Survey on High Level Synthesis for Image Processing Applications

Abstract-The term High Level synthesis refers to the generation of register transfer level designs from the user defined behavioral specifications in an automatic manner. The main goal of the HLS is to generate a RTL level design that implements the specified behavior while satisfying the design constraints and optimizing the given cost function. High level synthesis is used in image processing because of its vast advantages in the field of automatic electronic hardware design. The research in HLS on image processing leads to the development of a better image processing results automatically by configuring it with perfect parameters. So, by using HLS for image processing applications lead to new development in this area. Here, present a detailed survey about the recent developments in the HLS on image processing and discussed the recent techniques used in HLS for image processing applications. To locate the further improvement in using high level synthesis in the field of image processing the obstacles in the existing techniques are also discussed. Additionally, the discussion is given about their approaches so that the new development in high level synthesis for image processing is easier for the researchers. Finally, some of the research issue is also addressed to precede the further research on the same direction.

Keywords:-High Level Synthesis, behavioral specification, RTL, Image Processing, Efficiency.

1. INTRODUCTION

Development in technology in present world leads us to opt for an automatic design tool with user specified functions. It should perform high, flexible and low power also the designer should use it of less cost [12]. The semiconductors and electronic systems are much complex that designing them would be impossible without electronic design automation (EDA).It is a type of software tools used for designing VLSI chips with high accuracy by consuming a less time. All these EDA tools functions together with a design flow that is designed by the chip designer in order to design and analyze the chip. Widely used EDA tools include Cadence, Synopsys and Mentor Graphics [14].

- J.Hemalatha is currently a Research scholar in AnnaUniversity,Chennai , India. . E-mail: hemabalan2001@yahoo.com
- Dr M.Joseph ,Supervisor, India.

Image-processing methods are mostly used to correct input image data by adjusting of image parameters such as resolution, contrast, color tone and gradation [15]. Latest image-processing technologies eradicate such adjustments by breaking down captured images into basic components and then reconstructing them. Image processing plays an important role in the field of Biology, Astronomy, Medicine, Security, Biometrics, Satellite Imagery etc. for Noise removal, Contrast adjustment, Edge detection, Region detection, segmentation, Image compression, Digital in painting etc. Mathematical functions such as Calculus, Linear Algebra, Probability and Statistics, Differential Equations (ODEs and PDEs), Differential Geometry, Harmonic Analysis (Fourier, wavelets, etc).

High Level Synthesis is an automatic electronic hardware design process in which the input is the behavioral description of the desired function and the output is a hardware implementation of the

Integrated circuit performing the desired function. The behavioral descriptions are usually coded in C [1], ANSI C, and C++ etc. Due to the advancement in VLSI it is possible to implement systems that were not previously possible and architectures like SIMD array processors are best suitable for VLSI implementation [13]. The main objective of HLS in the field of image processing is to provide a speed and accurate design with reduced cost and area. High level synthesis tools and frameworks are used in designing some of image processing applications [6], [10]. Image processing algorithms are implemented in hardware by coding the behavioral description of the algorithms and feeding it to HLS tool and taking its output and converting it to logic level using appropriate tools [18]. The input behaviors are parsed and then applying suitable transformations we can design the structural RTL.

The essential issues of HLS for various image processing applications include behavioral specification language, target architecture, intermediate representation, operation scheduling, allocation/ binding and control generation. Quality of results obtained from high level synthesis can be improved by using global code motions [17]. The normal high level synthesis flow is shown in figure 1.

Fig1.Normal High Level Synthesis Flow

2. Related works

For doing a survey, high level synthesis on image processing are taken from standard publisher like, IEEE, Elsevier, Springer and Inderscience. Here, a review of papers is presented by categorizing based on papers on high level synthesis and papers based on HLS for image processing applications.

Section-1: High Level Synthesis

In this section we discuss about the High level Synthesis methods that are found in the literatures. R. Pasko *et al,* [1] implemented an Algorithm for the Elimination of Common Subexpressions. In their design for High level Synthesis on digital filtering, image processing, linear transforms, etc. they mainly concentrated in reducing the area of the hardware generated by optimizing the multiplication of a variable by a set of constants. The implemented algorithm was an efficient solution to the multiple constant multiplication problems. They designed the optimization procure by targeting the multiplication block area reduction. Their algorithm for the Elimination of Common Subexpressions worked by expressing the coefficients in a canonical signed digit (CSD)

format, then identified the multiple patterns in the coefficient set and finally removed those patterns and calculated them only once.

Akitoshi Matsuda [2] implemented a High-Level Synthesis Technology Using C Source Code Analysis. In his design a system-level C source code visualization and optimization was considered to facilitate consistent hardware design. His optimization of system-level C source code was more efficient compared to co-design and co-verification in hardware and software. According to his design flow, if the entire structure can be visualized, the system level C source code can be optimized, which resulted in an improvement in design quality and a shortening of the design cycle. His implementation eliminated unnecessary variables like if statement join, and for statement join in the C source code during software design. He also reduced the hardware design effort by using a combination of optimized C source code technology and high-level synthesis technology. The experimental results for his design exhibited a reduction in the circuit area by up to 10% in hardware design.

Bertrand Le Gal *et al*, [3] implemented a dynamic memory access management technique for high-performance DSP applications using high-level synthesis. Their design mainly focused on implementing memory interfacing modules that can be automatically generated from a high-level synthesis tool, which can efficiently handle predictable address patterns as well as random dynamic address computations.

The implementation includes a new address memory sequencer architecture that can perform pipeline memory accesses for static and dynamic access sequences for the memory access management in a high-level synthesis flow. The main target of their implemented system was the design of hardware parts of complex systems such as digital signal processing IP cores with specific performance constraint. The design when experimented exhibited a reduction in address transfers between the memory and the data path units leading to power saving and reduced latency.

Haifeng Zhou *et al*, [4] implemented a 0–1integer linear programming (ILP) technique for solving memory-mapping problems in high-level synthesis. Their implemented HLS method synthesizes the source memory using one or more memory modules from a target memory library at a higher level. They targeted memory modules commonly used in data intensive applications in the fields of speech, image and video processing. They found that the area cost of memory components for the previous methods of HLS was as high as 80% of that of the complete design and these highly motivated them to design a new technique with reduction of cost. If given the high-level specification of the source and the library modules in terms of word-count, bit width and the port configurations, their designed technique implements the source memory module by using target memory modules in an efficient manner so as to optimize a user-given cost function. They demonstrated that their mapping algorithm can generate a variety of cost-effective memory designs based on the user-given cost function and the target library.

A HLS Approach in Designing FPGA-Based Custom Coprocessor was implemented by Mahendra Samarawickrama *et al*, [5].They

designed a image pre-processing architecture (Vision system) using high level synthesis method which performed faster than the optimized software implementation on an Intel Core 2 Duo GPU. The high-level synthesis tool was used to design, implement and test the vision system within the context of required control, synchronization, and parameterization on a processor based platform. In addition, they also used both the HLS tools and HDL for the development of the processing cores. The automation of the entire design process from conceptualization to synthesize reduce the design time and cost. In their design the three steps (scheduling, allocation and binding) which form the core of transforming a behavior into a structure were closely interdependent to achieve performance/cost tradeoff of a design. A significant outcome of their research was that it took roughly five times less effort to implement and evaluate the work load on the FPGA when using HLS tool and hence it requires a low cost.

Andrew Stone and Elias S. Manolakos [6] designed a tool to facilitate the High Level Synthesis of Parallel Processing Array Architectures. Their designed tool can automatically convert abstract algorithmic descriptions (Dependence Graphs) to synthesizable VHDL models and test benches representative of distributed memory and control processor arrays (Signal Flow Graphs). The automatic conversion facility of their design freed the designer from coding and testing separately adapting the large space of available parallel architectures for a given problem. The DG2VHDL tool made possible the high level synthesis of processor arrays the computation of the Discrete Wavelet Transform and the estimation of Higher Order Statistics. The quality and scalability of the generated VHDL models was found to be near optimal.

A method of dynamically increasing the scope of code motions during the high-level synthesis of digital circuits was made possible by S. Gupta *et al*, [7]. Two branch balance techniques were implemented and applied dynamically during the scheduling process in high level synthesis in order to increase the scope for application of speculative code motions. The new technique increased the quality of high-level synthesis results of designs with complex and nested conditionals and loops by employing speculative code motions. The main function of the implemented two techniques was to add scheduling steps to the branch of a conditional construct with less scheduling steps which in turn 'balances' or equalizes the number of scheduling steps in the conditional branches thereby increasing its scope. The algorithm for dynamic branch balancing techniques was implemented in HLS synthesis frame work 'Spark'. Their experimental results proved that improvements of up to 38% in cycles on the longest path and 37% in controller size.

David J. Kolson *et al*, [8] brought out a new transformation for the scheduling of memory-access operations of high-level synthesis. That transformation was highly suited to memory-intensive applications such as video compression, image convolution with synthesized designs containing a secondary storage accessed by certain instructions. Their designed transformation worked by removing redundant or unnecessary instructions such as load and store during the

transformation of loops. The reduction of those instructions the leads to the decrease of secondary memory bandwidth demands in the design. They experimented the designed transformation in their pre-designed Percolation- Based Scheduler and observed a significant reduction in the number of memory operations and an increase in performance gain up to 100%.

M. Milford and J. McAllister designed a valved dataflow for FPGA memory hierarchy synthesis [9]. They designed a novel dataflow application modeling dialect, known as 'Valved Dataflow' as a solution to the problem to automated synthesis of multi-level memory hierarchies in high level synthesis technologies for FPGA devices. The designed technique was the first automated solution to that problem. They conducted experiments on their design and found out that the automatically generated implementations support real-time processing for current image processing application standards like H.264. They also observed an improvement in performance and cost of hierarchies automatically generated for Motion Estimation, Matrix Multiplication and Sobel Edge Detection applications on Virtex-5 FPGA.

Sumit Gupta *et al*, [10] implemented a high level synthesis tool 'SPARK' that was fed with a input of behavioral description in ANSI-C and generated a synthesizable register-transfer level VHDL as output for certain industrial image processing tools. In Spark instruction level parallelism and its re-instrumentation for high level synthesis was done by a previously designed parallelizing compiler technology. By using spark they designed the design flow, industrial image processing tools including speculative code motions and dynamic transformations and also they described how those transformations, optimizing synthesis and compiler techniques were used efficiently by a scheduling heuristic. They examined the design and found that code transformations used leads up to 70 % rise in performance without incrementing the overall area and critical path of the final output design.

ACoordinatedParallelizingCompiler Optimizations for high level synthesis of certain image processing tools was designed by Sumit Gupt. A *et al*, [11]. They adapted Spark for exploring the suitable source level and low level parallelizing transformations. They contributed to the Spark tool by implementing a set of parallelizing code transformations that were included at the source level and during scheduling. For optimizing the result and to depress the impact of control flow constructs of the quality of results those transformations can be used. They first applied a set of source level pre-synthesis transformations containing common sub-expression elimination (CSE), copy propagation, dead code elimination and loop-invariant code motion and more coarse level code restructuring transformations like loop unrolling. They then performed scheduling techniques by using a set of aggressive speculative code motions that can re-order, Speculate and duplicates (if needed) for maximally parallelizing the. Their implemented design dynamically coordinates CSE and code motions, that technique was known as "Dynamic CSE". They tested their technique and found an efficient performance.

Section-2: HLS for image processing applications

This section presents a analysis about the latest High Level Synthesis methods for Image processing available in the literature. Caaliph Andriamisaina *et al*, [12] designed a multimode architecture for image processing applications. Their design includes a dedicated design flow and its associated High level synthesis tool GAUT. The working process of the design includes a set of time- wise mutually exclusive tasks, specified through switch- case statements with their associated pragma (to define the associated throughput constraints) and then that tasks were sorted to define the order in which they were to be scheduled and bound. In their design the register, the steering logic, and the controller complexities were reduced by the designed joint-scheduling algorithm which works by maximizing the similarities between the control steps and also they adopted specific binding approaches for both operators and storage elements in which the similarities between the data paths were maximized. The result of the design was a multimode architecture that includes a single FSM and a shared data path (a set of multiplexers, storage elements, and operators) was generated with its test-bench. A promising reduction in the overall design area overhead was obtained from the designed High level Synthesis technique for image processing applications.

A. Boubekeur and G. Saucier [13] designed a High Level synthesis Tool for Massively Parallel Image Processing ASICs. The input to their tool was the a high level description of the algorithm (Image Processing algorithms) and the output was an optimized circuit organized as a SIMD (Single Instruction Multiple Data) mesh connected array of one-bit processing element with minimized resources. The designed tool rapidly generated the ASICs for low level image processing applications with massively parallel architecture and was able to generate a dedicated ASIC that was optimized in area and performance.

A Deeply Pipelined and Parallel Architecture for Denoising Medical Images was implemented by Frank Hannig *et al,* [14]. The implemented design performed an almost automatic synthesis of a highly complex, throughput optimized architecture of an adaptive multi-resolution filter, which includes 16 parallel working modules, where the most computationally intensive module achieves software pipelining of a factor of 85 (computations of 85 iterations overlap each other).Their implemented technique contributed to the complexity ,Productivity and Power efficiency of the design also it was capable of significantly reducing the well known productivity gap of embedded system design by almost two orders of magnitude.

Image and video processing platform for field programmable gate arrays using a high-level synthesis was designed by C. Desmouliers *et al*, [15].The designed IVPP(Image And Video Processing Platform) was mainly based on FPGA and it can be used to realize and to test complex algorithms for real-time image and video processing applications. The aim of their design was to accelerate the hardware development time by using pre-built hardware blocks. Their implemented design also had an added advantage of customizing logic into the existing framework by adding additional accelerator units. The

software based approach provided the platform a rapid development of image and video processing algorithms. Their IVPP was a cost-effective, rapid development and prototyping platform for key image processing applications.

Francois S. Verdier and Bertrand Zavidovique [16] designed a high level synthesis system for the designing of VLSI image processing architectures. Their designed system had a "front-end" data-flow emulator for the algorithm validation and ALPHA as RTL-synthesis system. The ALPHA system in their design produced an efficient solution for all the restricted domain of the applications by implementing a random search along the design space. For data path synthesis, control synthesis and data-path completion they used two simulated Annealing (SA) algorithms running in sequence. Since validation phase of the algorithm was tightly coupled with its integration phase allowed the generation of behaviorally correct systems and with reduced design cycle.

The method of using global code motions to improve the quality of results for high-level synthesis was implemented by Sumit Gupta *et al*, [17]. They designed a set of speculative code-motion transformations that enabled the movement of operations through, beyond, and into conditionals in order to maximize the performance and to overcome the effects of programming style on the quality of output circuits. They also designed supporting code-motion techniques and variable renaming techniques along with their code transformations in a High level synthesis framework known as 'Spark'. They fed the behavioral description of the selected image processing application in ANSI-C as input to Spark and generated synthesizable RTL VHDL. Their experimental results on MPEG-1, MPEG-2 and GNU image manipulation program applications revealed a good improvement in the quality of the output design.

A platform for high level synthesis of memory-intensive image processing algorithms was designed by Tim Papenfuss and Holger Michel [18]. Their design includes a custom memory system of buffers, caches and an optimized commercial memory controller that improves available SDRAM bandwidth by up to 4x.Their designed platform for HLS hold good to obtain sufficient throughput from external SDRAM when the image resolution was high. The implemented platform was designed using a high level synthesis tool known as 'CatapultC'. They demonstrated the design's efficiency by implementing two memory-intensive algorithms using 47.0Gbit/s and 5.7 Gbit/s of on-chip and off-chip memory bandwidth respectively. The designed platform includes custom memory system eradicating the usual obstacles like low throughput and high latency.

Yazhuo Dong and Yong Dou designed a parameterized architecture model in high level synthesis for image processing applications [19] to generate the hardware frames for all image processing applications automatically. They also described how to use data dependence analysis to develop a compilation and synthesis strategy for generating various parameters for various image processing applications. The data dependence analysis performs by keeping reduction in memory access and small size memory block as its objective while exploring a wide range of program

transformations in order to enhance the performance of the design. A special control unit was also designed to control the dataflow that made it to have a potential to store a small part of data values in internal RAM and smart buffer while still affording sufficient memory bandwidth for the custom data path.

A technique for scheduling and processor allocation during the synthesis of integrated heterogeneous pipelined processing elements for DSP applications (Image Processing) was designed by Ali Shatnawi *et al*, [20]. Their new technique for time scheduling and hardware allocation includes a designed algorithm which produced high efficiency than that of the results obtained from homogeneous implementations because of their reduced iteration period schedules and reduction in time complexity at design. They achieved efficiency in the output hardware implementation at logic-level by shrinking the processing units counts used without changing the rate and delay optimality criteria. When compared with the existing heterogeneous algorithms, the designed algorithm produced reduced time complexity schedules and iteration period for some applications. Improvement in performance of high level synthesis during the synthesis of image processing tools was detected.

Section- 3: Unsolved Problems

After analyzing the literature survey, the following are some of the issues identified that can be taken further to do the research. Even though various techniques presented to handle, still there is a need of good techniques to solve the following research issues.

- Power consumption is an important problem in major of the high level synthesis designs as the controller and data path decreases in a design the performance and the power consumption are affected.
- Some methods support the behavioral descriptions in C source code in ANSI C and may not support System C and C++ code.
- For control intensive applications such as dictionary searches, etc., design flow and sequencer-based architectures are not suitable where most of the computations are address computations.
- Most of the tool's features are under development and they are now in the advanced testing stage and they can be used to improve the efficiency.
- More comprehensive cost models can be developed including the control and interconnect costs of the code motions.
- Code motions such as conditional speculation are critical to code because they duplicate operations into multiple basic blocks.
- Manual coding uses less resource than that of the automatic coding systems. By making the automatic coding techniques effective it can be made efficient.
- Loop transformations have great impact on the quality of synthesis results in order to increase the quality of design better loop transformation techniques should be designed.

Section 4: Conclusion

A detailed survey of recent developments in HLS for image processing from the standard publishers like IEEE, Elsevier, Springer and Inderscience was presented. Most of the methods aim at increasing the overall performance of the design. In most of articles a modification in the scheduling process of high level synthesis is made to increase the overall system performance. The application of HLS in image processing includes architecture design, algorithm design, industrial image processing tools, filters etc. For reducing the area of design, scheduling methods and elimination methods are adapted. For reducing power memory access managements, parallel working modules and software pipelining are preferred. In order to achieve efficiency in cost of resign 0–1integer linear programming C-based HLS design flow AccelDSP and HDL design methods are used. From the review, the identification was that in future our research should target the reduction of area of the design. By making modifications and including more techniques in the HLS tools a hardware design for the desired function was automatically designed that utilize less hardware components with reduced time consumption.

Section-5: References.

[1] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde and D. Durackov, "A New Algorithm for Elimination of Common Subexpressions", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 18, No. 1, PP.58-68, January 1999.
[2] Akitoshi Matsuda, "A High-Level Synthesis Technology Using C Source Code Analysis", International Journal of Image Processing and Visual Communication, Volume 1, Issue 2, PP.29-34, October 2012.
[3] Bertrand Le Gal, Emmanuel Casseau, and Sylvain Huet, "Dynamic Memory Access Management for High-Performance DSP Applications Using High-Level Synthesis", IEEE Transactions on VLSI Systems, Vol. 16, No. 11, PP.1454-1464, November 2008.
[4] Haifeng Zhou, Zhenghui Lin and Wei Cao, "ILP method for memory mapping in high-level synthesis", Microelectronics Reliability, Volume 43, Issue 7, PP 1163–1167, July 2003.
[5] Mahendra Samarawickrama, Ranga Rodrigo, and Ajith Pasqua, "HLS Approach in Designing FPGA-Based Custom Coprocessor for Image Preprocessing", 5th International Conference on Information and Automation for Sustainability (ICIAFs),PP.167-171,December 2010.
[6] Andrew Stone and Elias S. Manolakos, "DG2VHDL: A Tool to Facilitate the High Level Synthesis of Parallel Processing Array Architectures", Journal of VLSI signal processing systems for signal, image and video technology, Volume 24, PP.99-120, February 2000.
[7] S. Gupta, N. Dutt, R. Gupta and A. Nicolau, "Dynamically increasing the scope of code motions during the high-level synthesis of digital circuits", IEE Proceedings on Computer and Digital Technolog, Vol. 150, No. 5,PP.330-337,September 2003.
[8] David J. Kolson, Alexandru Nicolau and Nikil Dutt, "Elimination of Redundant Memory Traffic in High-Level Synthesis", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 15, No. 11, PP.1354-1364, November 1996.
[9] M. Milford and J. McAllister, "Valved Dataflow for FPGA Memory Hierarchy Synthesis", IEEE International Conference on Acoustics, Speech and Signal Processing, PP.1645 – 1648, March 2012.
[10] Sumit Gupta, Nikil Dutt, Rajesh Gupta and Gupta Alex Nicolau, "SPARK: A High-Level Synthesis Framework For Applying Parallelizing Compiler Transformations", Proceedings of the 16th International Conference on VLSI Design, PP. 461 – 466, January 2003.
[11] Sumit Gupta, Nikil Dutt, Rajesh Gupta and Gupta Alex Nicolau, "Coordinated Parallelizing Compiler Optimizations and High-Level Synthesis", ACM Transactions on Design Automation of Electronic Systems, Vol. 9, No. 4, PP. 1-31,October 2004.
[12] Caaliph Andriamisaina, Philippe Coussy, Emmanuel Casseau and Cyrille Chavet, "High-Level Synthesis for Designing Multimode Architectures", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 29, No. 11,PP.1736-1749, November 2010.
[13] A. Boubekeur and G. Saucier, "A High Level Synthesis Tool For Massively Parallel Image Processing ASICs", Proceedings of the conference on Advanced Computer Technology, Reliable Systems and Applications in 5th Annual European Computer Conference,PP.53-57,May 1991.
[14] Frank Hannig, Moritz Schmid, Jurgen Teich and Heinz Hornegger, "A Deeply Pipelined and Parallel Architecture for Denoising Medical Images", International Conference on Field-Programmable Technology (FPT), PP.485-490, December 2010.
[15] C. Desmouliers, E. Oruklu, S. Aslan,J. Saniie and F.M. Vallina, "Image and video processing platform for field programmable gate arrays using a high-level synthesis", IET Comput. Digit. Tech., Vol. 6, PP. 414–425, May 2012.
[16] Francois S. Verdier and Bertrand Zavidovique, "A High Level Synthesis System for VLSI Image Processing Applications", VLSI Design, Volume 7, No. 4, PP.321-336, 1998.

[17] Sumit Gupta, Nick Savoiu, Nikil Dutt, Rajesh Gupta, and Alex Nicolau,"Using Global Code Motions to Improve the Quality of Results for High-Level Synthesis", IEEE Transactions On Computer-Aided Design Of Integrated Circuits And Systems, Vol. 23, No. 2, PP.302-312,February 2004.
[18] Tim Papenfuss and Holger Michel, "A Platform for High Level Synthesis of Memory-Intensive Image Processing Algorithms", Proceedings of the 19th ACM/SIGDA international symposium on Field programmable gate arrays, PP.75-78, 2011.
[19] Yazhuo Dong and Yong Dou, "A Parameterized Architecture Model in High Level Synthesis for Image Processing Applications", Asia and South Pacific Design Automation Conference, PP. 523 – 528, January 2007.
[20] Ali Shatnawi, Jehad Ghanim and M.O. Ahmad, "High level synthesis of integrated heterogeneous pipelined processing elements for DSP applications" Journal on Computer and Electrical Engineering Vol.30, No.8, PP.543-562, November 2004.