

Mean and variance statistic for image processing on FPGA

Sunny Arief Sudiro^{1*}, Aqwam Rosadi Kardian¹, Sarifuddin Madenda², Lingga Hermanto²

¹ STMIK Jakarta STI&K Jl. BRI No. 17 Radio Dalam Kebayoran Baru Jakarta Indonesia

² Gunadarma University Jl. Margonda Raya No. 100 Depok Jawa Barat Indonesia

ABSTRACT

Statistical formula processing an image data is commonly used in image processing. In software processing this formula and accessing data stored in memory is an easy task, but in hardware implementation, it is more difficult task due to many of constraints. This article presents hardware implementation of mean & variance statistic formula in effective and efficient way using FGPA Device. The design of circuit for both formulas proposed in this article need only two additions component (in two accumulators) and two shift-right-registers will be used for divisor circuits, one subtractor and one multiplier. In the experiment, processing an image size 8x8 pixels need 64 clocks cycle to conclude the mean & variance calculations. More than 1024 additions component is needed in some design so this design is more efficient.

Keywords: Mean, Variance, FPGA, Accumulator, Counter.

1. INTRODUCTION

A computer process for manipulating and analysing image is known as digital image processing. Basic statistic formulas that used in image processing using computer application are: histogram, variance, mean, etc. These all processes have high correlation with functions in pattern recognition based on features to recognize the object.

Many research for the implementation of function involving large data and statistical calculation in FPGA device. Implementation variance value in image fusion using FPGA device, computation speed is one of the reason among others. This approach based on FPGA technology provides fast, compact and low power consumption for image fusion (Gade and Khope, 2016).

The computation time or speed of process is one of the important things in processing large of data. Hardware implementation is one of solution, acceleration using FPGA presented in Iturk et al. (2008) for processing Markowitz' mean variance framework. This approach gives a 221x speed ratio comparing to software implementation. Another research in FPGA acceleration is proposed in Betkaoui et al. (2011), this approach focus on re-configurable architecture or component in FPGA and partitioning data for processing large graph data, and the result is reducing execution time from 120 minutes to 12 minutes or 10 times faster for common bio-informatics algorithm.

Research for mean calculation only using FPGA device is proposed by Kardian et al. (2016). In this article presents a method for mean calculation needs 1 addition operations, 1 division, 64 cycles for 8x8 image size. This work is actually the extended work based on previous method describe in Kardian et al. (2016).

An interesting work is presented in Ismael and Mahmood (2017), this research proposed an implementation 6 functions of statistical operation based on FPGA, which are: mean, variance, standard deviation, Root Mean Square (RMS), covariance and Mean Square Error (MSE). This approach used 1090 number of slices, 220 numbers of


OPEN ACCESS

Received: June 5, 2020

Accepted: December 31, 2020

Corresponding Author:

Sunny Arief Sudiro
sunnyarief@yahoo.com

 **Copyright:** The Author(s). This is an open access article distributed under the terms of the [Creative Commons Attribution License \(CC BY 4.0\)](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted distribution provided the original author and source are cited.

Publisher:

[Chaoyang University of Technology](https://www.chaoyang.edu.cn/)

ISSN: 1727-2394 (Print)

ISSN: 1727-7841 (Online)

Slice Flip-Flops and 1988 number of LookUp Tables (LUTs) for all 6 function of calculation. For mean calculation the result has an error 1.062 % and for variance calculation is 0.193 %.

Another method in hardware implementation for mean & variance calculation in an iterative manner is described in Bailey and Laiber (2013). This method has been implemented efficiently in hardware based architecture. This method distributes the divisions over several iteration steps not combinational divisions. This approach enables calculation operation of running statistics formula using simple elements in hardware. For 0.3 MP using HW shifter, this design used 131 of registers and 597 of LUT's.

2. MEAN AND VARIANCE STATISTIC FORMULA IN IMAGE PROCESSING

Diagram that drawn based on frequency of the appearance for every intensity value from the whole image pixel element known as histogram diagram (Martinez and Martinez, 2002; Woods et al., 2005). The first-order of statistical analysis calculation known as mean value and the 2nd-order is variance value. These values usually used for image feature extraction and segmentation processes. Mean value represent intensity of the image from the entire

pixel, while contrast of the image is variance value (Kardian et al., 2016). Equation (1) is formula for 'mean' (μ), and Equation (2) σ^2 is the normalized 2nd order statistical formula, based on these feature can be used to show level of contrast from the image (Martinez and Martinez, 2002).

$$\mu = \sum_{i=0}^L i.p(i), \text{ where } p(i) = \frac{H(i)}{NM} \quad (1)$$

$$\sigma^2 = \sum_{i=0}^{255} (i - \mu)^2 p(i) = \frac{1}{N \times M} \sum_{i=0}^{255} (i - \mu)^2 H(i), \quad i = f(n, m) \quad (2)$$

3. PROPOSED ALGORITHM

Based on Equations (1) and (2), for mean and variance value calculation, we can optimize these equations to Equations (3) and (4), and then propose the implementation using FPGA.

$$\mu = \sum_{i=0}^{255} i \cdot p(i) = \frac{1}{N \times M} \sum_{n=0}^N \sum_{m=0}^M f(n, m) \quad (3)$$

$$\sigma^2 = \left(\frac{1}{N \times M} \sum_{n=0}^{255} \sum_{m=0}^{255} i^2 \right) - \mu^2, \quad \text{if } i = f(n, m) \quad (4)$$

Pseudo code 1. Mean formula (1st version in Equations 1 and 2)

```
(1) f = % data %
(2) h = zeros(256,1);
(3) [N,M] = size(f);
(4) for i = 1:N % Histogram H(i) calculation%
(5)     for j = 1 : M
(6)         h(f(i,j) + 1) = h(f(i,j) + 1) + 1;
(7)     end
(8) end
(9) N1 = (N*M);
(10) for i = 1 : 256 % Probability p(i) calculation%
(11)     p(i) = h(i)/N1;
(12) end
(13) Mean = 0;
(14) for i = 1 : 256 % Mean μ calculation%
(15)     Mean = Mean + p(i) * (i-1);
(16) end
(17) Mean
```

Pseudo code 2. Mean and variance formula (proposed version in Equations 3 and 4)

```
(1) [M,N,L] = size(f); % f is the data
(2) htsum = 0;
(3) for i = 1:M; % Mean μ calculation%
(4)     for j = 1:N;
(5)         htsum = htsum + f(i,j);
(6)         vsum = vsum + ( f(i,j))^2;
(7)     end;
(8) end;
(9) optimalmean = htsum/(M*N).
(10) varians = vsum/(M*N)-optimalmean^2 % variance σ² calculation
```

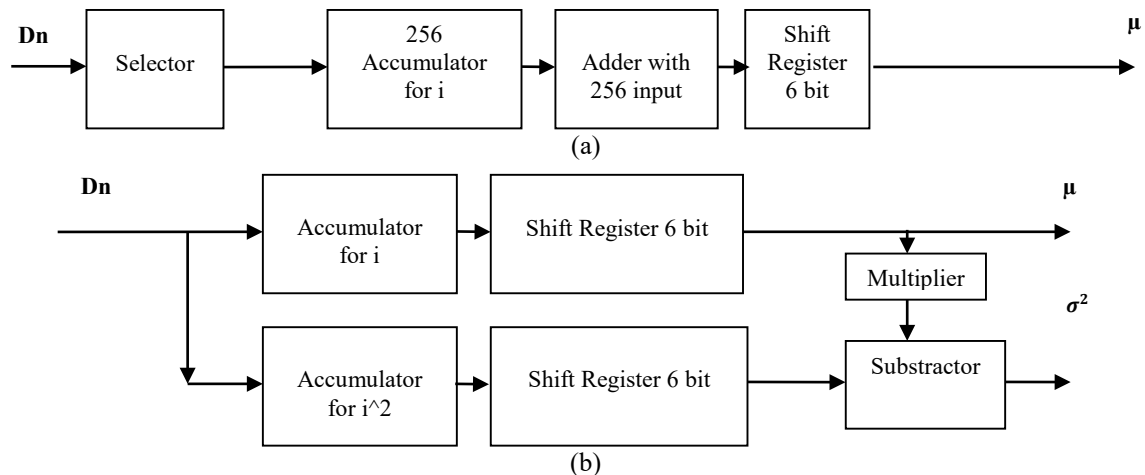


Fig. 2. The component block diagram (a) First design (Kardian et al., 2016) and (b) Second/proposed design

Pseudo code-1 is the pseudo code for variance calculation based on Equation (1) and (2). The three steps of calculations are: histogram $H(i)$, probability $p(i)$ and the last one is mean value $= \mu$. Pseudo code-2 (based on Equation (3) & (4)) is the algorithm of the variance formula (variance formula (σ^2)), see Equation (2). It is only need one step in this algorithm. The algorithm based on optimize formula reduces steps of calculation (and reduce time of operation, 2 x 256 clock cycles) and number of arithmetic operations e.g. addition, multiplication and division.

Fig. 1 shows mean & variance calculation result using basic algorithms (mean = 14.5156 variance = 235.7185) and optimized algorithms (optimalmeanrounded = 14, variansrounded = 250) for the similar data (Im), and obtain the same result. The result from the Matlab evaluation is used for comparison with values from FPGA components processes that concern only in fix value not the floating point value (including "optimalmeanrounded" and "variansrounded" values).

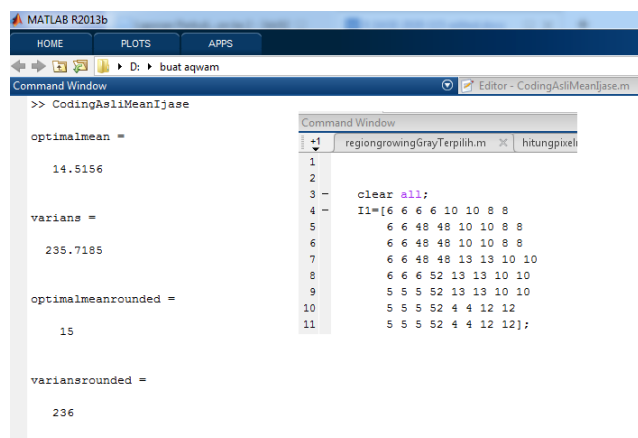


Fig. 1. Result in Matlab based on both value

4. HARDWARE IMPLEMENTATION

Fig. 2(a) is the design using pseudo code-1, this design use a selector for selecting the intensity value of pixel from the image. This intensity values are 0 to 255 and then send to their each accumulators (there are 256 accumulators with 256 addition components). All these accumulators value will be used as an input to 256 addition (256 additions are needed) so 512 additions are needed in this method (Kardian et al., 2016). And finally to get mean value, we divided with the number of pixels in this example is $8 \times 8 = 64$. A right register 6 bits is used to conclude the division operation. Fig. 2(b) shows the design of component based on optimized formula, and at the end reduce the addition and multiplication process. Dn is the input data stream coming from image that want to be processed.

Fig. 3 is entity diagram of the component and Fig. 4 is the schematic diagram of the component. This proposed design is the corrective design based on diagram in Fig. 2(b). For mean and variance value calculation, we don't have to calculate histogram, but in the same operation using one accumulator to keep the total value of histogram (htsum), see the pseudo code-2.

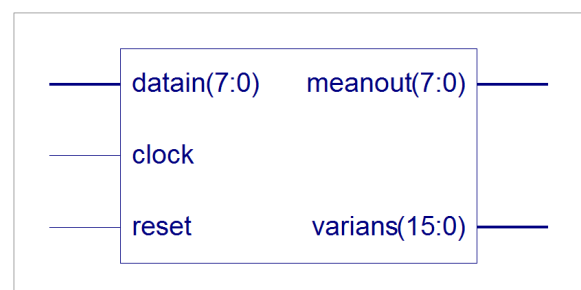


Fig. 3. The entity of component

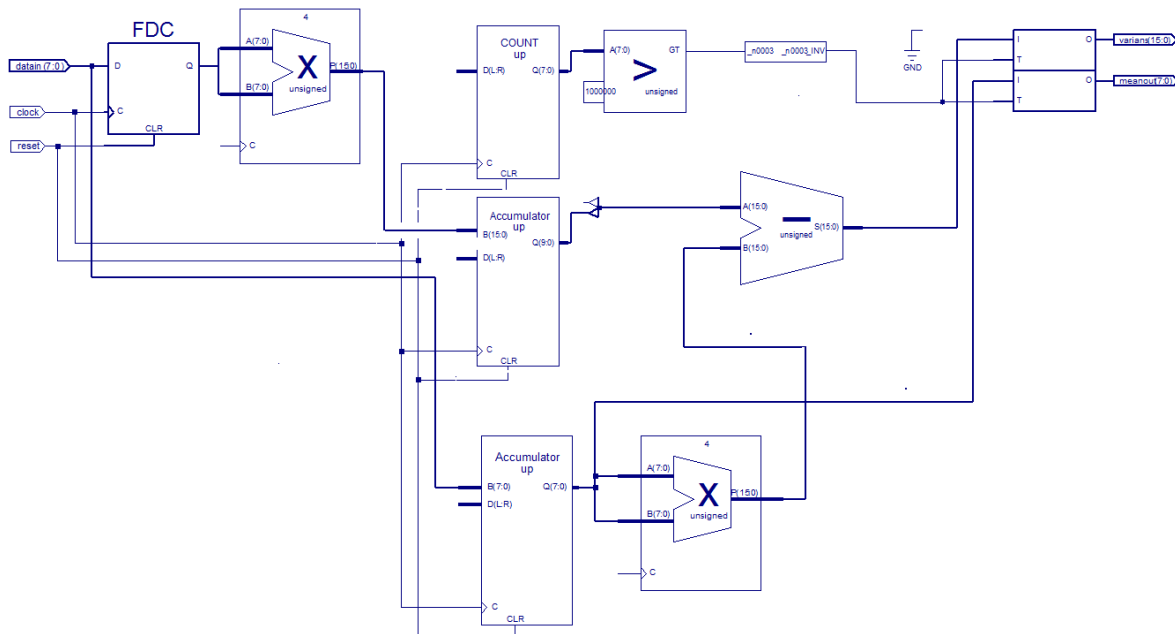


Fig. 4. Component RTL schematic diagram

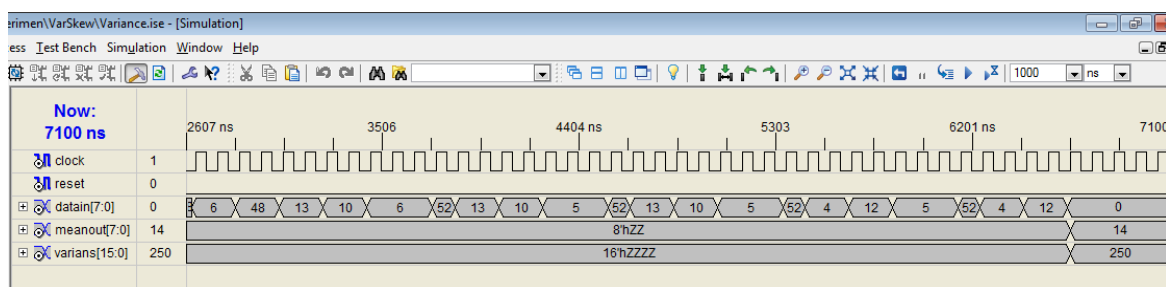


Fig. 5. Behavioural simulation results

We use one other accumulator for keeping 2nd order value of the pixel element (i^2). This method makes the mathematical operation reduction which imply in reducing time of operations and components. Components in the design are: two additions in two accumulators, two shift right register, multiplier and subtractor. This proposed design has the same processing time ($N \times M = 64$ clock cycles) for 8×8 image size and the result of mean and variance value are similar. Counter and buffer are used to keep the values and after 64 clocks send the final value of mean and variance.

From Fig. 4, we can see the schematic digram of component, there are one multiplier for (i^2) and then send to upper accumulator (second order value). In the lower accumulator is to hold (htsum) value. Counter and comparator is used to determined whether the operation is completed after 64 cycles (for image size 8×8) and send the correct value of mean and variance. We use data mapping for Shift Register in Fig. 2, when division is needed (see two square block on the right of Fig. 4). For example in VHDL code this formula is shown in Equation (5).

$$A \leq B(27 \text{downto } 6); \quad \%A = B / 64 \quad (5)$$

Finally this design only needs : one DFF, two multiplier, one counter, one comparator, two adder in two accumulator, one subtractor and two buffer with data mapping.

5. SIMULATION RESULT

10 images with 8*8 size and grayscale of pixel value (0-255) are used in simulation. Fig. 5 show the behavioural simulation in ISE software simulation result. From this figure we can see the result for calculating the same data in Matlab operation, the result are 14 (mean) and 250 (variance). All processes are done when 64 clock cycles are completed. The processes are simultaneously with the coming of data input (data in). Comparing to the Matlab evaluation result, mean value is 14.5, it means there is difference value which is 0.5 or an error of $((0.5/14)*100\% = 3.575 \%)$. For variance value the Matlab result is 235.72 with the difference is 14.28 or there is an error of 6.05 % $(14.28/235.72*100\%)$.

Table 1. Experiment Result for 10 data matrix

No	Matlab Result		FPGA Result		Error	
	Mean	Variance	Mean	Variance	Mean	Variance
1.	161.1563	1245.7000	161	1245	0.10%	0.06%
2.	58.4219	79.7126	58	79	0.72%	0.89%
3.	105.4844	994.7490	105	994	0.46%	0.08%
4.	136.2969	156.9900	136	156	0.22%	0.63%
5.	138.4844	49.2881	138	49	0.35%	0.58%
6.	67.1563	394.1000	67	394	0.23%	0.15%
7.	35.6250	203.0449	35	203	1.75%	0.02%
8.	76.7344	142.6700	76	142	0.96%	0.47%
9.	22.4688	107.7178	22	107	2.09%	0.67%
10.	203.3594	18.1365	203	18	0.18%	0.75%
Average Error Rate					0.71%	0.43%

Table 2. Comparing to other research

No	Properties	Previous Approach (Kardian et al., 2016)	Previous Approach (Ismael and Mahmood, 2017)	Previous Approach (Bailey and Laiber, 2013)	Proposed Approach
1.	Number of Slice	16	1090	NA	30
2.	Number of Slice Flip-Flops	24	220	NA	38
3.	Number of LUTs	31	1988	597	53
4.	Function	Mean	6 function	Mean & Variance	Mean & Variance

VARIANCE Project Status			
Project File:	Variance.ise	Current State:	Programming File Generated
Module Name:	mean2	• Errors:	No Errors
Target Device:	xc3s500e-4ft256	• Warnings:	7 Warnings
Product Version:	ISE, 8.1i	• Updated:	Sen 5. Sep 04:11:45 2016

Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	38	9,312	1%	
Number of 4 input LUTs	53	9,312	1%	
Logic Distribution				
Number of occupied Slices	30	4,656	1%	
Number of Slices containing only related logic	30	30	100%	
Number of Slices containing unrelated logic	0	30	0%	
Total Number 4 input LUTs	59	9,312	1%	

Fig. 6. Logic utilization summary

This error value is calculated based on Equation (6) and in Table 1 all error value from 10 data of experiment is presented.

$$ErrorValue = \frac{|FPGAValue - Matlabresult|}{Matlabresult} \times 100 \% \quad (6)$$

Fig. 6 shows the components design summary and the occupation of logic element information in FPGA device. Experiment result for other 10 data matrix (comming from 10 blox or crop images) can be seen in Table 1, we can see that the difference is very small less than 0.99.

Based on Table 1. we calculate mean square error (MSE) for mean and variance value using formula in Equation (7). We obtain MSE for mean value is 0.175 and variance value is 0.3347.

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y' - Y)^2 \quad (7)$$

$$\begin{aligned} \text{MSE}_{\text{Mean}} &= \frac{1}{10} ((161.1563 - 161)^2 + (58.4219 - 58)^2 \\ &+ (105.4844 - 105)^2 + (136.2969 - 136)^2 \\ &+ (138.4844 - 138)^2 + (67.7344 - 67)^2 \\ &+ (35.6250 - 35)^2 + (76.7344 - 76)^2 \\ &+ (22.4688 - 22)^2 + (203.3594 - 203)^2) \\ \text{MSE}_{\text{Mean}} &= 0.175 \\ \text{MSE}_{\text{Variance}} &= \frac{1}{10} ((1245.70 - 1245)^2 + (79.7126 - 79)^2 \\ &+ (994.7490 - 994)^2 + (156.9 - 156)^2 \\ &+ (49.2881 - 49)^2 + (394.100 - 394)^2 \\ &+ (203.0449 - 203)^2 + (142.67 - 142)^2 \\ &+ (107.7178 - 107)^2 + (18.1365 - 18)^2) \\ \text{MSE}_{\text{Variance}} &= 0.3347 \end{aligned}$$

Comparing to previous result (Kardian et al., 2016), our approach needs little bit more FPGA resources but for two function (mean and variance calculation, not only “mean” calculation), see Table 2. Comparing with result from Bailey and Laiber (2013) this approach needs smaller resources, NA here mean that there are no information regarding the use of FPGA Slice and FPGA Slice Flip Flops in this article. Approach from Ismael and Mahmood (2017) is for 6 function, if we calculate average resource used in this approach we obtain $1090/6 = 181.6$ slice, $220/6 = 36.6$ Flip-Flops and $1988/6 = 331.3$ of LUT for each function, our result still need smaller resources.

6. CONCLUSION

Hardware base component using FPGA device for efficient mean and variance calculation has been obtained. This approach needs two additions and two shift right registers. For 8x8 image size 64 clock cycles is needed to calculate the value of mean and variance. Overall components need only 53 of 4 input LUTs and 38 flip-flops slices. The result difference (error) of variance value comparing to Matlab result for 10 data of experiment is 0.43 % because of floating point constraint in FPGA.

ACKNOWLEDGMENT

This research is as part of research dissertation and supported by Yayasan Pendidikan Gunadarma and Yayasan Ilmu Komputer Jakarta.

REFERENCES

Bailey, D.G., Laiber, K.M.J. 2013. Efficient hardware calculation of running statistics. 28th International

- Conference on Image and Vision Computing New Zealand (IVCNZ 2013), 1–6.
- Betkaoui, B., Thomas, D.B., Luk, W., Przulj, N. 2011. A framework for FPGA acceleration of large graph problems: Graphlet counting case study. International Conference on Field-Programmable Technology, 1–8.
- Gade, P.B., Khope, S.R. 2016. FPGA based multifocus image fusion using variance Method. Irjet International Research Journal of Engineering and Technology (IRJET), 3.
- Irturk, A., Benson, B., Laptev, N., Kastner, R. 2008. FPGA acceleration of mean variance framework for optimal asset allocation. Workshop on High Performance Computational Finance at SC08 International Conference for High Performance Computing, Networking, Storage and Analysis, 1–8.
- Ismael, S.F., Mahmood, B.S. 2017. A novel way to design and implement statistical operations based on FPGA. International Journal of Computer Applications (0975–8887), 167.
- Kardian, A.R., Sudiro, S.A., Madenda, S. 2016. Efficient implementation of mean formula for image processing using FPGA device. 1st International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE), Yogyakarta, Indonesia, ISBN: 978-602-60280-0-6.
- Martinez, W.L., Martinez, A.R. 2002. Computational statistics handbook with MATLAB. Chapman & Hall/CRC, USA.
- Woods, R.E., Eddins, S.L., Gonzales, R.C. 2005. Digital image processing using MATLAB. Pearson Education.