Design of filter for image de-noising using discrete wavelet transform for ASIP

Mood Venkanna*, Rameshwar Rao and P. Chandra Sekhar

Department of Electronics and Communication Engineering,
University College of Engineering,
Osmania University, India
Email: venkatmood03@gmail.com
Email: rameshwar_rao@hotmail.com
Email: sekharp@osmania.ac.in
*Corresponding author

Abstract: Application specific instruction-set processors (ASIP) is a customised processor for user specific application. Though a significant research has been done on this, still it is most promising technology, due to lack of efficient methodologies for designing the processor configuration according to the applications. Again ASIP solution explores the trade-off between the dedicated hardware design and flexibility among software. It endeavours to fulfil the functionality of an algorithmic with low power costs and less complexity. In this paper, an approach is considered to design a processor for image de-noising. The design of suitable filter is an important task for the transmission and real-time processing. Designing ASIPs requires a suitable design of custom data path simultaneously modify the instruction-set, decoder including the compiler. We present an ASIP based on custom architecture design using the discrete wavelet transform (DWT) as a filter. It starts with the general purpose data path like MIPS. It customises the data path iteratively for better power utilisation, usable area and performance. All the experiments have been synthesised using Xilinx FPGA and also verified in Spartan board. The subjective evaluations of the filter is analysed through various figures. Further it is implemented in HDL to support the customised processor.

Keywords: image filtering; wavelet transform; application specific instruction-set processor; ASIP; impulse noise; field programming gate array; FPGA; VHDL.

Reference to this paper should be made as follows: Venkanna, M., Rao, R. and Chandra Sekhar, P. (2021) 'Design of filter for image de-noising using discrete wavelet transform for ASIP', *Int. J. Computational Vision and Robotics*, Vol. 11, No. 2, pp.201–213.

Biographical notes: Mood Venkanna received his BTech in ECE from JNTUH in 2003, MTech from JNTU, Hyderabad in 2008 and pursuing PhD in University College of Engineering Osmania University Hyderabad Telangana India. He has 14 Years of Teaching Experience. Presently, he is working as an Associate Professor in Department of Electronics and Communication Engineering in St. Martin's Engineering College Hyderabad. He has published papers in reputed international journals and national journals and attended in international conferences and national conferences. His area of interest is embedded systems and VLSI Design.

Copyright © 2021 Inderscience Enterprises Ltd.

Rameshwar Rao obtained his Bachelor of Engineering in Electronics and Communication Engineering from University College of Engineering, Osmania University, Hyderabad. He obtained both his MTech in Communication Engineering and PhD from IIT, Bombay. He worked as a Vice Chancellor of JNTU Hyderabad. His area of interest includes digital communication, digital design using VHDL and VLSI design.

P. Chandra Sekhar received his BE degree from Nagpur University, MTech degree from JNTU Hyderabad and PhD from Osmania University in 1991, 1999 and 2009 respectively. He has been awarded with Post Doctoral Fellowship by Shizuoka University, Japan for one year. Currently, he is working as a Professor and HOD Department of ECE, UCE, Osamina University Hyderabad, India.

1 Introduction

The embedded design technologies improve the productivity of the devices and systems. The key steps in the process used as compilation and synthesis implementation, integration and verification. Easily it completes the computation task according to required hardware units to reduce challenges in terms of scheduling time, low power, temperature, scalability, design complexity, efficiency, and flexibility. It consists of ASICs, ASIP and field programming gate array (FPGA) as well as the programming unit such as the DSP. These processor designs are used in various environments and time to market. High-performance processors, memory architectures and various communication design includes in the use of embedded processors and are very pervasive in real world computing. The components are used to implement the design interface, exchanging run-time information, monitoring according to our application demanding. The general purpose processor (GPP) is designed to implement digital devices but this kind of processor system does not know the program specifications or application features. A digital signal processor (DSP) is an essential unit of an embedded system for applications of signal processing. It is used as a GPP is a single chip VLSI unit and provides faster processing instructions. In ASIP system, the embedded designer implements the processor and memory architecture as per the desired applications. With the help of retarget able compiler technology, an ASIP achieves the design complexity and the desired functionality (Wong et al., 2014; Hoffmann et al., 2012; Schliebusch et al., 2015; Chen et al., 2016; Jain et al., 2015; Peters et al., 2015).

For image processing and computer vision application, elimination of noise remains a potential difficulty. Noise causes serious problems in areas that use computing image derivatives. It perturbs the observable image information due to addition or subtraction with the pixels' grey level values. It also interfere the activities communication channels or sensors used to transmit or receive the digital images. For better filter performance, many nonlinear filters are exploited in this regard. However, such filtering approaches are the initial stages of any image processing application. For tissue characterisation or automated detection frequency analysis have been more helpful (Gonzalez and Woods, 2016; Jain, 2009; Pitas and Venetsanopoulos, 2010; Chan et al., 2015; Windyga, 2016).

ASIP aims to create a link between ASICs and other hardware however it lacks the desired flexibility, reusability and programmability with little performance. However,

ASIC designs are very time-consuming and costly compared to FPGA. One of the on-chip design techniques is application specific instruction-set processor (ASIP) for achieving speed and flexibility at the same time. As the design customisation is focused more on the application domain, these are better specialised and optimised as compared to the DSPs. These can provide the desired features based on energy consumption, timing performance, as well as the required area. In this regard, the architecture description languages (ADLs) generates the software tool and the register transfer level (RTL) in order to help processor designer (Xilinx Inc., 2015; Brown and Rose, 2016; Nohl et al., 2010; Karam et al., 2016).

Application areas including image, hot spot elaboration kernels and video signal processing are benefited using specialised instructions. In these areas innovative algorithms with highly nonlinear operators are expanding. In these cases, ASICs cannot provide the desired flexibility. On the other hand, requirement of low energy cost, high computational performance and low silicon area in mobile and handheld gadgets make DSP solutions unsuitable. It needs special processor for better performance as a trade-off to flexibility (Brown and Rose, 2016; Nohl et al., 2010; Karam et al., 2016; Digital Signal Processing, 2015; Hoare et al., 2016; Texas Instruments TMS320, 2016; SPRUGH7, 2010).

Although entirely-hardwire solutions yields the best performance, a programmable processor excels in the applications that support multiple standards, or future changes to application requirements. In addition, programmability is one of the strategies to reduce cost and time-to-market duration. On the other hand, the instruction-set of an ASIP is selected in a way that a specific application can be programmed and executed much faster than a GPP. The processor specialisation also provides the desired trade off among performance, flexibility, and speed. ASIPs have the advantage of both efficiency and programmability. The design and implementation of ASIP have been relatively less explored in the literature (SPRUGW0B, 2010; SPRY147, 2010; Goslin, 2010; Probell, 2017; Schneider and Ferdinand, 2017; Engblom and Ermedahl, 2015; Bauer et al., 2017).

The adaptive algorithm operates in spatial domain in which any transformation can be applied directly to the image. To improve time and area efficiency, many enhancement techniques with different transformation, have been applied in this field (Pitas and Venetsanopoulos, 2010; Chan et al., 2015; Windyga, 2016). It helps to eliminate the noise from images by removing the extreme values from the image using filters. This has motivated the authors to apply the DWT-based filter for such task for better performance. It has been performed using the behavioural VHDL model so as to improve image clarity and readability.

2 Noise model

To examine the median filter performance, four impulse noise models have been implemented under practical conditions. These models are explained below.

- *Noise model 1:* salt-and-pepper impulse noise models are implemented in which the fixed external values such as 0 and 255 for 8-bit monochrome image that have been generated with equal probability randomly corrupt the pixels. The respective noisy image pixel is indicated as $X_{i,j}$ with respect to every image pixel located at (i, j)

having the intensity $S_{i,j}$. The probability density function (PDF) corresponding to $X_{i,j}$ with the noise density p can be expressed as:

$$f(x)=\begin{cases}\frac{p}{2}, & for \quad x=0\\ 1-p, & for \quad x=s_{i,j}\\ \frac{p}{2}, & for \quad x=255\end{cases} \tag{1}$$

- *Noise model 2:* in this model, impulse noise has been modelled more realistically using two fixed ranges instead of two fixed pixels. These ranges each having length *m* appear at both ends. In case *m* equals to 10, noise may take a value having a range of between [0, 9] or [246, 255]. Thus:

$$f(x)=\begin{cases}\frac{p}{2m}, & for \quad 0\le x<m\\ 1-p, & for \quad x=s_{i,j}\\ \frac{p}{2m}, & for \quad 255-m<x\le 255\end{cases} \tag{2}$$

3 Pipelined architecture

To make the critical path shortened or to increase the parallelism of the architecture, the pixel elaboration has been split over a pipelined architecture. It increases the data throughput as required in this work. However, it increases silicon area overhead, latencies, and pipeline dependencies. There exist data dependencies among neigh bored instructions which cause an instruction for an operand. It is essential for interlocking pipeline mechanisms. The pipeline designed allows for instruction processing using the following stages:

1 the instruction is fetched from the program memory

2 decoding of the instruction is performed that provides the control signals to operate the parts

3 data memory is used to load the operand

4 a comparison is made between the loaded operand with the abscissa axis edges for correct approximation interval identification

5 ROM used to store the correct piecewise segment parameter is addressed using the previous comparison results

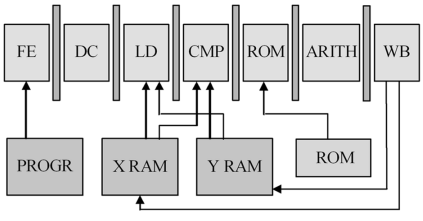
6 the output is computed using the fetched parameters as per the expressions of the piecewise segment

7 the output is stored back to the data memory.

Special features of architectures such as the multiple parallel processing engines in NPUs, address generation units in DSPs and multiple memory banks are the components

of the domain specific processors. It helps to accelerate the applications of the hardware domain.

Figure 1 Pipeline and memory organisations for the ASIP



4 Wavelet transform-based filtering

In this transformation the smaller difference terms coefficients are removed using filtering. The detailed coefficients $d_j(k)$ of DWT can be removed without any large difference made to the designated structure.

4.1 Discrete wavelet transform (DWT)

The function $g(s) \in A^2(P)$ can be defined using wavelet series expansion with respect to the mother wavelet $\theta(x)$ and the scaling function $\phi(x)$ as:

$$g(s) = \sum_k a_{j_0}(k)\vartheta_{j_0,k}(s) + \sum_{j=j_0}^{\infty}\sum_k d_j(k)\theta_{j,k}(s) \tag{3}$$

Here, we begin with an initial arbitrary scale j_0 with the approximation coefficients $a_{j_0}(k)$. These detailed and approximation coefficients can be expressed as:

$$a_{j_0}(k) = \left\langle g(s), \tilde{\vartheta}_{j_0,k}(s)\right\rangle = \int g(s)\tilde{\vartheta}_{j_0,k}(s)ds \tag{4}$$

$$d_j(k) = \left\langle g(s), \tilde{\theta}_{j,k}(s)\right\rangle = \int g(s)\tilde{\theta}_{j,k}(s)ds \tag{5}$$

To expand the continuous function samples function as a sequence of numbers DWT is used. The DWT transform pair can be expressed as:

$$W_\phi(j_0, k) = \frac{1}{\sqrt{M}}\sum_{s=0}^{M-1} g(s)\tilde{\vartheta}_{j_0,k}(s) \tag{6}$$

$$W_\theta(j, k) = \frac{1}{\sqrt{M}}\sum_{s=0}^{M-1} g(s)\tilde{\theta}_{j,k}(s) \tag{7}$$

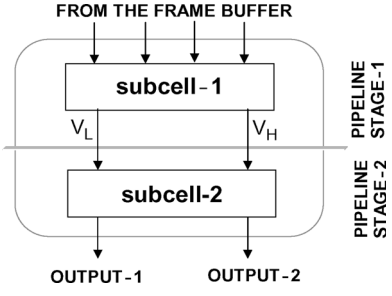
For $j \geq j_0$ and:

$$g(s) = \frac{1}{\sqrt{M}} \sum_{k} W_{\vartheta}(j_0, k)\, \vartheta_{j_0,k}(s) + \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_{k} W_{\theta}(j, k)\theta_{j,k}(s) \tag{8}$$

where $g(s)$, $\vartheta_{j_0,k}(s)$, and $\theta_{j,k}(s)$ are functions of discrete variable $s = 0, 1, 2, ..., M - 1$.

To implement the two-dimensional DWT generally digital filters and down samplers are used. Figure 2 provides the block diagram of the proposed structure.

Figure 2 Proposed structure for 2D DWT



Both the hard thresholding and soft thresholding filters are used.

1 Hard thresholding

In this the difference term can be treated as:

$d = 0$ *if* $|d| < \lambda$, *else it is untouched*

2 Soft thresholding

It is similar to above however, for all the remaining values in which $d > \lambda$ following condition is applied to the remained coefficients.

$$d \leftarrow \text{sgn}(d)(|d| - \lambda)$$

The literature suggests that λ is best set to $\frac{\sigma\sqrt{2\ln n}}{\sqrt{n}}$. In this case σ (standard deviation) has been considered over all the difference terms. The term n corresponds to the number of difference terms. In this work, soft thresholding technique is considered.

5 DSPs-based ASIP design

Data-intensive applications that includes the internet and video browsing with mobile sets has made the application of DSPs ever increasing. The DSP processors while remains cheap and consume less power can satisfy the industry demand accurately. Although use of four processors on a board is not advisable due to size limitation however, feature size shrinkage has made space for single-chip DSPs having the desired memory as well as I/O interface. The cost can be reduced further with system-on-chip for simplified board design.

Recent developments of field programmable gate arrays (FPGAs) have opened up new avenues to DSP design platform. It provides a better solution in critical time-to-market or in crucial design adaptability conditions. A using parallel structures and arithmetic algorithms, DSP system designed using FPGA will be benefited. The performance can be improved with the help of multiple general-purpose DSP kits. To enhance the data bandwidth, distributed arithmetic for FPGA array multiplication can be a potential solution that provides abetter throughput because of off-the-shelf DSP algorithms. In this regard, the TI TMS320C6x supports both the floating and fixed point as it is composed of several processors and also supports VLIW (Peters et al., 2015). The block diagram of standard VLIW architecture is shown in Figure 3 and the DSP algorithm process is shown in Figure 4.

Figure 3 Basic VLIW units for embedded systems

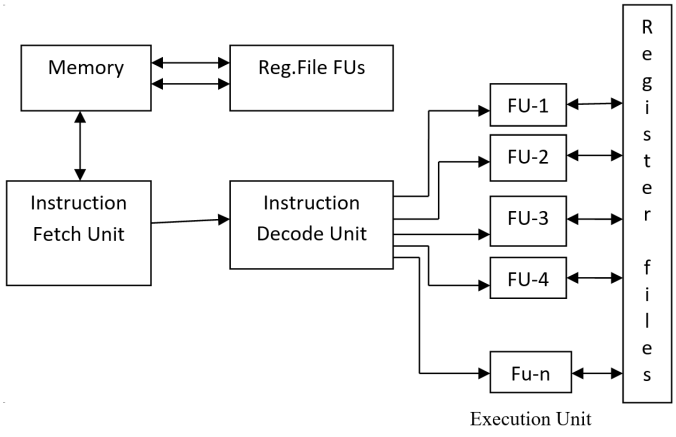
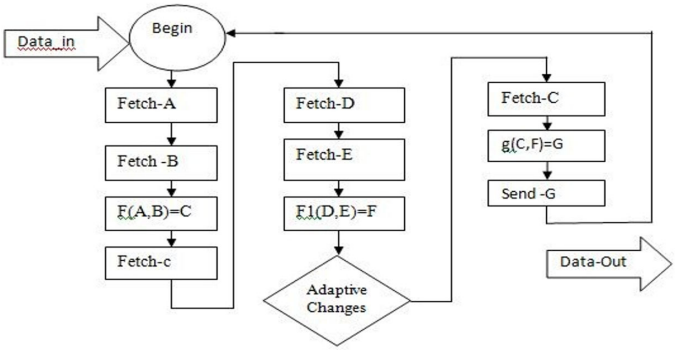


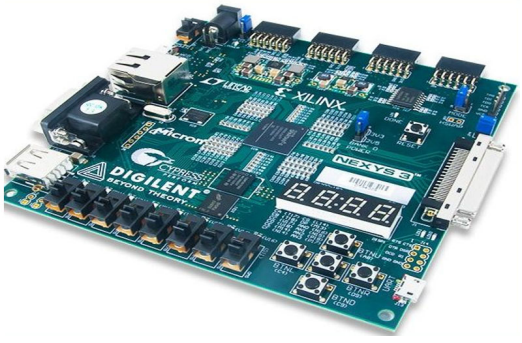
Figure 4 DSP algorithm process



Unlike normal instruction having mnemonic and register/memory operands, ASIP architecture is controlled using the instruction-set format that may utilise either general purpose or configuration registers. While the RISC processor's instructions usually triggers both the functional units as well as the general purpose registers whereas the ASIP takes into account the configuration registers. These configuration registers utilises specific data flow configuration which are hardwired in the concerned system.

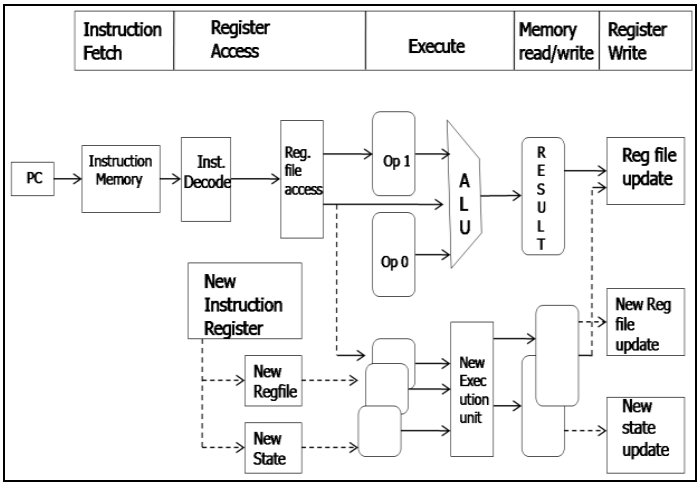
With the help constant operands this register can eliminate the requirement to encode the corresponding addresses in the instruction word similar to other general purpose registers. Figure 5 shows the basic Xilinx Spartan board.

Figure 5 Xilinx Spartan board (see online version for colours)



A major benefit of using ASIPs is that the instructions set are able to be customised as per the application domain. It helps to trade-off among the silicon area, computational performance, and energy consumption. To improve the efficiency of the architecture complex pipelined instructions need to be implemented. This reduces the final assembly program length which helps in our case as it is emphasised on the number of clock cycles essential for complete elaboration. As the work focuses on image/video processing a portion of the assembly program has been repeated hundreds of thousands of times based on the size of the image, i.e., a single iteration has been used per pixel. Figure 6 provides the custom instruction integration in ASIP.

Figure 6 Custom instruction integration in ASIP



6 Simulation results

In this work, application of DWT has been made to the pixel blocks. It is a computation intensive phase similar to the decompression times. The images used are having a pixel size of 512 × 512 grey scale and are the standard available test images. In Figure 7 shows the following figures in sequence:

a original image

b noisy image with noise density of 30% SPN; filtered images

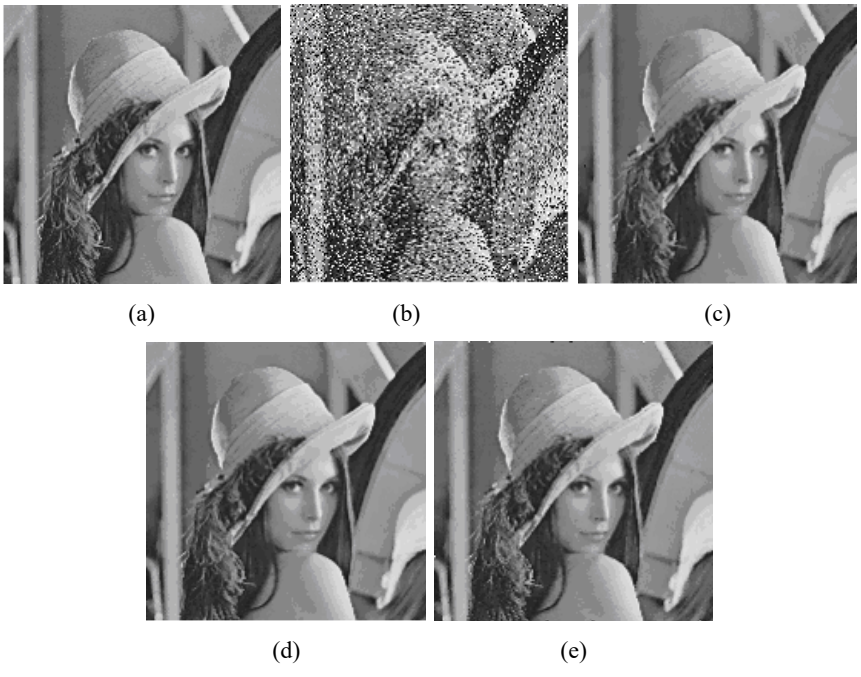
c DWT filter with mask 3 × 3

d DWT filter with mask 5 × 5

e DWT filter with mask 7 × 7.

It has been found that the DWT algorithm run time is below one second. This is observed that DWT with Haar wavelet need not require the DCT transforms.

Figure 7 (a) Original image (b) Noisy image with noise density of 30% SPN; filtered images (c) DWT filter with mask 3 × 3 (d) DWT filter with mask 5 × 5 (e) DWT filter with mask 7 × 7



6.1 Implementation using VHDL

Figure 8 shows application of Haar transform schematically to achieve the desired objective.

The 64 8-bit lines has been used to fed the data is into the module to lessen the delay during data fetching stage of the module. For simulation purpose, we have used an 8 × 8 image. Figure 9 shows simulation result of a 20 ns with satisfactory result. It shows many non-zero coefficients occupying the matrix. This is due to the reason that there is an absence of any threshold co-efficient. In this work the detail coefficients are not set to zero.

Figure 8 Hardware implementation using 2D-Haar transform

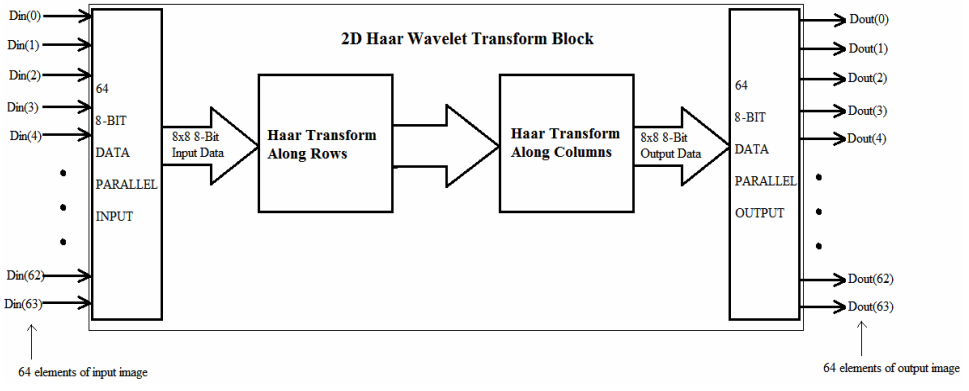
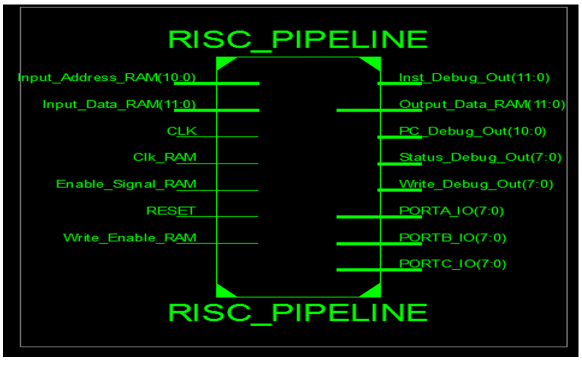


Figure 9 Test bench simulation results (see online version for colours)



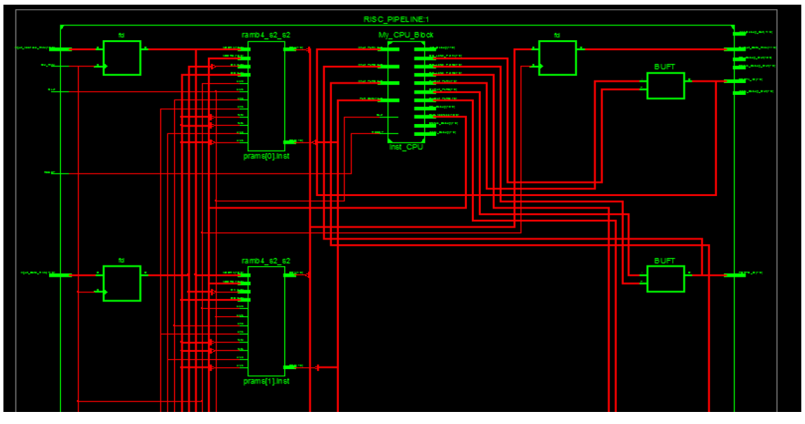


Table 1 and Table 2 provide the statistical report of the proposed design and the cell usage report of the proposed design respectively whereas Figure 7 shows the internal structure of DWT filter.

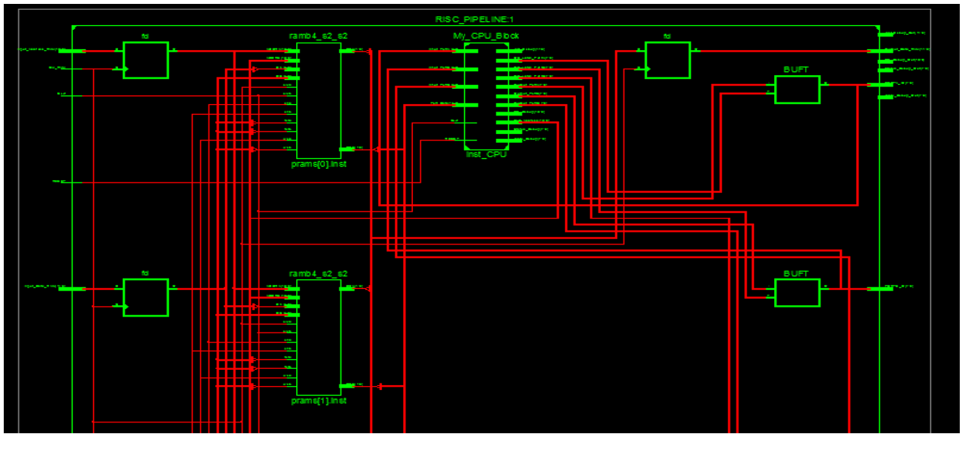
Table 1 The statistical report of the proposed design

Statistics	*Numbers*
Registers	50
8-bit register	45
1-bit register	8
Comparators	18
8-bit comparator greater	7
8-bit comparator great equal	10

Table 2 The cell usage report of the proposed design

Design statistics	*Cell usage*
IOs	83
BELS	1,924
AND2	749
AND3	46
INV	612
OR2	498
OR3	11
OR4	8
FlipFlops/latches	385
FD	385
IO buffers	83
IBUF	74
OBUF	9

Figure 10 RTL of DWT filter (see online version for colours)



7 Conclusions

The work provides the list of the ASIP architecture design with respect to the memory organisation, architecture pipelining as well as architecture customisation. A hardware-efficient systolic-like architecture using 2D DWT has been designed with the help of a new data-access scheme that includes a novel folding approach. The complete computation has been decomposed into two separate stages which are concurrently implemented in a modular structure that comprises of two fully-pipelined sub-cells. It allows free realisation of 2D DWT. A 2D Haar wavelet transform without multiplier has been implemented that has reduced the complexity of the system with respect to area and time. This is due to the fact that implementation using DWT remains computationally expensive. In future, more simple algorithm can provide better solution to such issue.

References

Bauer, L., Shafique, M. and Henkel, J. (2017) 'A computation-and communication-infrastructure for modular special instructions in a dynamically reconfigurable processor', in *Proc. Int. Conf. on Field Programmable Logic and Applications*, IEEE, pp.203–208.

Brown, S. and Rose, J. (2016) 'FPGA and CPLD architectures: a tutorial', *IEEE*, Vol. 13, No. 2, pp.42–46.

Chan, R.H., Ho, C-W. and Nikolova, M. (2015) 'Salt and pepper noise removal by median type noise detectors and detail preserving regularization', *IEEE Trans. Image Process.*, October, Vol. 14, No. 10, pp.1479–1485.

Chen, R., Jia, Z., Li, Y., Xia, H. and Li, X. (2016) 'The application specific instruction processor for AES', in *3rd Int. Conf. Electronics Computer Technology*, pp.394–396.

Digital Signal Processing (2015) *Wikipedia* [online] http://en.wikipedia.org/wiki/Digital_signal _processor.

Engblom, J. and Ermedahl, A. (2015) 'Pipeline timing analysis using a trace-driven simulator', in *Proc. 6th International Conference on Real-Time Computing Systems and Applications (RTCSA)*, IEEE Computer Society Press, December.

Gonzalez, R.C. and Woods, R.E. (2016) *Digital Image Processing*, 3rd ed., Prentice Hall.

Goslin, G.R. (2010) *A Guide to Using Field Programmable Gate Arrays (FPGAs) for Application-Specific Digital Signal Processing Performance*, Xilinx Inc.

Hoare, R.R. et al. (2016) 'Rapid VLIW processor customization for signal processing applications using combinational hardware functions', *EURASIP Journal on Applied Signal Processing*, ID 46472, Vol. 71.

Hoffmann, A. et al. (2012) *Architecture Exploration for Embedded Processors with LISA*, Kluwer Academic.

Jain, A.K. (2009) *Fundamentals of Digital Image Processing*, Prentice-Hall, Englewood Cliffs, NJ.

Jain, M.K., Balakrisnana, M. and Kumar, A. (2015) 'ASIP design and methodologies: survey and issues', *Proceedings of IEEE*, VLSI, pp.76–81.

Karam, L.J., AlKamal, I., Gatherer, A., Frantz, G.A., Anderson, D.V. and Evans, B.L. (2016) 'Trends in multicore DSP platforms', *Signal Processing Magazine*, IEEE, November, Vol. 26, No. 6, pp.38–49.

Nohl, A., Schirrmeister, F. and Taussig, D. (2010) 'Application specific processor design: architectures, design methods and tools', *2010 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, November.

Peters, H. et al. (2015) 'Application specific instruction-set processor template for motion estimation in video applications', *IEEE Trans. Circuits and System for Video Tech.*, April, Vol. 15, pp.508–527.

Pitas, I. and Venetsanopoulos, A. (2010) *Nonlinear Digital Filters: Principles and Applications*, Kluwer, Boston, MA.

Probell, J. (2017) 'Architecture considerations for multi-format programmable video processors', *J. Signal Process. Syst.*, Vol. 50, No. 4, pp.33–39.

Saponara, S. et al. (2017) 'Cost-effective VLSI design of non linear image processing filters', *IEEE DSD*, pp.322–329.

Schliebusch, O. et al. (2015) 'A framework for automated and optimized ASIP implementation supporting multiple hardware description languages', *IEEE ASP-DAC05*.

Schneider, J. and Ferdinand, C. (2017) 'Pipeline behaviour prediction for superscalar processors by abstract interpretation', in *Proc. SIGPLAN Workshop on Languages, Compilers and Tools for Embedded Systems (LCTES'99)*, ACM Press, May.

SPRUGH7 (2010) *TMS320C66x DSP CPU and Instruction Set*, Texas Instruments, November.

SPRUGW0B (2010) *TMS320C66x DSP CorePac*, Texas Instrument User Guide, November.

SPRY147 (2010) *TI's New TMS320C66x Fixed- and Floating-Point DSP Core Conquer the Need for Speed*, Texas Instruments, November.

Texas Instruments TMS320 (2016) *Wikipedia* [online] http://en.wikipedia.org/wiki/TexasInstrumentsTMS320#C6000 Series.

Windyga, P.S. (2016) 'Fast impulsive noise removal', *IEEE Transactions on Image Processing*, January, Vol. 10, No. 1, pp.370–381.

Wong, S., Vassiliadis, S. and Cotofana, S.D. (2014) 'Future directions of (programmable and reconfigurable) embedded processors', in *Domain-Specific Processors: Systems, Architectures, Modelling, and Simulation Workshop*, pp.235–257.

Xilinx Inc. (2015) *Homepage* [online] http://www.xilinx.com.