Asian Journal of Information Technology 5 (9): 976-984, 2006
© Medwell Online, 2006

Design of Hardware Accelerators for Content Based Video Indexing

[1,2]A. Ben Abdelali, [3]A. Mtibaa, [1]E. Bourennane and [2]M. Abid
[1]Laboratory LE2I, University of Burgundy, 21000 Dijon, France
[2]Laboratory C.E.S, National Engineering School of Sfax,(ENIS), Sfax, Tunisia
[3]Laboratory EμE, Faculty of Sciences of Monastir, 5019, Monastir, Tunisia

Abstract: Content based video indexing is currently employed in variety of new applications, including video surveillance, Access to real time incoming video, intelligent transmission of video data, etc. Several methods aiming at automating this time and resource consuming process have appeared in literature. They employ several content analysis and feature extraction techniques including, typically, signal and image processing techniques to analyze color, texture, form, sound, etc. Implementing such applications on a general purpose computer can be easier, but not sufficient to support the complexity and the performance needs of some new applications. The use of application specific hardware permits to offer much greater speed and performance. In this study we focus on hardware implementation of content based video indexing techniques by using the FPGA technology. We aim to propose hardware modules that can satisfy requirements of applications under hard constraints, such as real time applications and applications of high complexity. We represent tow examples of micro-architectures related to the dominant colors descriptor and the compact color descriptor. The synthesis and simulation results for the provided solutions are also given.

Key words: Hardware accelerator, video indexing, FPGA, computer color descriptor, dominant colors descriptor

INTRODUCTION

Video understanding and semantic information extraction represent an important step towards more efficient manipulation and retrieval of visual media. The increased availability and usage of digital video lead to a need for automated video content analysis techniques. Several methods aiming at automating this process, which is time and resource consuming, have appeared in literature[1-4]. A standard called MPEG-7 has also been elaborated[5,6]. The standard specifies a set of visual descriptors that include color, texture, shape and motion[7-9]. Most standardized descriptors are low-level arithmetic ones, chosen so as to ensure their usefulness in wide range of possible applications[10]. Low-level features serve as a basis for a number of analysis algorithms for higher level features (such as structuring or classification) or they can be used directly for similarity matching.

Low level visual features extraction defines the process of creating a descriptor from a given visual media item. This process may include complex algorithms that can be also combined in a specific scheme to be used as input for the extraction of higher level features. High level semantic content description is based on a combination of low level techniques and other content interpretation algorithms for classification, recognition or summarization[2,11-15]. We obtain complex indexing systems that combine a large number of techniques. These systems exploit multimodal and multi level content analysis techniques for automatic video indexing[16,17,13,1,18]. Fig. 1 gives an example of a structure of a complex content based multimodal indexing system. This structure can take different configurations depending on the application needs. In fact, the used algorithms can change relatively to the program type (sport, films, etc.), the treatment objectives (segmentation, classification, summarization, identification, etc.) and the media types. Several content analysis and feature extraction techniques can be used. They include, typically, signal and image processing techniques to analyze color, texture, form, sound, etc.

Dedicated indexing tools and systems to help in searching, storing, filtering and managing the information not only work with data that has been previously stored, but also with live data being broadcasted through high-speed means such as digital cable or data used for surveillance. Several new applications scenarios exist for the content based video indexing systems[19-23]. These systems have demanding applications that can be driven by processing power, portability, performance, flexibility, cost, etc. Implementing such applications on a general purpose computer is not sufficient to support the complexity and performance need. The use of application specific hardware permits to offer much greater speed and performance. In[24,25] the authors suggest hardware

Corresponding Author: A. Ben Abdelali, Laboratory LE2I, University of Burgundy, 21000 Dijon, France

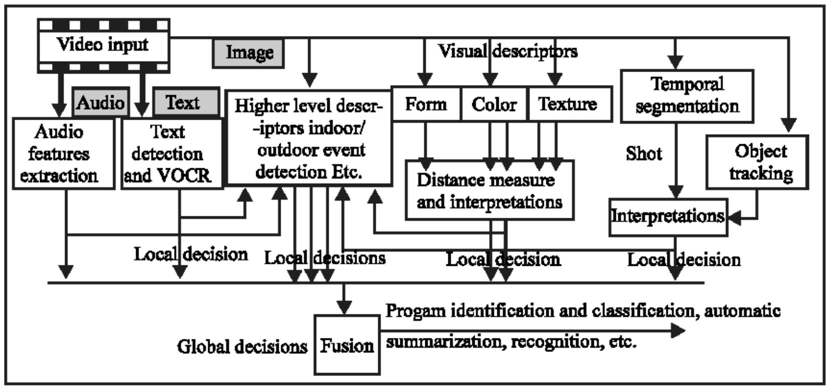


Fig. 1: Example of possible structure of a content based multimodal indexing system

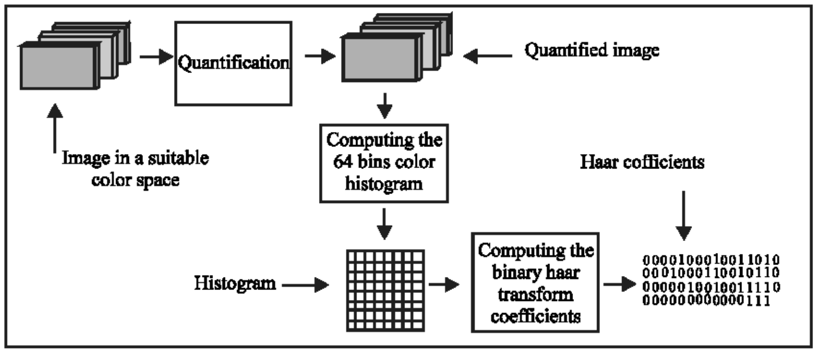


Fig. 2: The CCD extraction stages

implementation of a shape and a motion MPEG-7 descriptors. Specific hardware module for video indexing can be coupled to a general purpose or dedicated processor to accelerate the critical treatments and to support intensive parallel treatments[20,26].

The key issue in the design of new multimedia systems is to find a good balance between flexibility and high-processing power on one side, and area and energy-efficiency of the implementation on the other side. In this study the use of FPGA architectures can be very useful. These architectures are, today, the common devices for reconfigurable computing[27]. They permit to satisfy demanding applications by combining speed with flexibility. They are currently being used for the acceleration of a wide variety of applications in a large number of systems. In[20,26] the authors propose the use of reconfigurable hardware acceleration as a solution to implement a video-based driver assistance application which is based on MPEG-7. They propose to exploit dynamic hardware reconfiguration to ensure resources optimisation and flexibility.

We aim to design appropriate systems and hardware accelerators to support demanding video indexing applications.

ALGORITHMS SPECIFICATION

Specification of the Compact Color Descriptor (CCD): The CCD or compact color histogram descriptor which is a color descriptor in the MPEG-7 standard represents the global distribution of colors in an image or video. This descriptor can be used for very fast image and video retrieval[28]. It is obtained by applying the Haar wavelet transform to a color histogram and binary quantizing the resulting coefficients of the transform. The resulting descriptor is 63 bits long.

The feature extraction for the binary Haar histogram includes two principal stages Fig. 2. The first stage is to obtain a color histogram using a suitable color space (the HSV color space is recommended in the MPEG-7 standard) and quantization scheme. The second stage is the computation of the Haar transform coefficients. The second stage does not depend on the nature of the color space or the quantization table used in the first stage.

We use a 64-bins color histogram, but the technique is equally applicable to histogram for larger or smaller number of quantization bins. The quantified color centers are obtained by a uniform quantification using the RGB color space as in[29,30]. Each color component (R, G, B) is

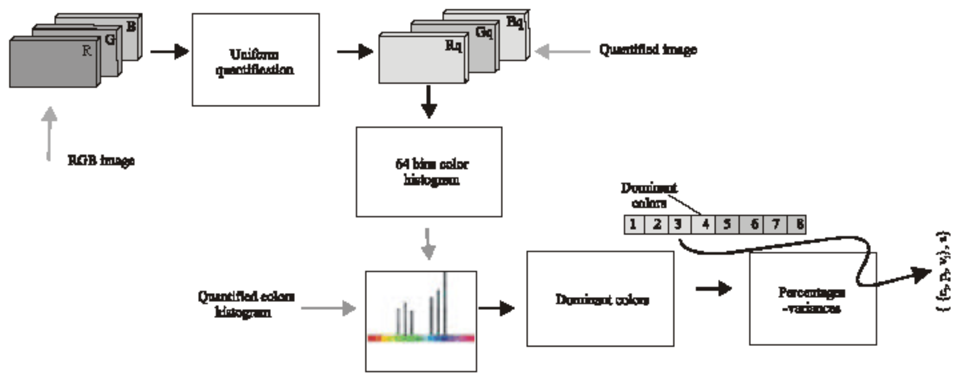


Fig. 3: The DCD extraction stages

composed of integers in the interval [0, 255]. The quantification is made uniformly in four centers (32, 96, 160, and 224). We obtain 64 possible quantified colors.

The histogram values corresponding to the 64 quantification centers are placed on a two dimensional 8-by-8 grid. The histogram values corresponding to the first 8 centers are placed in the first row and so on. The Haar transform coefficients are computed in a hierarchical way[31]. In the first level, the 8-by-8 grid of histogram values is divided into two halves vertically. The sum of the histogram values on the left half is subtracted from the sum of histogram values in the right half. The first Haar coefficient is quantified to 1 if the result is greater than 0 and is quantified to 0 otherwise. The process is then repeated recursively. In the second level, the left (right) half of the 8-by-8 grid is split horizontally into two 4-by-4 halves and the sum of histogram values in one half is subtracted from the other resulting in the second (third) Haar coefficient. The resulting second (third) Haar coefficient is then binary quantified. This is repeated recursively at the third, fourth, fifth, and sixth levels resulting in 4, 8, 16, and 32 coefficients respectively. Therefore, there are a total of 63 (1+2+4+8+16+32) binary Haar coefficients that form the compact descriptor.

Specification of the Dominant Color Descriptor (DCD): The dominant color descriptor provides a compact description of the dominant representative colors in an image. The feature descriptor consists of the representative colors, their percentages and optionally colors variances and spatial coherency of the dominant colors. The number of dominant colors can vary from image to other; a maximum of eight dominant colors can be used. The DCD can be defined for each object, regions, or the whole image by

$F = \{\{c_i, p_i, v_i\}, s\};$

c_i : representative color N° i.

p_i : percentage of c_i.

v_i : color variance of c_i.

s : Spatial coherency

In the method represented by Fig. 3, dominant colors are extracted from the colors histogram[29,30]. The representative colors are obtained by considering the most frequent colors. The input image is represented in the RGB color space. A uniform quantification is performed before calculating the histogram values. An other more complex method is proposed in[32] and adopted for the MPEG-7 part 3 (visual)[7,33]. In this study, the representative colors are computed from each image instead of being fixed in the color space, thus allowing the feature representation to be more accurate. In order to compute this descriptor, the colors present in a given image or region are first clustered by using the clustering method proposed in[34]. In this clustering, the pixel color values are vector quantized using a modified Generalized Lloyd Algorithm (GLA). Colors are represented in the perceptually uniform CIE LUV color space. The problem with this method is the high level of complexity especially when we extend the use of the dominant colors descriptor for video shots. Computing the dominant colors of a shot with the GLA clustering algorithm is not suitable[35]. The reason is that the number of pixels in a shot is too large; using the GLA will be very time-consuming. The method described in Fig. 3 can be well suitable for this caned of application especially with a high performance hardware implementation.

HARDWARE IMPLEMENTATION STUDY

In this study we describe the structures of the hardware implementation solutions of the Haar and the dominant colors blocks. We give, also, the synthesis and the simulation results of the presented solutions. The Simulation framework is represented by Fig. 4. The test

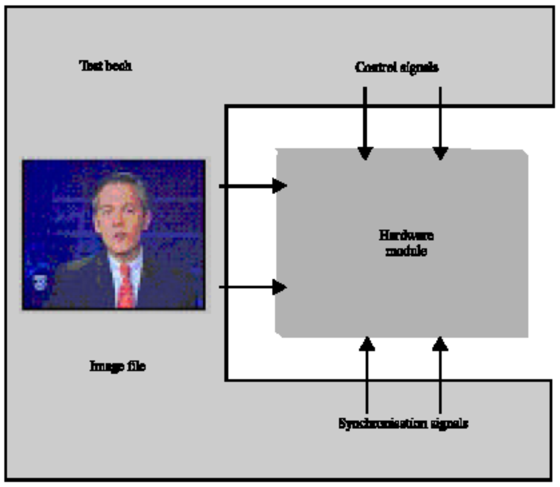
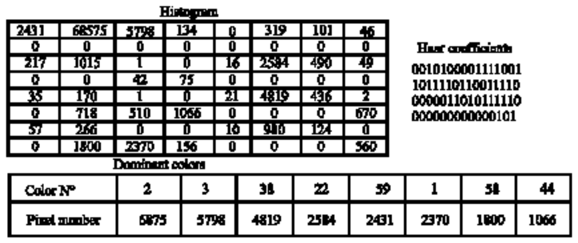


Fig. 4: Simulation framework



Histogram

2431	68575	5798	134	0	319	101	46
0	0	0	0	0	0	0	0
217	1015	1	0	16	2584	490	49
0	0	42	75	0	0	0	0
35	170	1	0	21	4819	436	2
0	718	510	1066	0	0	0	670
57	266	0	0	16	980	124	0
0	1800	2370	156	0	0	0	560

Haar coefficients

0010100001111001
1011110110011110
0000011010111110
0000000000000101

Dominant colors

Color N°	2	3	38	22	59	1	58	44
Pixel number	6875	5798	4819	2584	2431	2370	1800	1066

Fig. 5: MATLAB functional validation results

bench is based on the input frames and control signals. To verify the correctness of the RTL simulation results we refer to the MATLAB implementation results of the considered algorithms. Histogram, Haar coefficients and dominant colors results given by Fig. 5 correspond to the image shown in Fig. 4.

The synthesis results for the different implemented modules are obtained by " Xilinx Synthesis Technology (XST) tool» for the xcv300 FPGA family.

Hardware implementation of the haar transform: To calculate the Haar coefficients we use an iterative procedure which consists in calculating, initially, the coefficients using two values of the histogram Table: H32, H33, H34, H35…H62, H63. Next, we calculate the coefficients that use 4 values; in this step we do not carry out all the addition operations because we already done a part of these operations during the calculation of the previous coefficients (H32…H63). For example H16 is calculated using the additions made for H32 and H33. For the coefficients using 8 values we use the calculation carried out for the previous coefficients (H16..., H31). To be able to use the already calculated partial sums we must evaluate the Haar coefficients in the order represented by Table 1. To facilitate the coefficients

Table 1: Calculation order of the haar coefficients

h32, h33, h16	h34, h35, h17	h36, h37, h18	h38, h39, h19	h8, h9, h4
H40, h41, h20	h42, h43, h21	h44, h45, h22	h46, h47, h23	h10, h11, h5
H48, h49, h24	h50, h51, h25	h52, h53, h26	h54, h55, h27	h12, h13, h6
H56, h57, h28	h58, h59, h29	h60, h61, h30	h62, h63, h31	h14, h15, h7
H2 h3 h1				

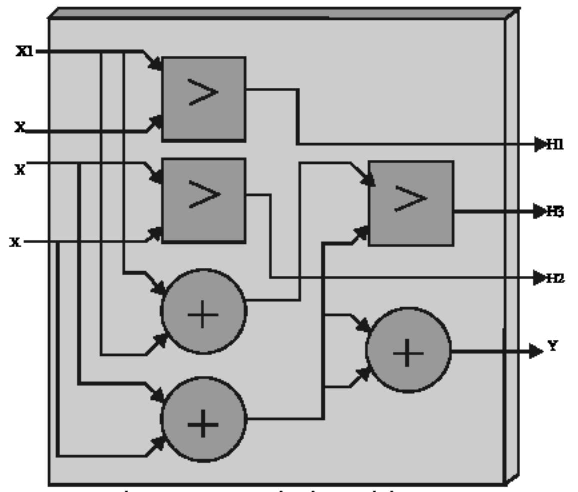


Fig. 6: Internal structure C-Block module

Table 2: Performance of the C-Block module

Slices	LUTs	Critical path
55	108	13.184 ns

calculation in this order, we have to reorganize the histogram values.

We demonstrate[36] that the calculation of the Haar coefficients can be done by using repetitively a function called C_3Haar task. The internal structure (C_Block) of this task is represented by Fig. 6. This task permit to calculate three Haar coefficients (h1, h2, h3) by using 4 values of the histogram (x1, x2, x3, x4). It produces also an intermediate sum (y) which will be used later to calculate other Haar coefficients by the same principle.

The calculation of the 63 Haar coefficients is carried out by using each time the C_3Haar task. From each 4 values of the reorganized histogram, we calculate three coefficients (2 using 2 values of the histogram Table and 1 using 4 values) and an intermediate sum. Each time that we have 4 intermediate sums, we apply the C_3 Haar procedure for them. We obtain 3 other Haar coefficients (2 using 8 values of the histogram Table and 1 using 16 values) and an intermediate sum which will be used for the calculation of the remaining coefficients (2 using 32 values and 1 using 64 values. It is about the coefficient N°2, N°3 and N°1).

The performance estimation of the implementation structure (C-Block) of the C_3Haar task is given in Table 2.

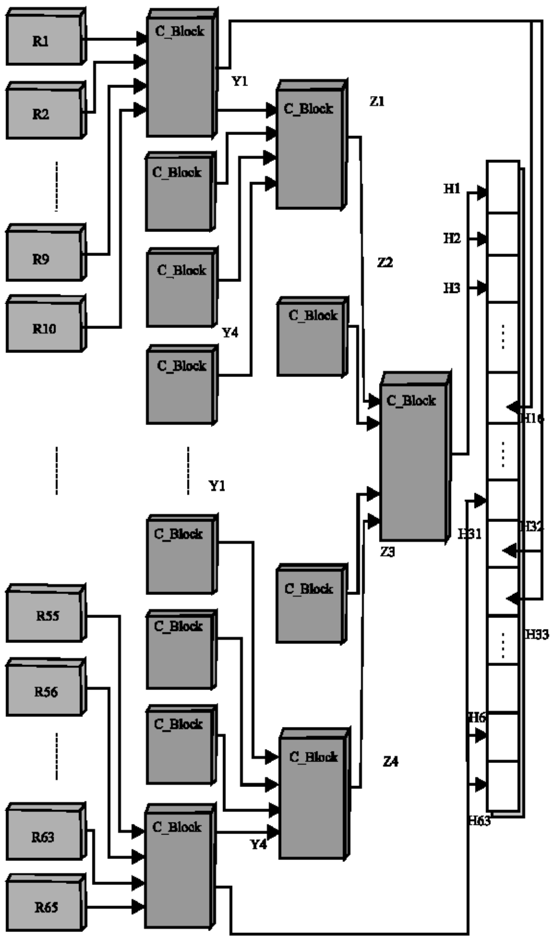


Fig. 7: Implementation structure of parallel solution

Table 3: Performances of the parallel solutions

Solution	Slices	Function generators	Flip Flops
Parallel solution	1866	2250	1215
Intermediate solution	559	630	423

Different architectural implementation solutions can be considered to make the Haar coefficients computation[36]. The choice of a solution depends on the application requirements and the system environment. These solutions are based on the C_Block module. One possible solution consists in using only one module C-Block. The histogram values are placed in a RAM. To access to the histogram values in the desired order we use a ROM for addressing the RAM. The calculation block will be used in a sequential mode. This solution permits to minimize the required surface, but the execution time can be significantly improved by considering parallel solutions.

A second solution Fig. 7 consists in using 64 registers and 21 C-Block. For a simultaneous access to the different histogram values the total processing is done in six cycles Fig. 8. In this solution the reorganization stage is not necessary, it can be made by using an adequate interconnection between the different components of the implementation architecture. This solution has a simple structure and controller; however, it requires a big number of CLB in the FPGA. The simulation results of this solution are represented in Fig. 8. We obtain the same results as in Fig. 4.

An Intermediate parallel solution is represented in Fig. 9. It consists in using 20 registers and 6 calculation blocks C_Blocks. This choice of the number of calculation blocks permit to avoid additional memory accesses to save the intermediate sums. In this solution the histogram values must be reorganized as for the first solution. The RTL simulation results of this solution are given in Fig. 10.

Table 3 gives an estimation of the number of slices occupied by the data path of each parallel solution. Execution time estimation can be easily obtained by considering the time performance of the C_Block module

Fig. 8: RTL simulation results of the parallel solution of the haar transform

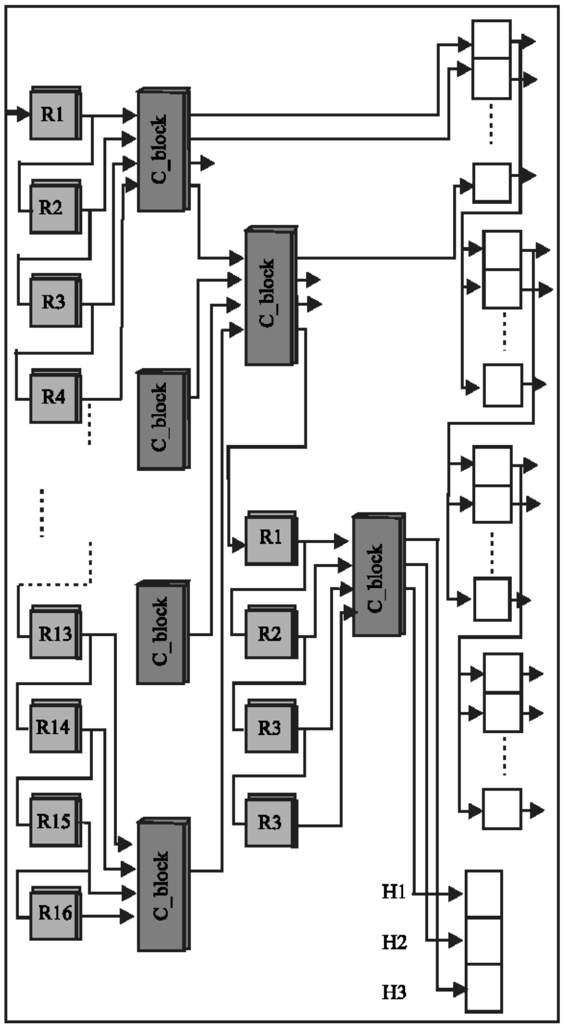


Fig. 9: Implementation structure of the intermediate solution

and the simulations results from which we can deduct the number of required cycle.

Hardware implementation of the dominant colors selection block: In this study we focus in hardware solutions for the block permitting to find out the 8 dominant colors in the DCD. The solution represented in Fig. 11 consists in searching the maximal value in each step. This necessitates reading the histogram values for 8 times. The progression of the different operations is managed by a controller. Histogram values can be placed in a RAM or directly accessed from registers, depending on the implementation solution of the histogram block[37].

A more efficient solution is represented by Fig 12. Only one access to each histogram value is enough to determine all the dominant colors. We allocate 8 registers R_1, R_2...,R_8 which will contain the 8 dominant colors. Each histogram value « DATA » will be compared to the R_1 value. If the DATA value is higher than the R_1 value, R_1 takes the DATA value and the other registers will be shifted: R_j takes the value of R_{j-1} for j>1. The value of R8 will be lost because it does not belong to the 8 dominant colors. If not (DATA is not higher than $R_{1)}$ we compare A to R_2 and we use the same reasoning for the rest of register.

We have implemented this solution by using the architecture structure represented in Fig. 12. The multiplexers permit to select the new values which will be placed in the corresponding registers. Table 4 gives the control signals of the different MUXs. For example if the control signal is equal to 01 the register R_j takes the DATA value and if the control signal is 11 it takes the R_{j-1} value (register N° j-1) . The structure of an A block which generate the control signal for the corresponding MUX is given by Fig. 13.

Fig. 10: RTL simulation results of intermediate implementation solution of the haar transform

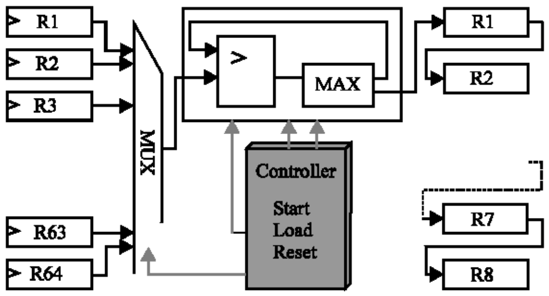


Fig. 11: Sequential solution for dominant colors

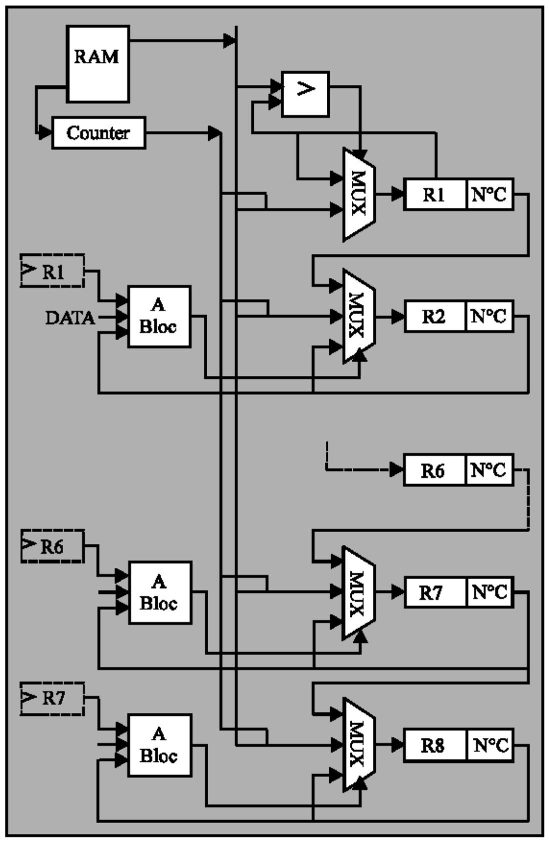


Fig. 12: Parallel solution for dominant colors

Table 4: Control signals for MUXs

	DATA	R_j	R_{j-1}
R1	1	0	x
Rj (1< j <9)	01	00	11

Table 5: Performance estimation

No. of slices	254
minimal period	12.295ns
F maxi	81.334MHz

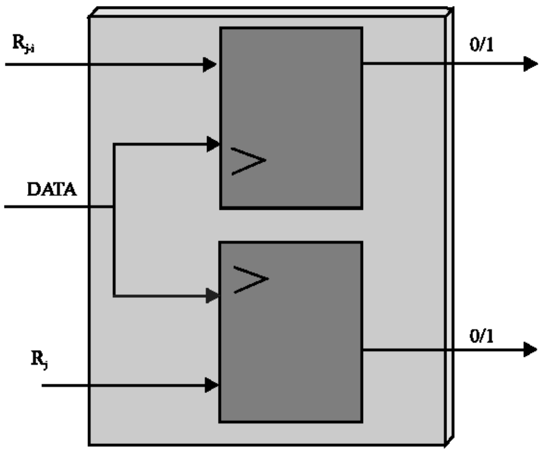


Fig. 13: The A-block structure

Performance estimation of the data path of the implementation structure of this solution is given in Table 5. We use 18 bits histogram bins. The RTL simulation results are represented in Fig. 14. We obtain the 8 dominant colors.

CONCLUSION

The usage of new integration technologies and hardware solutions for embedded systems can be very useful for systems requiring high performances in term of processing power, flexibility, consumption, etc. Multimedia applications represent a good example for such constrained systems. In this study we have interested to content based video indexing. We have described hardware implementation solutions of the Haar

Fig. 14: Simulation results of the parallel solution of the dominant colors module

transform and the dominant colors modules that represent the principal extraction steps respectively for the Compact Color Descriptor and the Dominant Colors Descriptor algorithms. The described solutions can be used for video indexing under real time constraint. The synthesis and simulation results for the provided solutions have been given.

REFERENCES

1. Werner Bailer, Franz Höller, Alberto Messina, Daniele Airola, Peter Schallauer and Michael Hausenblas, 2005. State of the Art of content analysis tools for video, audio and speech. Report, FP6-IST-507336 PrestoSpace Deliverable D15.3 MDS3.
2. Ying Li, Tong Zhang and Daniel Tretter, 2001. An overview of video abstraction techniques. Document HPL-2001-191, Imaging Systems Laboratory, HP Laboratories Palo Alto.
3. Hualu Wang, Ajay Divakaran, Anthony Vetro, Shih-Fu Chang and Huifang Sun, 2003. Survey of compressed-domain features used in audio-visual indexing and analysis. J. Vis. Commun. Image R., 14,pp: 150-183.
4. Ahanger, G. and T.D.C. Little, 1996. A survey of technologies for parsing and indexing digital video and image representation. Journal of Visual Communication, (Special Issue on Digital Libraries), 7,pp: 28-43.
5. Chang, S.F., A. Puri, T. Sikora and H. Zhang, 2001. Overview of the MPEG-7 Standard. IEEE Tr. CSVT, 11,pp: 688-695.
6. Manjunath, B.S., P. Salembier and T. Sikora, 2002. Introduction to MPEG-7 Multimedia Content Description Interface, J. Wiley, New York..
7. Manjunath, B.S., Jens-Rainer Ohm, Vinod V. Vasudevan and Akio Yamada, 2001. Color and texture descriptors. IEEE Tr. CSVT, 11,pp: 703-715.
8. Bober, M., 2001. MPEG-7 visual shape descriptors. IEEE Tr. CSVT, 11: 716-719.
9. Jeannin, S. and A. Divakaran, 2001. MPEG-7 visual motion descriptors. IEEE Tr. CSVT, 11,pp: 720-724.
10. Mezaris, Vasileios, I. Kompatsiaris, N V. Boulgouris, and M.G. Strintzis, 2004. Real-time compressed-domain spatiotemporal segmentation and ontologies for video indexing and retrieval. IEEE Transactions on circuits and systems for video Tech., 14,pp: 606-621.
11. Birant, B., Örten, Medeni Soysal and A. Aydin Alatan, 2005. Person identification in surveillance video by combining MPEG-7 experts. 6th International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS 2OO5, Montreux, Switzerland.
12. Jae-Gon Kim, Hyun Sung Chang, Young-tae Kim, Kyeongok Kang, Munchurl Kim, Jinwoong Kim and Hyung-Myung Kim, 2002. Multimodal approach for summarizing and indexing news video. ETRI J., pp: 24.
13. Chu Hong, H.O.I., 2002. A study of content-based video classification, indexing and retrieval. Master of Philosophy, Chinese University of Hong Kong.
14. Shih-Fu Chang, 2003. Content-based video summarization and adaptation for ubiquitous media access. IEEE International Conference on Image Analysis and Processing (ICIAP), Montavo, Italy.
15. Andres Dorado, Janko Calic and Ebroul Izquierdo, 2004. A rule-based video annotation system. IEEE Transactions on Circuits and Systems for Video Tech., 14,pp: 5.
16. Snoek, C.G.M. and M. Worring, 2005. Multimodal Video Indexing: A review of the stateof-the-art. Multimedia Tools and Applications, 25: pp: 5-35.
17. Hsu, W., L. Kennedy, C.W. Huang, S.F. Chang, C.Y. Lin and G. Iyengar, 2004. News video story segmentation using fusion of multi-level multi-modal features in TRECVID 2003. (ICASSP 2004) IEEE International Conference on Acoustics, Speech, and Signal Processing Montreal, Quebec, Canada.
18. Di Zhong and Shih-Fu Chang, 2000. Video Shot Detection Combining Multiple Visual Features. Department of electrical engineering, Columbia University ADVENT Technical Report #092.
19. Ebrahimi, T., Y. Abdeljaoued, R.M. Figueras I Ventura and O. Divorra Escoda, 2003. MPEG-7 CAMERA. ICIP2001, III: 600-603.
20. Denchev, R. and W. Stechele, 2005. An Experimentation Environment for MPEG-7 based Driver Assistance. Eurocon 2005, Belgrade.
21. Steiger, O., T. Ebrahimi and Cavallaro, 2005. A, real-time generation of annotated video for surveillance. Prceeding of IEE workshop on image analysis for multimedia interactive services, WIAMIS.
22. Kongwah Wan, Xin Yan, Xinguo Yu, and Changsheng Xu, 2003. Real-time goal-mouth detection in mpeg soccer video. ACM Multimedia Conference.
23. Di Zhong, Raj Kumar and Shih-Fu Chang, 2001. Real-time personalized sports video filtering and summarization, proceedings of the ninth ACM international conference on Multimedia, pp: 623-625.
24. Andreas Savakis, Pawel Sniatala and Radoslaw Rudnicki, 2003. Real-time Video Annotation using MPEG-7 Motion Activity Descriptor. Conf. MIXDES 2003, Lodz, pp: 26-28.

25. Bret Woz and Andreas Savakis, 2004. A VHDL MPEG-7 shape descriptor extractor. ACM/SIGDA 12th international symposium on field programmable gate arrays, Monterey, California, USA pp: 246-246.
26. Stechele, W. and S. Herrmann, 2005. Reconfigurable hardware acceleration for video-based driver assistance. Workshop on hardware for visual computing, Tübingen.
27. Tesier, R. and W. Bulleson, 2001. Reconfigurable Computing for digital signal processing: A survey. J. VLSI Signal Processing, 28: 7-27.
28. Krishnamachari, S., A. Yamada, M. Mottaleb, E. Kasutani and Multimedia, 2000. content Filtring, browsing, and matching using mpeg-7 compact color descriptors. International conference on visual information systems (Visual2000), France, pp: 200-211.
29. Mohamed Hammami, Chen Li, Boulbaba Ben Amor and V.I.A.L. Christian, 2002. Classification d'images par concept. MEDIANET 2002, Sousse-Tunisia.
30. Emna Fendri, Hanane Ben Abdallah and Mohsen Ardebilian, 2002. Une nouvelle approche pour la détection des transitions progressives. MEDIANET, Sousse-Tunisia.
31. Krishnamachari, S., M. Mottaleb, Philips Research (USA), 2000. Color compact descriptor for fast image and video segment retrieval. San Jose, CA: SPIE, 3972: 581-589.
32. Yining Deng, B.S., Manjunath, Charles Kenney, Michael S. Moore and Hyundoo Shin, 2001. An efficient color representation for image retrieval. IEEE TR. IP, pp: 10.
33. ISO/IEC JTC1/SC29/WG11, 2000. MPEG-7 Visual part of experimentation Model .ISO/IEC JTC1/SC29/WG11, N3321b.
34. Deng, Y., C. Kenney, M.S. Moore and B.S. Manjunath, 1999. Peer group filtering and perceptual color quantization. Proc. IEEE Int. Symp. Circuits Syst, 4: 21-24.
35. Hao-Wei Chang, 2002. A study on content-based video retrieval. Institute of computer and information science, National Chiao Tung University. http://debut.cis.nctu. edu.tw/pages /Demo/CBVR/paper_E.pdf.
36. Ben Abdelali, A., A. Mtibaa, E. Bourennane, and M. Abid, 2005. Structures parallèles pour l'implémentation de la transformée de Haar sur FPGA. Journées Francophones sur l'Adéquation Algorithme Architecture JFAAA'05, Dijon, France.
37. Ben Abdelali, A. and A. Mtibaa, 2005. Toward hardware implementation of the compact color descriptor for real time video indexing. Advances in Engineering Software, V., 36: 435-496.