MACHINE INSTRUCTION ANALYSIS FOR DCT ALGORITHM USING DLX ARCHITECTURE

Believa Dyanneley Aritspoundy[1], Raditiana Patmasari[2], Nyoman Bogi Aditya Karna[3]
[1] Telecommunication Engineering , Faculty of Electrical Engineering , Telkom University
[1]believadyanne@student.telkomuniversity.ac.id,
[2]raditiana@telkomuniveristy.ac.id, [3]aditya@telkomuniversity.ac.id

Abstract

To reduce size of images, compressing images are required. This thesis presents a proof about the advantages using DeLuXe (DLX) microprocessor to do image compression and ASIP which help to reduce the power of microprocessor to save more energy for long term using. The simulation using winDLX processor which has to show the algorithm for Discrete Cosine Transform (DCT) in image compression process, coded in assembly DLX programming language which is MIPS.

The result of the programs made to simulate DCT using DLX microprocessor requires total of 14763 cycles executed with total of 5920 instructions. The instructions which are often used in this experiment is Load Float which is used to load the value of matrices before being store to the memory and multiplied to other matrices because mostly the matrices in this simulation consist of decimal numbers.

Keywords: ASIP, Discrete Cosine Transform, DLX microprocessor.

1. Introduction

Internet of Things (IoT) refers to the ever-growing network of physical objects which has connection to the internet to communicate. The IoT itself help a lot for people's daily activities to finish people's job easily without having a real-interface. The aim of this thesis is making an efficient microcontroller work as CCTV (security and surveillance IoT implementation) to report the situation around itself by sending images as report which have been compressed and simulated in DLX Microprocessor with Application Specific Instruction set Processor (ASIP) instructions which give benefit to less the power and less storage consumptions [9].

This thesis is focusing on Discrete Cosine Transform (DCT) Application in JPEG image compression that suppose to work by separating images into parts of differing frequencies. The instruction of ASIP designed to accelerate most used and huge data functions which helps to solve the problem by reducing the power or battery usage.

The DLX Microprocessor is 32-bit RISC CPU designed by John Hennessy and David Patterson. The RISC processor use simple instructions which is only three instruction formats to be executed within one clock cyle. Its load / store architecture is distinguished by its pipe performance, the use of a basic load / store instruction set and its reliability as a compiler objective. Load and store operations transfer the data between the general purpose registers to memory.

This thesis topic also related to the previous paper of one of the senior in Telkom University

with title "Evaluation Of DLX Microprocessor Instructions Efficiency For Image Compression" written by Nyoman Karna, Nimas Fatihah from Telkom University , Indonesia and Dong-Seong Kim from Kumoh National Institute of Technology in Gumi, Korea [11].

1.2 Problem Formulation

The problem of Internet of Things devices nowadays are a low power efficiency and high data storage consumption caused by those devices are still using a general instructions of microprocessor. This thesis give solution to cut those data storage consumption and create an efficient micro controller simulated in winDLX program with Application Specific Instruction set Processor (ASIP) machine instructions.

2. Basic Concept

2.1 Microprocessor

A microprocessor is a CPU made on small chips that can perform logic and arithmetic operations and can communicate with other connected devices. This mechanism is called the Arithmetic and Logic Unit (ALU) which is a programmable chip with microprocessor operations such as adding, comparing, reducing, and taking numbers. The microprocessor transfers data between itself and the system of input / output, simple arithmetic and logical operations, and program flow as well.

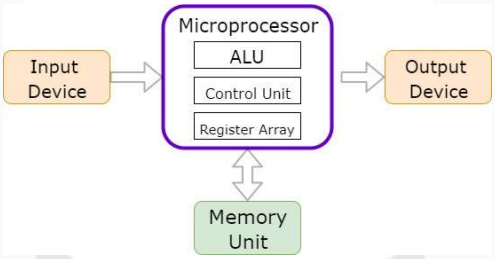

Figure 2.1 Microprocessor Architecture

2.2 DLX Microprocessor

The DLX Microprocessor is 32-bit RISC CPU designed by John Hennessy and David Patterson. The RISC processor use simple instructions which is only three instruction formats to be executed within one clock cycle [7]. Load and store operations transfer the data between the general purposes registers to memory. The memory contains half-word or word, and performs byte. There is aligned way for data and instructions which are words and other instructions must start at addresses (offset) multiple of four and for half words must start at

even byte addresses.

2.3 ASIP

Application Specific Instruction Set Processor (ASIP) is a programmable microprocessor, where instruction set is designed with efficient power to implement assembly instructions. ASIP is the best option for the domain of image processing that shows how it can significantly improve energy efficiency [6]. The aim of ASIP in this project is to design and optimize for a specific application which reduces the efficiency of power consumption.

2.4 Image Compression

Image compression is to compress the size of an digital image file by removing or reducing the repetitive data sequences from the original size to enable the effective storage of image. There are two different compression techniques are used for image compression, that are lossy and lossless compression. Lossless compression is a method used to reduce the bit rate of a file with the smallest possible number of bits while maintaining the same quality as before it was compressed without loss any information and distortion of the image. Lossy compression is the method which discards some parts of an original photo to save a little more bandwidth or storage space. However, this method does not modified the previous photo become bad. The process is irreversible, which means once you convert to lossy, you can not go back to the original one.

2.5 Discrete Cosine Transform

The DCT works by separating images into differences of frequency. In quantization step, the compression actually occur, only the most important frequencies that remain are used to retrieve the image in the decompression process [3]. DCT has many advantages which are has the ability to amount energy in the lower frequencies for image data and has the ability to reduce the blocking effect. The general equation for DCT 2D (N by M image) encoding is

$$F(u,v) = \left(\frac{2}{N}\right)^{\frac{1}{2}}\left(\frac{2}{M}\right)^{\frac{1}{2}}\sum_{i=0}^{N-1}\sum_{j=0}^{M-1} \Lambda(i).\Lambda(j).cos\left[\frac{\pi.u}{2.N}(2i+1)\right]cos\left[\frac{\pi.v}{2.M}(2j+1)\right].f(i,j)$$

2.6 JPEG

The Joint Photographic Experts Group (JPEG) simple coding scheme is the most common mode uses discrete cosine transform. The JPEG method used for black and white images and also for color images. The basic for JPEG algorithm is the DCT with various methods uses in this JPEG standards which are lossy compression and lossless compression method.

3. Work System

3.1 Flow Chart

The flow chart below shows how JPEG image is being compressed with Discrete Cosine Transform using DLX emulator. Data compression refers to the process of happen in this thesis focusing on Discrete Cosine Transform.

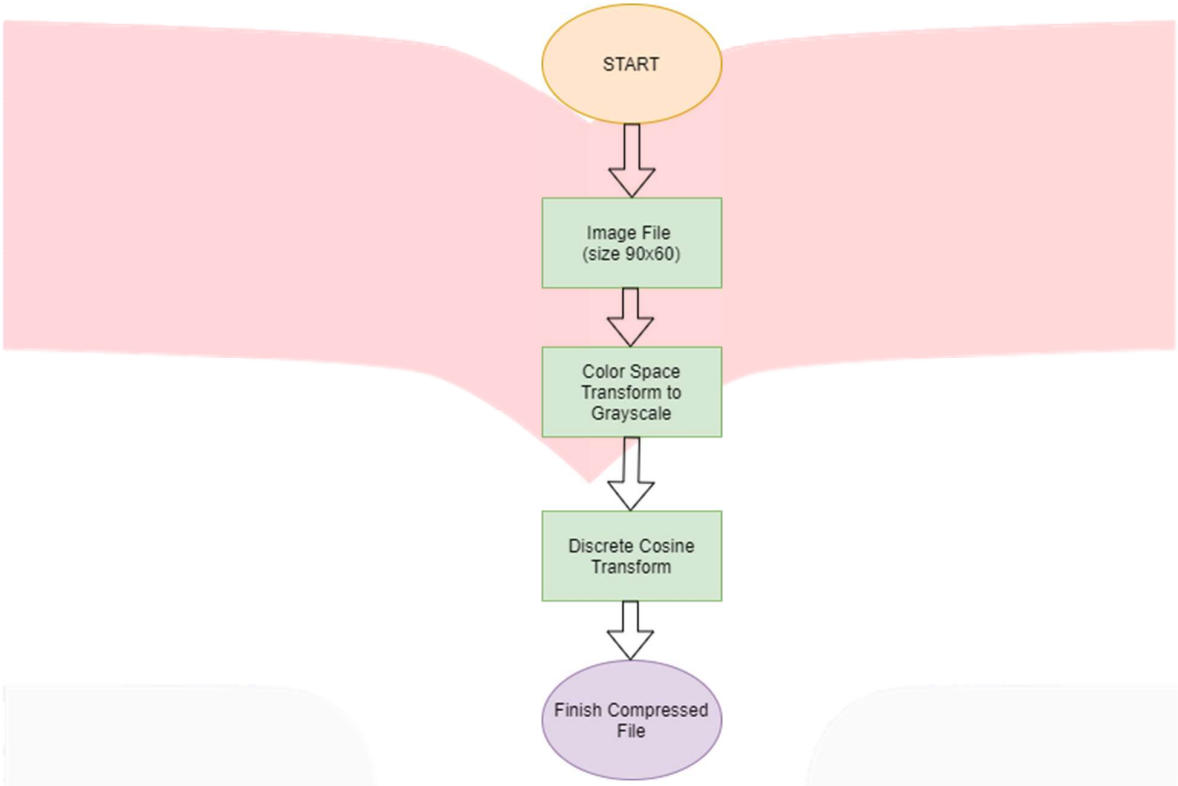

Figure 3.1 Image Compression Flow Chart.

In Figure 3.1 below shows the basic flow chart of image compression system. The first step is adding the image input to be compressed. Second step is color space which works as take an RGB image and convert to grayscale color. Next is DCT is to get other data representation and move from spatial to a frequency domain.

3.2 DCT on Matlab

The first step for DCT is to applied in original image with size of the image used here is 90x60 that caused the image shown smaller and blurry. Use Matlab to start measuring image pixel value of original or color image known as RGB image. The original image shown on Figure 3.2 below:

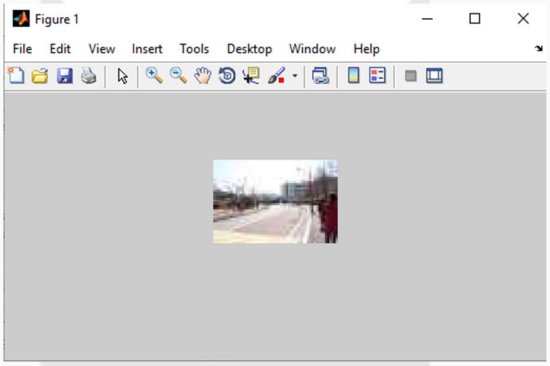

Figure 3.2 Original Image on Matlab.

Then the RGB layer broken down into grayscale layer which shown on Figure 3.3. As the grayscale image processed on Matlab, the details of 8x8 blocks image pixel value for original image shown and substracted by 128 resulting new matrix pixel value for grayscale image shown on Figure 3.3 below:

ORIGINAL IMAGE PIXEL VALUE

imgray <40x60 uint8>

	1	2	3	4	5	6	7	8
1	240	241	242	243	244	245	246	246
2	240	241	242	243	244	245	246	247
3	241	241	242	243	245	246	247	247
4	242	242	243	244	245	246	247	248
5	242	243	244	245	246	247	248	248
6	243	243	244	245	247	248	249	249
7	243	244	245	246	247	248	249	250
8	244	244	245	246	247	248	249	249

GRAYSCALE IMAGE PIXEL VALUE

imgray2 <8x8 uint8>

	1	2	3	4	5	6	7	8
1	112	113	114	115	116	117	118	118
2	112	113	114	115	116	117	118	119
3	113	113	114	115	117	118	119	119
4	114	114	115	116	117	118	119	120
5	114	115	116	117	118	119	120	120
6	115	115	116	117	119	120	121	121
7	115	116	117	118	119	120	121	122
8	116	116	117	118	119	120	121	121

Figure 3.3 Original and Grayscale Image Pixel Value.

3.3 DCT Matrix Multiplication on WINDLX

To continue the multiply step between the grayscale image pixel value matrix and the DCT matrix, the image pixel value must be store first to the memory on WINDLX. The next step is to multiply the grayscale matrix with the DCT matrix on the Figure 3.4 and Figure 3.5.

$$M = \begin{bmatrix} 112 & 113 & 114 & 115 & 116 & 117 & 118 & 118 \\ 112 & 113 & 114 & 115 & 116 & 117 & 118 & 119 \\ 113 & 113 & 114 & 115 & 117 & 118 & 119 & 119 \\ 114 & 114 & 115 & 116 & 117 & 118 & 119 & 120 \\ 114 & 115 & 116 & 117 & 118 & 119 & 120 & 120 \\ 115 & 115 & 116 & 117 & 119 & 120 & 121 & 121 \\ 115 & 116 & 117 & 118 & 119 & 120 & 120 & 121 \\ 116 & 116 & 117 & 118 & 119 & 120 & 120 & 121 \end{bmatrix}$$

Figure 3.4 Matrix M Grayscale Image

$$T = \begin{bmatrix} 0.353 & 0.353 & 0.353 & 0.353 & 0.353 & 0.353 & 0.353 & 0.353 \\ 0.49 & 0.415 & 0.277 & 0.097 & -0.097 & -0.277 & -0.415 & -0.49 \\ 0.461 & 0.191 & -0.191 & -0.461 & -0.461 & -0.191 & 0.191 & 0.461 \\ 0.415 & -0.097 & -0.49 & -0.277 & 0.277 & 0.49 & 0.097 & -0.415 \\ 0.353 & -0.353 & -0.353 & 0.353 & 0.353 & -0.353 & -0.353 & 0.353 \\ 0.277 & -0.49 & 0.097 & 0.415 & -0.415 & -0.097 & 0.49 & -0.277 \\ 0.191 & -0.461 & 0.461 & -0.461 & -0.191 & 0.461 & -0.461 & 0.191 \\ 0.097 & -0.277 & 0.415 & -0.49 & 0.49 & -0.415 & 0.277 & 0.097 \end{bmatrix}$$

Figure 3.5 Matrix T DCT Matrix

For the input value of grayscale image it needs DLX assembly machine instructions shows on Table 3.1 above.

No.	Instructions	Description	Amount
1	SF	Store Float	832
2	Addf	Add Float	448
3	LF	Load Float	1280
4	FP	Floating Point	960
5	Multf	Multiply Float	512

Table 3.6 Instruction List Multiplication Matrix M and Matrix T

After the calculation, the result of this matrix is shown like this:

$$MT = \begin{bmatrix} 302.5 & -88.31 & 66.48 & -56.63 & 35.55 & -4.058 & 20.24 & 31.52 \\ 302.6 & -88.59 & 66.9 & -57.12 & 36.04 & -4.473 & 20.52 & 31.62 \\ 303.8 & -89.54 & 67.46 & -56.46 & 36.14 & -4.109 & 20.55 & 32.24 \\ 305.6 & -88.96 & 67.82 & -57.24 & 36.70 & -4.149 & 21.05 & 32.24 \\ 307.8 & -89.75 & 67.62 & -57.57 & 36.17 & -4.116 & 20.60 & 32.06 \\ 309.1 & -90.98 & 68.59 & -57.40 & 36.76 & -4.167 & 20.90 & 32.78 \\ 310.2 & -90.01 & 67.73 & -57.58 & 36.67 & -4.606 & 21.24 & 32.15 \\ 310.6 & -89.66 & 68.08 & -57.23 & 37.02 & -4.253 & 21.59 & 32.50 \end{bmatrix}$$

Figure 3.7 Matrix MT Result Value

The final step is to multiply matrix MT with transpose DCT matrix to find the final result value of this DCT matrix equation. The result of multiplication between both of the matrices shows on this matrix D below:

$$D = \begin{bmatrix} 108.5 & 98.33 & 138.88 & 114 & 112.5 & 96.84 & 128 & 130.8 \\ 108.5 & 98.23 & 138.9 & 113.9 & 112.5 & 96.79 & 128 & 131.8 \\ 109.2 & 98.52 & 138.7 & 114.2 & 113.2 & 97.85 & 129 & 132 \\ 110.5 & 99.12 & 140.1 & 114.9 & 113.5 & 97.62 & 129.1 & 132.9 \\ 110.4 & 100.1 & 141.2 & 116 & 114.4 & 98.51 & 130.2 & 133 \\ 111.4 & 99.99 & 141.4 & 116 & 115.4 & 99.34 & 131.2 & 134.1 \\ 111.5 & 101 & 142.3 & 116.8 & 115.5 & 99.44 & 130.2 & 134 \\ 112.5 & 101 & 142.3 & 116.9 & 115.5 & 99.45 & 130.2 & 134 \end{bmatrix}$$

Figure 3.8 Matrix D Result Value

From all of the programs created with total of 5920 instructions, here are the list of instructions executed on this experiment shown on Figure 3.9 below. As all of the instructions listed the instruction that often used on this experiment is Load Float which is used to load the value of matrices before being multiplied to other matrices. The numbers of matrices value are mostly a decimal so the instructions used here is load float and not load word considered load word only consist of integers. The total of floating point registers (FPRs) used in this simulation is 1980 which is more than general purpose registers (GPRs).

No.	Instructions	Description	Amount
1	SF	Store Float	1428
2	Addf	Add float	924
3	LF	Load Float	2384
4	FP	Floating Point	1980
5	Multf	Multiply Float	1056
6	Addi	Add Immediate	64
7	SW	Store Word	64

Figure 3.9 List of Instructions of All Programs

4. Conclusion

From all the instruction sets of DLX microprocessor only some of the instructions used in these programs which are Addi, Addf, Lw, Sw, Sf, Lf, and Multf. For inputting the value of image process there are 64 store word instructions and 64 add immediate instructions. The next program is multiplication between image matrix and default DCT matrix there are 832 store float

instructions, 448 add float instructions, 1280 load float instructions, and 512 multiply float instructions. The last program for multiplying matrix result (MT) with DCT Transpose matrix there are 576 store float instructions, 448 add float instructions, 1024 load float instructions and 512 multiply float instructions.

Each of the programs has it owns statistic result for final DCT matrix. From the statistics evaluation can be concluded that the program which has fewer instructions than other has better performance achieved on this DLX microprocessor.

BIBLIOGRAPHY:

[1] S. Chen, H. Xu, D. Liu, B. Hu and H. Wang, " A Vision of IoT: Applications, Challenges, and Opportunities With China Perspective," IEEE Internet of Things Journal, vol. 1, no. 4, ISSN 2327-4662, pp.349-359, Aug 2014.

[2] Dake Liu, "ASIP (Application Specific Instruction-set Processors) design," IEEE International Conference on Changsa, Hunan, China, pp.1-8,2016, 20-23 Oct 2009

[3] Maneesha Gupta, Dr. Amit Kumar Garg, "Analysis of Image Compression Algorithm Using DCT," International Journal of Engineering Research and Applications (IJERA) on, vol. 2, Issue 1, pp.515-521, Jan-Feb 2012.

[4] B. Koc, Z. Arnavut and H. Kocak," Lossless Compression of Dithered Images," IEEE Photonics Journal, vol. 5, no. 3, ISSN 1943-0655, pp. 6800508-6800508, June 2013.

[5] What is Image Compression, https://www.keycdn.com/support/what-is-image-compression, 21 11 2018.

[6] G¨otz Kappen, Lothar Kurz, Tobias G. Noll," Comparison of ASIP and Standard Microprocessor based Navigation Processors," https://pdfs.semanticscholar.org/fdfe/2fa42d9e2435e4ad8e382c942a001502fa4e.pdf,

[7] Sumalatha S., Rajeswari, and Jayalaxmi H.," RISC Architecture based DLX Processor for Fast Convolution and Correlation RISC Architecture based DLX Processor for Fast Convolution and Correlation," https://pdfs.semanticscholar.org/795f/94ad8d8f945e80f12b2a91df7daf6f275bb9.pdf,

[8] " The DLX Instruction Set Architecture," http://www.cs.utexas.edu/pingali/CS378/2015sp/ lectures/ DLX.pdf ,

[9] Dawoud Senouda D., R. Peplow.," Digital System Design-Use of Microcontroller", https://www.riverpublishers.com/pdf/ebook/RP E9788793102293. Pdf.

[10] NItesh Kumar More, Sipi Dubey., ”JPEG Picture Compression Using Discrete Cosine Transform,”RCET, Bhilai, India.

[11] Nyoman Karna, Nimas Fatihah, Dong-Seong Kim.,” Evaluation of DLX Microprocessor Instructions Efficiency For Image Compression,” Telkom University, Indonesia., Gumi, South Korea.,