 board.

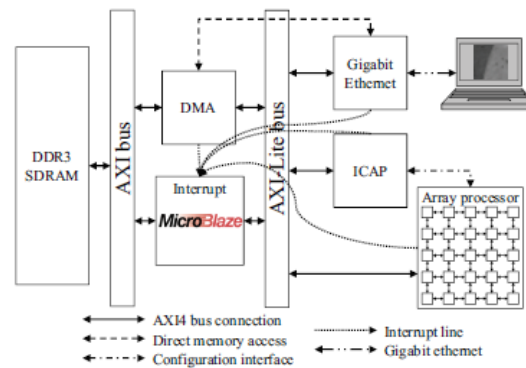


Figure 2. Block diagram for system configuration design to interrupt, with ML605 board

The author of [12] proposed an image encryption operation implemented on a Cyclone II EP2C35 FPGA. The image encryption is performed based on two types of architecture design using 14-bit or 8-bit formats. The author used a greyscale image as a benchmark image to perform the image encryption operation. Two greyscale images with sizes $128 * 128$ were used, and the memory storage was performed on M4K RAM. The total number of logic elements used in this process was 11,675 and total memory usage was 262 or 144 bits. A flow diagram of the image encryption system process is shown in Figure 3.



Figure 3. Flow diagram of the image encryption process

Various types of image processing algorithms implemented on FPGA are proposed by the author in [13]. The author used Xilinx System Generator for MATLAB to perform the simulation processes. The image processing algorithms included image enhancement, threshold, contrast stretching, edge detection and boundary extraction for greyscale and colour image algorithms. A greyscale image was used as a benchmark image for some of the image processing and for the output. The image size for processing was between $238 * 212$ pixels and $700 * 500$ pixels.

The author in [14] proposed a cryptographic algorithm using a matrix approach implemented on a Virtex-5 VC5VLX110T FPGA. The system used greyscale and RGB colour images to carry out the encryption process. The encryption system was configured with a 125-MHz clock and implemented on three types of architecture designs, i.e., 8-bit, 16-bit and 24-bit systems. A few steps of the matrix function process are taken before the final encryption process. Figure 4 shows the steps required for performing the encryption process.

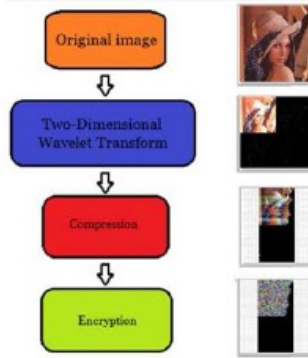


Figure 4. Steps for performing the encryption process

Optical seam tracking with parallel image processing implemented on a camera system is proposed by the author in [15]. The proposed system is implemented on a Xilinx Spartan-6 FPGA and performs in a real-time application. The real-time application processing system must use a greyscale image as the benchmark before proceeding with further processing steps. The processing frame rate of the system is up to 1,000 frames per second and the clock rate is set to 48 MHz. Figure 5 shows the flow diagram for the processing system.

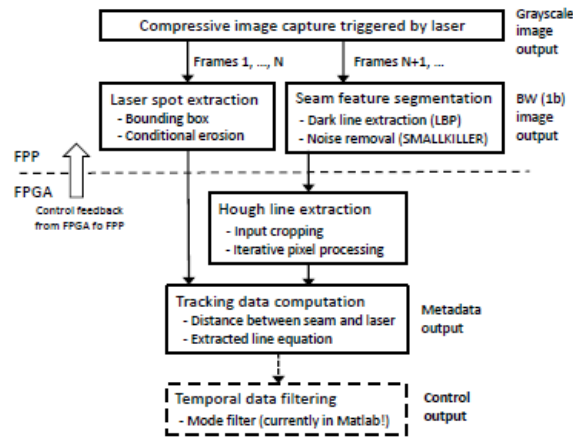


Figure 5. Flow diagram for the processing system

Table 1 shows a comparison between the different types of greyscale processing applications on various types of FPGA. All these systems have the common feature that the processing algorithm relates to a greyscale application. Each of the greyscale processing applications is different, but some of the usages are similar. For example, the system in [11] used a greyscale video as a benchmark video, those in [12,14,15] used the greyscale image as benchmark image and in [13] the greyscale image is an output of the processing system. The systems in [11-14] are of the simulation type, where the process is run using a benchmark video or image, while the system in [15] is a real-time application system.

The systems in [13] and [14] do not have any memory usage due to the fact that the process is running on the FPGA company-provided simulation software. This process is able to test the algorithm design before applying it on a real FPGA platform. The simulation software chosen in [13] is Xilinx System Generator for MATLAB, while in [14] it is MicroBlaze. Both of these are able to test the processing algorithm design in order to reduce the code upload time to the FPGA.

The greyscale image enhancement in [13] is a suitable reference for the proposed method discussed in section III. The resize step and the preprocessing and post-processing pixel conversion steps discussed provide a relevant reference for image resizing before running the greyscale conversion process. An architecture design with different bit formats discussed in [12] and [14], is used in the proposed greyscale conversion system. The different bit formats in the conversion setting will affect the output and the processing speed of the system. More detail on the greyscale conversion system design is given in section III.

Table 1 Comparison of discussed systems

Features	[11]	[12]	[13]	[14]	[15]
Year	2013	2014	2013	2014	2015
Development Platform	Virtex-6	Cyclone II EP2C35 FPGA	Xilinx FPGA	Virtex-5 FPGA	Xilinx Spartan6 FPGA
Function	Laplacian Filtering	Image Encryption	Image Segmentation	Image Encryption	Optical Seam Tracking
Function Application	Simulation	Simulation	Simulation	Simulation	Real Time Application
Greyscale Usage	Benchmark Video	Benchmark Image	Output	Benchmark Image	Benchmark Image
Hardware Language	Not Stated	Not Stated	HDL	Matrix Operation	Not Stated
Maximum Frame Rate (fps)	10000	Not Stated	Not Stated	Not Stated	1,000
Clock Rate (MHz)	100	50	Not Stated	125	48
Memory Usage	DDR3-SDRAM	M4KRAM	Not Stated	Not Stated	BRAM
Software	ModelSim, MATLAB	Quartus II 7.2 ISE	Xilinx System Generator, MATLAB	Microblaze	MATLAB

3. RESEARCH METHOD

In section III, the camera system architecture design is briefly discussed and the greyscale conversion algorithm design [13,15] is explained, giving details of the data extraction and data processing. The camera system design starts with capturing the image or video using the TRDB-D5M camera. The block diagram for the architecture design is shown in Figure 1. The architecture design was developed using the Qsys tools provided in the Quartus II version 13.0 software.

The first step in the capture process is putting the video through the decoder driver to convert the captured image or video into a 2592 * 1944 format size with colour (8 (bits) * 1 (plane)), in RGB format. The data are converted to a 320 * 240 format size via the video clipper driver and video scaler driver. After the data are converted, the video RGB resampler further converts the data between 24-bit and 16-bit formats. These conversions are required because processing is faster with the 320 * 240 format size and the 16-bit format. Figure 6 shows the architecture design flow for the video during the conversion process.

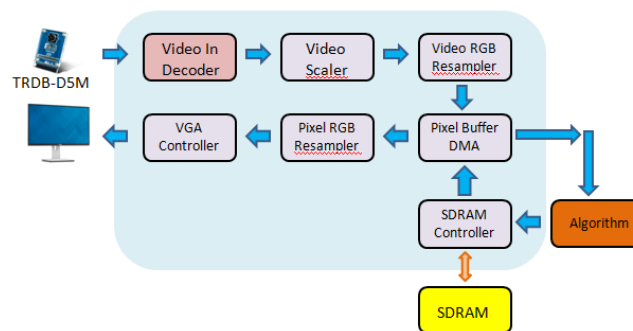


Figure 6. The architecture design flow for the video during image data conversion

In order to be suitable for the greyscale conversion system, the original data from the pixel buffer DMA module are extracted and processing is performed before saving to SDRAM. The algorithm design is implemented in Nios II SBT for Eclipse. The SDRAM has an important role in storing the preprocessing and processed data with a 320 * 240 format size and a 16-bit format. Without the processing system, the data are stored directly into SDRAM from the pixel buffer DMA module. When the data are correctly stored, the pixel buffer DMA will read the data and send the image stored in the memory to the VGA controller. At the

same time, the image will undergo format conversion from 16 bits to 30 bits via the pixel RGB resampler driver. When the VGA controller receives the data, it will create a timing signal compatible with the VGA monitor attached to the VGA port on the DE2-70 board, to enable display. The architecture design for data storage and display is shown in Figure 6. The blue arrow shows the flow of data between the memory, the DMA driver and the VGA controller.

The completed architecture design is loaded onto the Altera DE2-70 board through the programmer. When the design is confirmed as working well, by capturing the video using the TRDB-D5M camera and displaying it on the VGA monitor, the greyscale conversion algorithm design can proceed.

The greyscale conversion system is designed in the C language using Nios II software build tools for Eclipse. In order to perform greyscale conversion of the captured video, the original 16-bit binary image data needs to be converted depending on the formulae (1) [16]. The converted binary data are then saved into SDRAM before running the resampler and displaying the output on the VGA monitor. RGB values indicate different locations within the 16 bits of binary data. The data locations are shown in Figure 7.

$$\text{Greyscale} = \frac{R+G+B}{3} \quad (1)$$

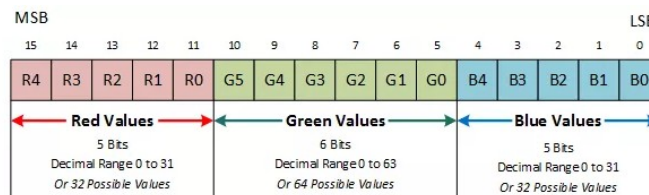


Figure 7. Binary data locations of RGB

In order to ensure binary data converted equal to formulae, data for red values must move 11 bits to the right, using “AND” to multiply by 0x1f in order to replace blue values with red values. Data for green values will then shift left by 6 bits and using “AND” to multiply by 0x1f makes the sixth bit of the green values 0, and bits 1-5 of the green values is remained and blue values using “AND” to multiply by 0x1f. In the final step of the design, the red, green and blue values us sum up and divided by 3 and shift to the origin location of each values. The C programming coding for the data conversion is shown in Figure 12.

4. RESULTS AND ANALYSIS

In this section, the output of the proposed architecture of the camera system and the algorithm for greyscale conversion are discussed in detail. The architecture was loaded onto the Altera FPGA board after the design was completed. The initialization of the board is shown in Figure 8. The architecture includes a user trigger switch, designated switch 0. Trigger pin switch 1 is used to reset the camera capture recording function to prevent data corruption displaying on the VGA monitor. Figure 9 shows the initialization of the architecture design on the VGA monitor display without triggering switch 0.

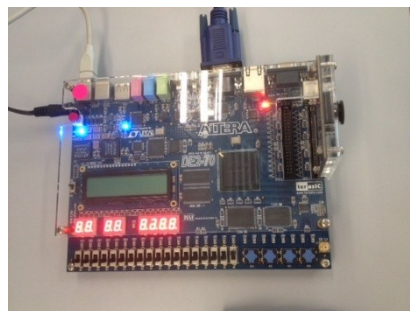


Figure 8. Initialization of camera system on Altera DE2-70

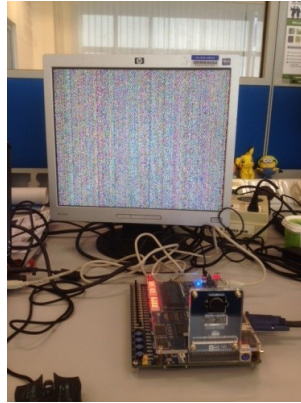


Figure 9. Initialization of camera system on VGA monitor

When switch 1 is triggered, the camera system is able to perform live capture and display on the VGA monitor. When the video captured by the camera is correctly displayed on the VGA monitor, the architecture design is considered to be complete. Figure 10 shows the captured result, correctly displayed on the VGA monitor.

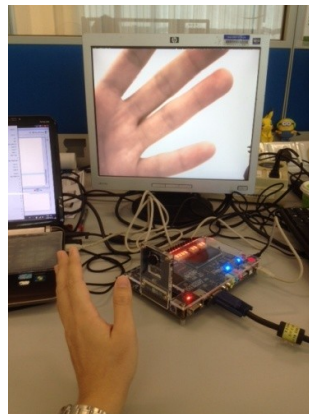


Figure 10. Video captured by camera, correctly displayed on VGA monitor

In order to ensure correct data flow between the architecture modules, SignalTap II is used to analyse the data generated and the locations for saving. Figure 11 shows the generated output of SignalTap II. SDRAM Write Enable (WE) is a one-bit signal to inform SDRAM to perform the write data action. The video DMA/slave_writedata is the last module, which generates 32 bits of image data before writing to SSRAM, while SDRAM_wire_dq is the final storage location of the preprocessing or processed data before display on the VGA monitor. Figure 11 shows that the data are correctly stored in SDRAM. Since the SDRAM is able to store 16 bits of data at once, the 32 bits of data generated by the video DMA will be split into two 16-bit items. The waveform data generated by SignalTap II are shown in hexadecimal form.

The highlighted columns in Figure 11 show that the data are correctly stored in SDRAM. As an example, the data highlighted in blue in the video DMA show that the video DMA module has generated 32 bits of hexadecimal data: 00037E36h. When the data are saved in 16-bit form in SDRAM via the SDRAM controller, the 32 bits of hexadecimal data 00037E26h are split into two 16-bit hexadecimal data items, i.e., 7E36h and 0003h. SDRAM_dq shows that the data are split correctly and stored in the 16-bit SDRAM module. The same situation occurs for the data highlighted in green. The data between the blue and green rectangles in the video DMA are useless data caused by processing delays, which are not saved into SDRAM.

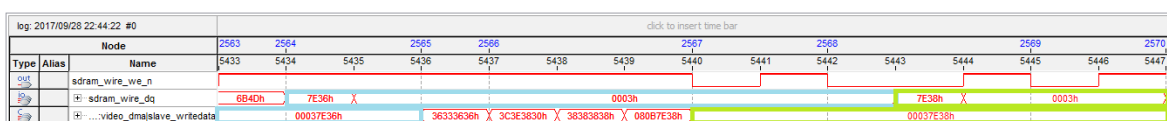
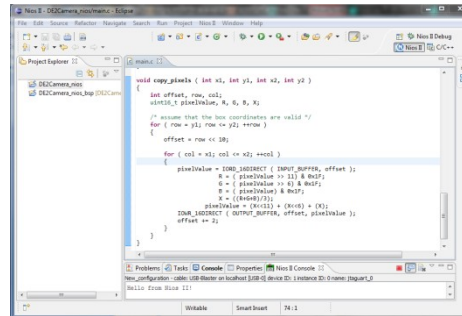


Figure 11. SignalTap II analysis result

Greyscale conversion is the last step of the proposed system. The greyscale conversion methods refer to the method discussed in section III, i.e., $\text{greyscale} = (R + G + B) / 3$. The data are read from the SSRAM, undergo data processing and are saved into SDRAM. Figure 12 shows the C programming steps which perform the greyscale conversion with 16-bit data processing, as designed using Nios II software build tools for Eclipse.



```

void copy_pixels ( int x1, int y1, int x2, int y2 )
{
    int offset, row, col;
    int16_t pixelValue, R, G, B, X;
    /* assume that the line coordinates are valid */
    for ( row = y1; row <= y2; row++)
    {
        offset = row << 30;
        for ( col = x1; col <= x2; col++)
        {
            pixelValue = E0M_READ32 ( INPUT_BUFFER, offset );
            R = ( pixelValue >> 11 ) & 0x00FF;
            G = ( pixelValue >> 5 ) & 0x00FF;
            B = ( pixelValue & 0x00FF );
            X = ( (R+G+B) / 3 );
            pixelValue = (X << 16) + (X);
            E0M_WRITE32 ( OUTPUT_BUFFER, offset, pixelValue );
            offset += 4;
        }
    }
}

```

Figure 12. The C code for greyscale conversion

The greyscale conversion processed data replace the data in SDRAM. The data are retrieved and pass through the RGB resampler module before being sent to the VGA monitor through the VGA controller. Figure 13 shows the result of the greyscale conversion, where the camera system output is replaced by the greyscale image output on the VGA monitor. The total number of logic element used in this process is 4258 of 68416 which is 6% of total logic element. Total registered used is 2807 and total thermal power dissipation is 215.06mW.

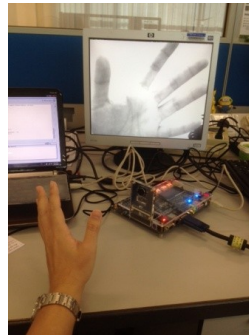


Figure 13 Greyscale conversion output

5. CONCLUSION



In this paper, a simple greyscale conversion method is implemented on a complete camera system. The images were captured using a TRDB-D5M camera and displayed on a VGA monitor. The results show that the camera system works perfectly. A simple video processing greyscale conversion, designed using the C programming language in Eclipse, was implemented on the camera system. The results demonstrated that the greyscale conversion design was able to retrieve and replace the data in the camera system. The results, as seen on the VGA monitor, show that simple processing can be implemented on the camera system. The logic element used is less compared to the existing works. Therefore, the results of the proposed method form a good basis for future moving-object tracking studies.

REFERENCES

- [1] Terasic, "TRDB-D5M: Terasic D5M Hardware Specification", April 2008.
- [2] C. Li and W. Chen, "A novel FPGA-based hand gesture recognition system," *Journal of Convergence Information Technology*, vol. 7, no. 9, pp. 221–229, 2012.
- [3] Nai-Jian Wang "A Real-time Multi-face Detection System Implemented on FPGA", 2012 IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS 2012) November 4-7, 2012
- [4] ALTERA: "DE2-70 Development and Education Board: User Manual" v1.08, 2009.
- [5] ALTERA: "SOPC Builder: User Guide", v1.0, Dec., 2010.
- [6] ALTERA: "Quartus II Handbook" v14.1, Dec. 2014.
- [7] Terasic, "TRDB-D5M: Terasic D5M Hardware Specification", April 2008.
- [8] ALTERA: "Video IP Cores for Altera DE-Series Boards", October 2015.
- [9] X. Ma, M. Li, H. Wang, and I. Gong, "Design and implementation of video image encryption system based on FPGA," in *Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on*,

- 2012, pp. 68-72.
- [10] ALTERA: "Nios II Software Developer's Handbook", v13.1, Jan., 2014.
- [11] Z. Ping Ang, A. Kumar, Y. Ha, "High speed video processing using fine-grained processing on FPGA platform," Proceedings of the IEEE 21st Annual International Symposium on Field-Programmable Custom Computing Machines, 2013, pp. 85-88.
- [12] Rajagopalan, S., H.N. Upadhyay, J.B.B. Rayappan and R. Amirtharajan, "Dual cellular automata on FPGA: An image encryptors chip," Res. J. Inform. Technol., 6: 223-236, 2014.
- [13] Neha P.Raut, Prof A.V.Gokhale "FPGA implementation of image processing algorithms using Xilinx System Generator", IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) Volume 2, Issue 4 (May- June 2013), PP 26- 36, e-ISSN: 2319-4200, p-ISSN No.:2319-4197.
- [14] Ramirez-Torres, M., Murguia, J., Mejia-Carlos, M.: Fpga implementation of a reconfigurable image encryption system. In: ReConFigurable Computing and FPGAs (ReConFig), 2014 International Conference on. pp. 1-4. IEEE (2014)
- [15] Tero Santti, Jonne K. Poikonen, Olli Lahdenoja, Mika Laiho, Ari Paasio (2015); Online seam tracking for laser welding with a vision chip and FPGA enabled camera system, IEEE International Symposium on Circuits and Systems (ISCAS), May 2015, 1985 - 1988.
- [16] A. Güneş, H. Kalkan and E. Durmuş, "Optimizing the color-to-grayscale conversion for image classification", Signal, Image and Video Processing, Publisher Springer London, (2015) Oct., pp. 1-8, 29.

BIOGRAPHIES OF AUTHORS

	<p>Chan Boon Cheng is currently an M.Sc. by research student at School of Microelectronic Engineering, University Malaysia Perlis. In 2015, he completed his B.Eng. degree in Electronic Engineering at University Malaysia Perlis, Malaysia. He has served Sensmaster SDN.BHD as graduate trainee and failure analysis assistance in 2014. His research interest includes object tracking algorithm and implementation using SOC devices.</p>
	<p>Associate Professor Dr. Asral Bahari Jambek is a member of the School of Microelectronics Engineering, Universiti Malaysia Perlis (UniMAP), and was a Programme Chairperson for the Electronics Engineering Degree Programme, UniMAP. He has more than 15 years experience in integrated circuit and system design in both the industry and academic sectors, and has been involved at various levels of VLSI design such as transistor modelling, digital circuit design, analogue circuit design, logic synthesis and physical place and route, architecture design and algorithm development. Currently, he is actively researching new techniques to minimize power consumption in portable system-on-chip design. His research interests include integrated circuits and systems design, digital signal processing (DSP), low power algorithms and architectures design, and image and video processing.</p>