Implementasi Transformasi Wavelet Diskret Haar pada FPGA Xilinx Spartan-3E

Sasmito Aji[1]

***Abstract*— This paper discusses the efficient and accurate implementation of the discrete Haar Wavelet Transform (HWT) by selection of the best fit filter bank structure and data format. There are three filter bank structures including one polyphase and two lattice filter bank structure and also three data formats including two fixed point and one single precision floating point data format which are used in implementation of discrete Haar wavelet transform are discussed in relation with the using of FPGA resources. Beside that this paper also discuss about the effect of decomposition level on utilization of FPGA resources. The decomposition level is varied from 1st to 6th level and then its effect on the FPGA resources will be observed. The implementation of the Haar wavelet transform is performed on Field Programmable Gate Array (FPGA) Xilinx Spartan-3E and it is intended for processing of voice signals. By using the lattice filter bank structure and fixed-point data format, the implementation of Haar Wavelet Transform with six decomposition levels only require 5% of FPGA's slice and give accuracy 98.9%.**

***Intisari*— Penelitian ini membahas implementasi transformasi wavelet diskret Haar yang efisien dan akurat dengan melakukan pemilihan struktur *filter bank* dan format data yang tepat. Ada tiga struktur *filter bank,* yang meliputi satu struktur *filter bank* polifase dan dua struktur *filter bank lattice* serta tiga format data, yang meliputi dua format data *fixed point* dan satu format data *single precision floating point,* yang digunakan dalam implementasi transformasi wavelet diskret Haar yang diteliti pengaruhnya terhadap penggunaan *resources*. Selain itu pada penelitian ini juga dibahas mengenai pengaruh tingkat dekomposisi terhadap penggunaan *resources* FPGA. Tingkat dekomposisi divariasikan dari satu tingkat dekomposisi sampai dengan enam tingkat dekomposisi untuk kemudian diteliti pengaruhnya terhadap *resources* FPGA. Implementasi transformasi wavelet diskret Haar dilakukan pada Field Programmable Gate Array (FPGA) Xilinx Spartan-3E dan ditujukan untuk pemrosesan isyarat suara. Dengan menggunakan struktur *filter bank lattice* dan format data *fixed point*, implementasi transformasi wavelet diskret Haar dengan enam tingkat dekomposisi hanya memerlukan *slice* sebanyak 5% dan memiliki akurasi hasil sebesar 98,9%**

***Kata Kunci*— Transformasi Wavelet Diskret Haar, FPGA, Verilog, *Filter bank*, Format Data.**

I. PENDAHULUAN

Saat ini ada banyak penelitian yang mengulas transformasi wavelet baik itu yang mengulas mengenai kegunaannya maupun yang mengulas mengenai proses implementasi transformasi wavelet baik itu pada perangkat lunak maupun perangkat keras. Salah satu media perangkat keras yang digunakan untuk implementasi transformasi wavelet adalah FPGA. Ada banyak penelitian yang membahas mengenai transformasi wavelet pada FPGA, baik itu penelitian yang memanfaatkan transformasi wavelet untuk pengolahan isyarat maupun penelitian untuk yang berfokus kepada cara meningkatkan kecepatan dan efisiensi penggunaan *resources* untuk implementasi transformasi wavelet di FPGA dan kedua macam penelitian adalah saling terkait [1, 2, 3, 4, 5, 6].

Penelitian yang berfokus kepada perangkat keras, terutama FPGA tidak bisa lepas dari optimasi dan efisiensi penggunaan *resources*. Hal ini dikarenakan terbatasnya sumber daya atau *resources* yang tersedia di dalam FPGA sehingga harus dicari teknik dan metode implementasi yang optimal dan efisien dalam penggunaan *resources* serta mampu memberikan hasil komputasi yang benar. Hampir semua penelitian yang menggunakan FPGA selalu memiliki bahasan tentang kecepatan komputasi, efisiensi *resources* yang digunakan serta tentunya kebenaran hasil penelitian.

Ada banyak penelitian yang membahas mengenai implementasi transformasi wavelet pada FPGA dan optimasinya, yang salah satunya yang digunakan untuk untuk pengolahan citra dan video baik itu untuk kompresi maupun denoising [2, 4, 6, 7, 8, 9, 10]. Selain digunakan untuk pengolahan citra, implementasi transformasi wavelet pada FPGA juga digunakan untuk pengolahan sinyal suara [11]. Walaupun banyak penelitian yang mengimplementasikan transformasi wavelet pada FPGA, tetapi masih sedikit yang meneliti unjuk kerja mengenai struktur *filter bank* dan format data dalam transformasi wavelet itu sendiri.

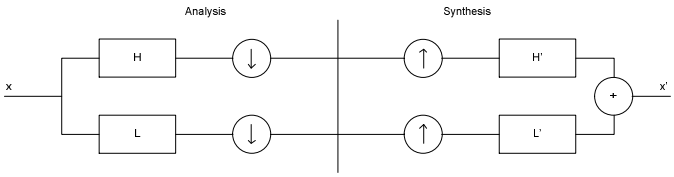
Penelitian tentang struktur *filter bank* dan format data sangat penting untuk menghasilkan implementasi transformasi wavelet yang efisien dan akurat. *Paper* ini mendiskusikan hasil implementasi transformasi wavelet yang meneliti strukur *filter bank* dan format data untuk menghasilkan transformasi wavelet yang efisien dan akurat. Ada tiga struktur *filter bank* dan tiga format data yang diteliti. Struktur filter bank yang diteliti meliputi struktur polifase dan *lattice*, dengan struktur *filter bank lattice* dibedakan kembali menjadi *lattice*1 dan *lattice*2, sedangkan untuk tipe format data terdiri dari satu *floating point* dan dua *fixed point*. Selain diteliti tentang struktur *filter bank* dan format data pada penelitian ini juga diteliti tentang pengaruh tingkat dekomposisi terhadap penggunaan *resources* FPGA. Penelitian dilakukan pada FPGA Xilinx Spartan 3-E (XC3S500E-4FG320) dengan program yang ditulis menggunakan Verilog HDL.

II. TRANSORMASI WAVELET DIKSRET HAAR

Dalam transformasi wavelet diskret, isyarat input x[k] dimasukan ke dalam tapis *low pass* (L) dan tapis *high pass* (H). Tapis ini akan memisahkan isyarat menjadi frekuensi rendah dan tinggi dengan panjang dan jumlah data yang sama. Untuk menghilangkan adanya *redundancy* data, setelah tahap pentapisan dilakukan *downsampling*. *Downsampling* dilakukan dengan membuang separuh data dan menyisakan separuh data yang lain dengan data yang dibuang adalah data

[1]*Instrument engineer, BP Tangguh LNG, Tangguh LNG Site Teluk Bintuni Papua barat, email:sasmito.adji@se1.bp.com*

dengan dengan indek ganjil dan data yang disisakan adalah data dengan indek genap. Untuk melakukan proses rekonstruksi, isyarat akan di-*upsampling* terlebih dahulu. *Upsampling* dilakukan dengan menyisipkan data nol di antara dua data masukan yang berurutan. Isyarat yang telah di-*upsampling* kemudian dilewatkan ke tapis *low pass* (L') dan tapis *high pass* (H'). Tapis *low pass* (L') dan *high pass* (H') harus bersesuaian dengan tapis *low pass* (L) dan *high pass* (H). Proses analisis dan rekonstruksi dalam transformasi wavelet diskret diilustrasikan dengan Gbr. 1. Tapis *low pass* (L) dan *high pass* (H) untuk tahap analisis dan tapis *low pass* (L') dan *high pass* (H') untuk tahap rekonstruksi transformasi wavelet diskret Haar ditunjukkan dengan persamaan (1)[12]. Dalam transformasi wavelet diskret Haar, tapis yang digunakan terdiri dari nilai $1/\sqrt{2}$ dan $-1/\sqrt{2}$, dengan tapis *low pass* selalu positif dan tapis *high pass* merupakan kombinasi positif dan negatif.



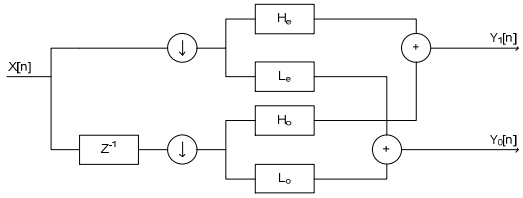
Gbr. 1. Analisis and Rekonstrruksi Transformasi Wavelet Diskret [13]

$$L=\left[\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\right] \qquad H=\left[\frac{1}{\sqrt{2}} \quad -\frac{1}{\sqrt{2}}\right]$$
$$L'=\left[\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\right] \qquad H'=\left[-\frac{1}{\sqrt{2}} \quad \frac{1}{\sqrt{2}}\right] \qquad (1)$$

A. Struktur Filter bank

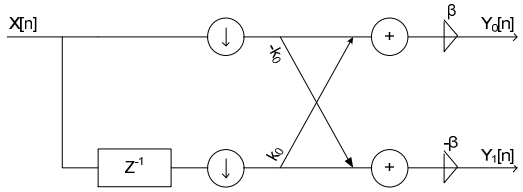
Untuk melakukan transformasi wavelet diskret Haar, tapis *low pass* dan *high pass* harus disusun membentuk *filter bank*. Penyusunan tapis sangat berpengaruh terhadap komputasi yang dilakukan sehingga pemilihan struktur *filter bank* yang tepat dapat menghasilkan komputasi yang optimal. Ada beberapa macam struktur *filter bank* yang dikenal yaitu *direct form*, polifase, *lattice*, dan *lifting*. Dalam penelitian ini akan dibahas dua struktur *filter bank* yaitu polifase dan *lattice* dengan struktur *filter bank lattice* dibedakan kembali menjadi *lattice*1 dan *lattice*2.

Struktur *filter bank lattice*1 dan *lattice*2 memiliki prinsip yang sama, yang mebedakan adalah algoritma komputasinya. Jika pada struktur *filter bank lattice*1, proses normalisasi (perkalian dengan β) dilakukan pada setiap tahap dekomposisi maka pada struktur *filter bank lattice*2 proses normalisasi dilakukan hanya sekali pada akhir tahap dekomposisi tertinggi. Struktur *filter bank* polifase dapat diilustrasikan dengan Gbr. 2 sedangkan struktur *filter bank lattice* diilustrasikan pada Gbr. 3. Kedua tipe struktur *filter bank* dapat dinyatakan dengan persamaan (2) dan (3).



Gbr. 2. Struktur *Filter bank* Polifase

$$\begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix} = \begin{bmatrix} L_e & L_o \\ H_e & H_o \end{bmatrix} \begin{bmatrix} X_e \\ z^{-1}X_o \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} X_e \\ z^{-1}X_o \end{bmatrix} \quad (2)$$



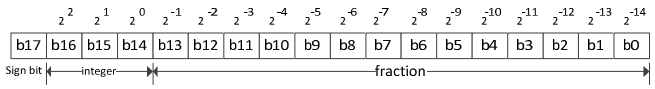
Gbr. 3. Struktur *Filter bank Lattice*

$$\begin{bmatrix} Y_0 \\ Y_1 \end{bmatrix} = \begin{bmatrix} \beta & k_0\beta \\ k_0\beta & -\beta \end{bmatrix} \begin{bmatrix} X_e \\ z^{-1}X_o \end{bmatrix} \text{ with } k_0 = 1 \;\&\; \beta = \frac{1}{\sqrt{2}} \quad (3)$$

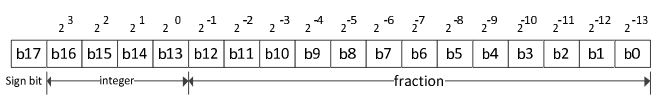
B. Format Data

Format data yang digunakan sangat berpengaruh terhadap akurasi dari hasil transformasi wavelet. Tetapi antara format data dan akurasi harus diperhitungkan agar bisa menghasilkan program yang efisien namun tetap akurat. Ada tiga format data yang diteliti yang terdiri dari dua format data *fixed point* dan satu format data *floating point* yang ditunjukkan pada Gbr. 4 s.d. 6. Dalam format data *fixed point* ada 18-bit data yang terdiri dari *sign* bit, *integer* dan *fraction*. Untuk format data *fixed point*1 terdiri dari 3-bit *integer* dan 14-bit *fraction*, sedangkan untuk format data *fixed point*2 terdiri dari 4-bit *integer* dan 13-bit *fraction*. Sedangkan dalam *floating point* data akan terdiri dari 1-bit *sign* (S), 8-bit *exponent* (Exp) dan 23-bit *mantissa* (M) dan dinyatakan dengan persamaan (4) [14].

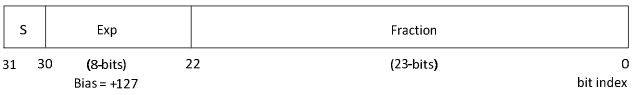
$$N = (-1)^S 2^{(Exp-127)} M \quad (4)$$



Gbr. 4. Data Format *Fixed Point*1



Gbr. 5. Data Format *Fixed Point*2



Gbr. 6. Data Format *Single precision Floating Point* [14]

C. Field Programmable Gate Array (FPGA)

FPGA dapat diartikan sebagai dengan peralatan logika yang terdiri dari susunan gerbang-gerbang logika dua dimensi dan saklar-saklar yang dapat diprogram. Gerbang-gerbang logika di dalam FPGA disusun dengan menggunakan konsep CLB (*Complex Logix Block*). Jumlah gerbang logika yang ada di dalam FPGA dari ribuan sampai dengan jutaan tergantung dari jenis FPGA. Semakin banyak jumlah gerbang logika yang ada di dalam FPGA maka fungsi yang dapat diimplementasikan di FPGA tersebut menjadi lebih banyak dan dapat lebih kompleks.

Selain terdiri dari berbagi macam gerbang logika dan saklar, di dalam FPGA juga terdapat flip-flop, operator numerik, dan berbagai macam fungsi atau IP (*Intelectual Property*). Flip-flop merupakan komponen dasar yang berguna untuk proses penyimpanan baik itu untuk penyimpan data sementara, *array*, ataupun sebagai memori. Operator numerik yang ada di dalam FPGA dapat berupa operator dasar seperti penjumlahan dan pengurangan dan operator yang lebih kompleks seperti pembagian dan perkalian. Sedangkan untuk fungsi-fungsi (IP) yang ada di dalam FPGA tergantung dari jenis FPGA. FPGA dengan kemampuan yang besar dapat memiliki fungsi-fungsi yang lebih kompleks dibandingkan dengan FPGA dengan kemampuan yang lebih kecil.

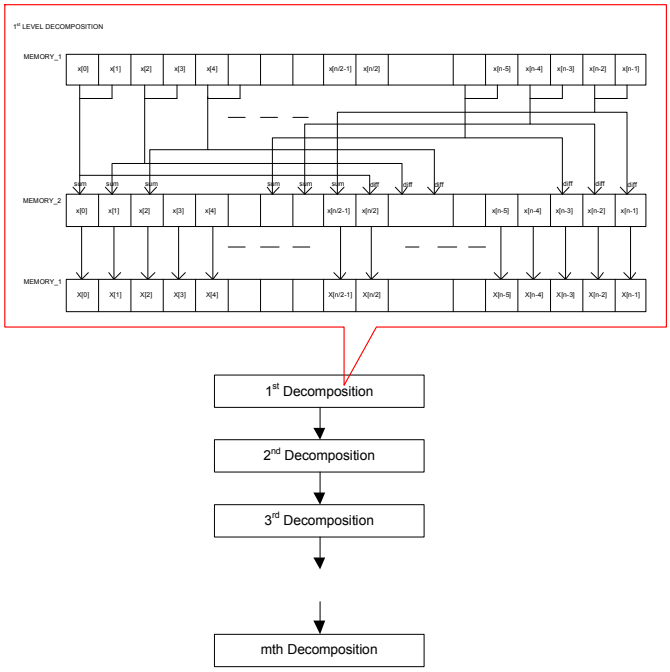
III. METODOLOGI

Penelitian yang telah dilakukan terdiri atas implementasi transformasi wavelet diskret Haar di perangkat lunak, dan di FPGA. Untuk dapat mengimplementasikan transformasi wavelet Haar di FPGA ada beberapa hal yang harus dilakukan yaitu perancangan sistem, pengembangan sistem, simulasi dan pengujian serta analisis. Implementasi di perangkat lunak hanya menggunakan fungsi-fungsi yang sederhana agar menyerupai implementasi di FPGA. Fungsi untuk transformasi wavelet Haar yang sudah ditulis di perangkat lunak kemudian diuji dan hasilnya dibandingkan dengan hasil transformasi wavelet diskret Haar yang menggunakan Matlab *toolbox*. Hasil implementasi transformasi wavelet di perangkat lunak akan dijadikan sebagai data referensi.

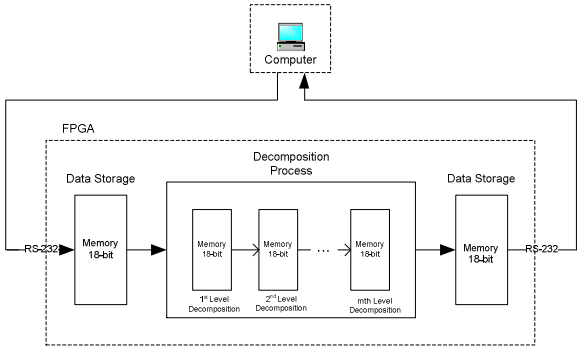
Ketika implementasi transformasi wavelet diskret Haar di perangkat lunak telah selesai maka tahap selanjutnya adalah implementasi transformasi wavelet Haar di FPGA Tahapan implementasi transformasi wavelet di FPGA meliputi:

A. Perancangan Sistem

Transformasi wavelet diskret Haar terdiri dari beberapa tahap dekomposisi yang pada masing-masing tahap dekomposisi terdiri dari operasi penjumlahan atau rata-rata dan pengurangan atau selisih. Secara sederhana tahapan dekomposisi dalam transformasi wavelet ini dapat diilustrasikan dengan Gbr. 7. Untuk dapat mengimplementasikan tahapan dekomposisi pada FPGA maka diperlukan memori. Memori digunakan untuk menyimpan data masukan, data hasil dekomposisi, serta data keluaran. Kapasitas memori disesuaikan dengan data yang akan diproses dan ditransformasikan.



Gbr. 7. Proses Dekomposisi dalam Transformasi Wavelet Haar



Gbr 8. Perancangan Sistem untuk Transformasi Wavelet Haar

Data masukan akan dilewatkan dari komputer melalui komunikasi serial RS-232 ke dalam FPGA, dan untuk kemudian data ini akan ditransformasikan. Data keluaran dari FPGA akan dilewatkan kembali melalui komunikasi serial RS-232 untuk selanjutnya ditampilkan di dalam komputer. Perancangan sistem untuk transformasi wavelet secara sederhana dapat diilustrasikan dengan Gbr. 8. Data yang dimasukan dan dikeluarkan FPGA dalam format biner sehingga diperlukan bantuan perangkat lunak untuk mengubah data biner ke desimal dan sebaliknya. Dalam penelitian ini data masukan yang diproses berasal dari isyarat suara hasil pengucapan huruf ”I” yang disimpan dalam format wav.

Isyarat suara akan dipotong menjadi satu periode dengan setiap periode terdiri dari 64 titik data. Satu periode isyarat suara merupakan satu buah data untuk transformasi wavelet.

Untuk menghasilkan transformasi wavelet yang optimal dan efisien akan dipilih struktur *filter bank*, format data dan tingkat dekomposisi yang tepat. Beberapa program transformasi wavelet dengan strukur *filter bank*, format data, dan tingkat dekomposisi yang berbeda diimplementasikan di FPGA. Penggunaan *resources* FPGA yang meliputi *slice* flip-flop, *Look Up Table* (LUT), *slice*, *Input Output Block* (IOB), *Random Access Memory* (RAM) dan *multipier* diteliti untuk berbagi macam implementasi. Implementasi dengan penggunaan *resources* yang paling minimal adalah yang paling efisien. Selain diteliti efisiensi, pada penelitian ini juga diteliti tentang akurasi hasil implementasi transformasi wavelet. Akurasi hasil implementasi transformasi wavelet diperoleh dengan membandingkan hasil pengujian transformasi wavelet dengan data data referensi. Pengujian dilakukan dengan cara memasukan data uji ke FPGA. Hasil transformasi wavelet di FPGA akan dikirim ke komputer untuk kemudian dapat dibandingkan dengan data referensi dan diperoleh tingkat akurasi.

B. Pengembangan Sistem

Program transformasi wavelet diskret Haar ditulis dengan menggunakan Verilog HDL di Xilinx ISE 13.2 yang berfungsi sebagai perangkat lunak untuk pengembangan FPGA Xilinx Spartan-3E. Program kemudian disimulasikan dengan menggunakan perangkat lunak ISIM 13.2 untuk memastikan program berjalan dengan benar dan sesuai dengan desain.

C. Pengujian di FPGA dan Analisis

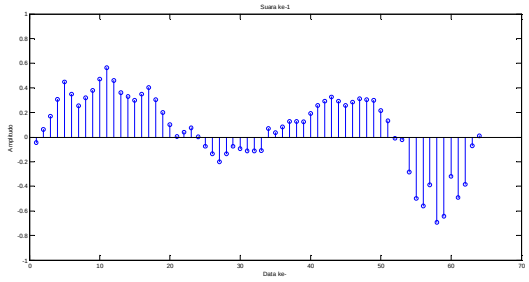
Setelah dilakukan simulasi, perlu dilakukan pemetaan program baik IO maupun logika ke dalam FPGA yang akan digunakan (XC3S500E FG320). Pemetaan dilakukan dengan menggunakan perangkat lunak PlanAhead 13.2. Ketika pemetaan selesai maka program dapat ditulis ke FPGA dengan bantuan perangkat lunak IMPACT dan transformasi wavelet di FPGA dapat mulai di uji. Data uji akan dikirim dari komputer ke FPGA melalui komunikasi serial RS-232. Data hasil pengujian akan dikirim balik dari FPGA ke komputer dengan media komunikasi yang sama. Data hasil pengujian akan dibandingkan kembali dengan data referensi untuk kemudian dianalisis.

IV. HASIL DAN PEMBAHASAN

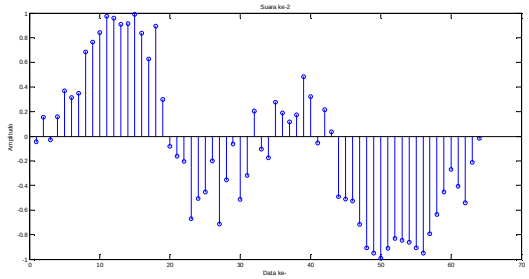
Implementasi transformasi wavelet diskret Haar yang telah dilakukan dibagi dalam dua bagian utama yaitu: implementasi di perangkat lunak, dan implementasi di FPGA. Implementasi di perangkat lunak akan menghasilkan data referensi sebagai pembanding dalam implementasi di FPGA.

A. Implementasi di Perangkat Lunak

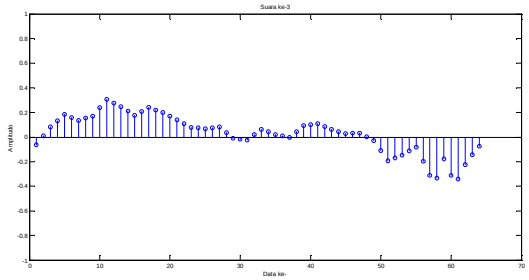
Implementasi transformasi wavelet diskret Haar di perangkat lunak dilakukan dengan membuat program dan melakukan pengujian terhadap program tersebut. Hasil pengujian dibandingkan dengan hasil transformasi wavelet diskret Haar yang menggunakan Matlab *toolbox*. Hasil pengujian menunjukan implementasi transformasi wavelet Haar di perangkat lunak memiliki akurasi 100% jika dibandingkan dengan hasil transformasi wavelet diskret Haar Matlab *toolbox*. Data hasil pengujian ini dapat dijadikan referensi untuk implementasi di FPGA. Data uji untuk transformasi wavelet diskret Haar ditunjukkan di Gbr. 9 s.d. Gbr. 12. Sedangkan hasil transformasi wavelet diskret Haar dengan tingkat dekomposisi ke-6 yang dilakukan dengan perangkat lunak ditunjukkan pada Gbr. 13 s.d. 16.



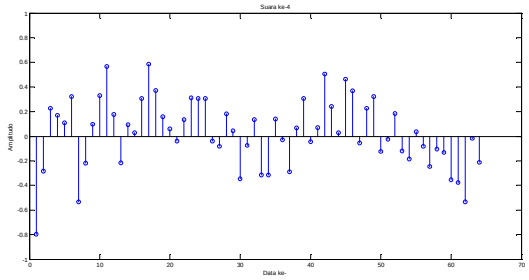
Gbr. 9. Data Uji ke-1



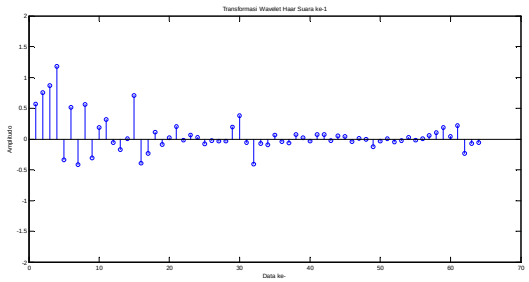
Gbr. 10. Data Uji ke-2



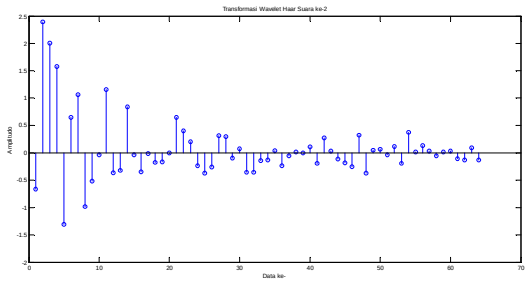
Gbr. 11. Data Uji ke-3



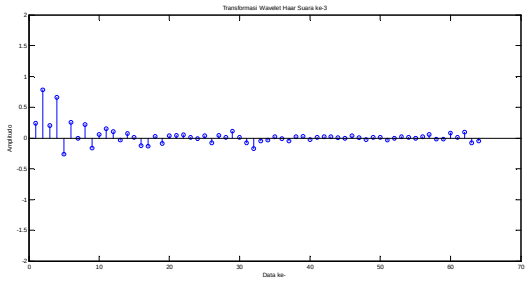
Gbr. 12. Data Uji ke-4



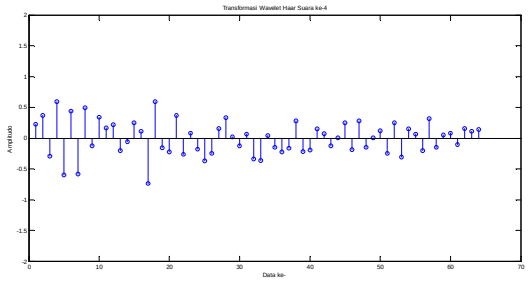
Gbr. 13. Hasil Transformasi Wavelet di Perangkat Lunak Data Uji ke-1



Gbr. 14. Hasil Transformasi Wavelet di Perangkat Lunak Data Uji ke-2



Gbr. 15. Hasil Transformasi Wavelet di Perangkat Lunak Data Uji ke-3



Gbr. 16. Hasil Transformasi Wavelet di Perangkat Lunak Data Uji ke-4

B. Implementasi di FPGA

Untuk memperoleh implementasi transformasi wavelet diskret Haar di FPGA yang efisien maka perlu dilakukan pemilihan struktur *filter bank*, format data, dan tingkat dekomposisi yang tepat. Implementasi di FPGA dilakukan tehadap tiga bagian besar dalam proses transformasi wavelet Haar yaitu bagian pengolahan data masukan, dekomposisi, dan bagian pengolahan data keluaran.

Bagian pengolahan data masukan dan data keluaran berkaitan dengan penerimaan data, pengiriman data, dan komunikasi antara FPGA dan komputer. Sedangkan dekomposisi merupakan bagian utama dalam proses transformasi wavelet. Di dalam bagian dekomposisi inilah akan diimplementasikan struktur *filter bank*, format data, dan tingkat dekomposisi.

1) Pemilihan Struktur Filter bank: Ada tiga struktur *filter bank* yang diteliti yaitu polifase dan *lattice* dengan struktur *filter bank lattice* akan dibedakan menjadi *lattice*1 dan *lattice*2. *Resources* yang diperlukan untuk mengimplementasikan transformasi wavelet diskret haar dengan struktur *filter bank* polifase ditunjukkan pada tabel I dan untuk struktur *filter bank lattice*1 ditunjukkan pada tabel II sedangkan struktur *filter bank lattice*2 ditunjukkan pada tabel III.

Dari tabel I, tabel II dan tabel III dapat diperoleh bahwa struktur filterbank *lattice*2 memerlukan jumlah *resources* yang lebih sedikit dan lebih efisisen dibandingkan dengan struktur *filter bank* polifase dan *lattice*1. Jika kedua macam implementasi transformasi wavelet diuji dan hasilnya dibandingkan dengan data referensi maka diperoleh akurasi 99.67 % (error = 0.33%) untuk struktur *filter bank* polifase dan akurasi 99.65% (error = 0.35%) untuk strukur *filter bank lattice*. Jadi implementasi dengan *filter bank* polifase dan *lattice* memiliki akurasi yang sangat baik.

TABEL I. *RESOURCES* UNTUK *FILTER BANK* POLIFASE

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	339 (3%)	653 (5%)
Jumlah LUT (max. 9312)	471 (5%)	1553 (16%)
Jumlah *Slice* (max. 4656)	287 (6%)	862 (18%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	3 (15%)	3 (15%)
Jumlah *Multiplier* 18x18 (max. 20)	4 (20%)	4 (20%)

a. *Resources* yang diperlukan untuk program dekomposisi
b. *Resourcse* yang diperlukan untuk program keseluruhan yang meliputi pengolahan data masukan, dekomposisi dan pengolahan data keluaran

TABEL II. *RESOURCES* UNTUK *FILTER BANK LATTICE*1

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	210 (2%)	528 (5%)
Jumlah LUT (max. 9312)	350 (3%)	1437 (15%)
Jumlah *Slice* (max. 4656)	205 (4%)	783 (18%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	3 (15%)	3 (15%)
Jumlah *Multiplier* 18x18 (max. 20)	2 (10%)	2 (10%)

TABEL III. *RESOURCES* UNTUK *FILTER BANK LATTICE*2

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	160 (1%)	478 (5%)
Jumlah LUT (max. 9312)	346 (3%)	1417 (15%)
Jumlah *Slice* (max. 4656)	190 (4%)	763 (16%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

2) Pemilihan Format Data: Ada tiga format data yang diteliti yaitu *fixed point*1, *fixed point*2, dan *single precision floating point*. Ketiga macam format data diimplementasikan pada transformasi wavelet diskret Haar yang menggunakan struktur *filter bank lattice*2.

Struktur *filter bank lattice*2 terbukti efisien berdasarkan IVB.1. *Resources* yang diperlukan untuk ketiga format data ditunjukkan pada tabel IV s.d. tabel VI.

TABEL IV. *RESOURCES* DENGAN FORMAT DATA *FIXED POINT* 1

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	160 (1%)	478 (5%)
Jumlah LUT (max. 9312)	346 (3%)	1417 (15%)
Jumlah *Slice* (max. 4656)	190 (4%)	763 (16%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

TABEL V. *RESOURCES* DENGAN FORMAT DATA *FIXED POINT* 2

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	160 (1%)	478 (5%)
Jumlah LUT (max. 9312)	346 (3%)	1417 (15%)
Jumlah *Slice* (max. 4656)	190 (4%)	763 (16%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

TABEL VI. *RESOURCES* FOR HWT WITH DATA FORMAT *FLOATING POINT*

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	1789 (19%)	2187 (23%)
Jumlah LUT (max. 9312)	1930 (20%)	3411 (36%)
Jumlah *Slice* (max. 4656)	1422 (30%)	2221 (47%)
Jumlah IOB (max. 232)	70 (30%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	4 (20%)	4 (20%)

Berdasarkan tabel III s.d tabel VI, untuk kedua format data *fixed point* jumlah *resources* yang diperlukan hampir sama walaupun format data tersebut memiliki jumlah *integer* dan *fraction* yang berbeda. Jadi asalkan memiliki ukuran data yang sama, format data *fixed point* akan memerlukan jumlah *resources* yang hampir sama.

Lain halnya dengan format data *floating point*, format data ini memerlukan *resources* yang jauh lebih banyak. Misalnya untuk *slice* yang jumlahnya 600% lebih banyak dibandingkan dengan *slice* untuk format data *fixed point*. Tetapi walaupun memerlukan jumlah *resources* yang banyak, tetapi format data *floating point* menawarkan akurasi yang baik serta rentang data yang sangat lebar. Suatu kelebihan yang tidak dimiliki oleh format data *fixed point*.

Dalam format data *fixed point* akurasi dan range merupakan sesuatu yang berlawanan. Akurasi semakin baik jika jumlah bit *fraction* semakin banyak, tetapi hal ini akan membuat range data menjadi lebih sempit. Oleh karena itu dalam format data *fixed point* harus dipertimbangkan range data dari sinyal yang akan diolah serta akurasi yang diharapkan. Untuk kemudian baru bisa ditentukan format data *fixed point* yang tepat.

Mengingat besarnya jumlah *resources* yang diperlukan untuk implementasi transformasi wavelet dengan format data *floating point*, maka format data *fixed point* menjadi pilihan yang tepat. Selain karena efisien dalam penggunaan *resources*, format data *fixed point* juga bisa memberikan tingkat akurasi yang cukup baik.

Hal ini terbukti ketika digunakan untuk implementasi transformasi wavelet dengan struktur *filter bank* polifase atau *lattice* pada bagian IVB.1, akurasi yang diperoleh bisa mencapai 99.67%.

3) Pengaruh Tingkat Dekomposisi: Dengan menggunakan struktur *filter bank lattice* dan format data *fixed point* diimplementasikan transformasi wavelet dengan beberapa macam tingkat dekomposisi mulai dari dekomposisi tingkat 2 sampai dengan dekomposisi tingkat 6. Dan jumlah *resources* yang diperlukan untuk implementasi ada pada tabel VI s.d. tabel IX. Khusus untuk dekomposisi tingkat 1 jumlah *resources* nya sudah ditabelkan pada tabel V.

TABEL VII. *RESOURCES* UNTUK DEKOMPOSISI TINGKAT 2

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	168 (1%)	485 (5%)
Jumlah LUT (max. 9312)	385 (4%)	1462 (15%)
Jumlah *Slice* (max. 4656)	208 (4%)	789 (16%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

TABEL VIII. *RESOURCES* UNTUK DEKOMPOSISI TINGKAT 3

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	176 (1%)	492 (5%)
Jumlah LUT (max. 9312)	434 (4%)	1506 (16%)
Jumlah *Slice* (max. 4656)	236 (5%)	811 (17%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

TABEL IX. *RESOURCES* UNTUK DEKOMPOSISI TINGKAT 4

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	186 (1%)	505 (5%)
Jumlah LUT (max. 9312)	428 (4%)	1502 (16%)
Jumlah *Slice* (max. 4656)	230 (4%)	807 (17%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

TABEL X. *RESOURCES* UNTUK DEKOMPOSISI TINGKAT 5

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	189 (2%)	506 (5%)
Jumlah LUT (max. 9312)	467 (5%)	1529 (16%)
Jumlah *Slice* (max. 4656)	250 (5%)	820 (17%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

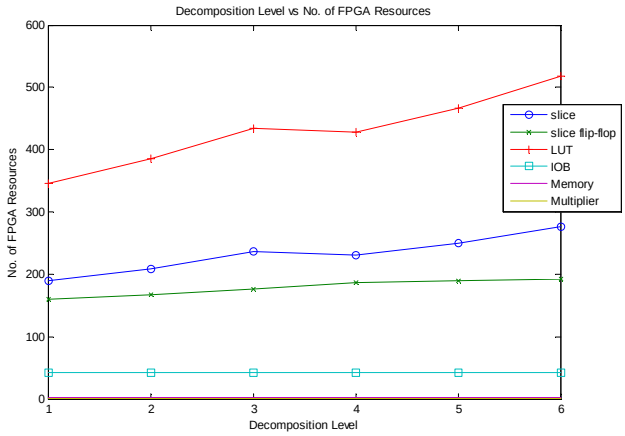
TABEL XI. *RESOURCES* UNTUK DEKOMPOSISI TINGKAT 6

Logic Utilization	Dekomposisi [a]	Total Program [b]
Jumlah *Slice* Flip-Flop (max. 9312)	193 (2%)	511 (5%)
Jumlah LUT (max. 9312)	518 (5%)	1601 (17%)
Jumlah *Slice* (max. 4656)	276 (5%)	858 (18%)
Jumlah IOB (max. 232)	42 (18%)	13 (5%)
Jumlah RAM (max. 20)	2 (10%)	2 (10%)
Jumlah *Multiplier* 18x18 (max. 20)	1 (5%)	1 (5%)

Jika penggunaan *resources* tersebut digambarkan dalam bentuk grafik maka akan diperoleh Gbr. 17. Dari gambar

tersebut terlihat jelas bahwa kenaikan tingkat dekomposisi tidak terlalu berpengaruh terhadap kenaikan jumlah *resources*. Pada dekomposisi tingkat 6 jumlah *resources* yang diperlukan hanya 1% lebih banyak jika dibandingkan dengan dekomposisi tingkat 1. Ketika data uji ditransformasikan dengan dekomposisi tingkat 6 maka diperoleh tingkat akurasi 98.13% untuk format data *fixed point*1 dan 98.90% untuk format data *fixed point*2.

Dengan menggunakan struktur *filter bank* dan format data yang tepat implementasi transformasi wavelet diskret Haar hanya memerlukan jumlah resource yang relatif kecil. Dengan menggunakan struktur *filter bank lattice* dan format data *fixed point* hanya diperlukan *slice* sebesar 1% untuk dekomposisi tingkat 1 dan 2% untuk dekomposisi tingkat 6. Struktur *filter bank lattice* mampu mempersingkat jumlah komputasi serta mampu mengurangi penggunaan memori dan pengali secara *sign*ifikan. Hanya diperlukan dua buah memori dan satu buah pengali untuk mengimplementasikan transformasi wavelet dengan struktur *filter bank lattice*2.



Gbr. 17. Tingkat Dekomposisi dan *Resources* FPGA

Penggunaan format data sangat berpengaruh terhadap penggunaan *resources* FPGA. Format data *floating point* menawarkan akurasi yang sangat baik serta range data yang sangat lebar, tetapi penggunaan data *single precision floating point* memerlukan jumlah *resources* yang jauh lebih banyak misalnya untuk *slice* yang jumlahnya 600% lebih banyak dibandingkan *slice* yang diperlukan oleh format data *fixed point*. Format data *fixed point* memerlukan *resources* FPGA yang efisien dan bisa menghasilkan akurasi yang baik asalkan formatnya (jumlah *integer* dan *fraction*) disesuaikan dengan sinyal yang akan diolah. Selain itu dalam format data *fixed point* asalkan jumlah bit nya sama maka diperlukan *resources* yang sama walaupun jumlah *integer* dan *fraction* yang berbeda.

Peningkatan jumlah dekomposisi tidak meningkatkan jumlah *resources* FPGA secara *sign*ifikan hal ini dikarenakan jumlah data yang didekomposisikan menurun menjadi setengah jumlah semula seiring dengan tingkat dekomposisi yang meningkat. Pada dekomposisi tingkat 6, jumlah *resources* yang diperlukan hanya 1% lebih banyak dibandingkan dengan jumlah *resources* pada dekomposisi tingkat 1. Tetapi pada dekomposisi tingat 6 akurasi yang diperoleh menurun dari 99.67% ke 98.13%. Penurunan akurasi disebabkan perambatan error akibat adanya rounding error yang terjadi dari dekomposisi tingkat 1 sampai dengan dekomposisi tingkat 6. Perambatan error akibat pembulatan bisa dihindari jika format data yang digunakan adalah *single precision floating point.*

V. KESIMPULAN

Pada penelitian ini telah diimplementasikan dan dioptimasi transformasi wavelet Haar diskret pada Xilinx Spartan-3E FPGA (XC3S500E FG320). Optimasi yang dilakukan terkait dengan struktur *filter bank* dan format data serta pengaruh tingkat dekomposisi terhadap penggunaan *resources* FPGA.

Berdasarkan bagian IV dapat diperoleh bahwa struktur *filter bank lattice*2 merupakan struktur *filter bank* yang paling efisien dalam penggunaan *resources*. Selain itu format data *fixed point*2 adalah format data yang paling tepat sebab bisa memberikan efisiensi *resources* yang tinggi dan akurasi yang baik. Selain itu peningkatan tingkat dekomposisi tidak meningkatkan penggunaan *resources* secara *sign*ifikan. Jika dibandingkan, transformasi wavelet dengan dekomposisi tingkat ke-6 hanya memerlukan *resources* 1% lebih tinggi dari pada transformasi wavelet dengan dekomposisi tingkat ke-1.

Transformasi wavelet yang diimplementasikan dengan struktur *filterbank lattice*2, format data *fixed point*1 dan dengan tingkat dekomposisi tetap tingkat ke-6 mampu memberikan akurasi yang sangat baik yaitu 98,14% atau tingkat kesalahan hanya 1,86% sedangkan yang diimplementasikan dengan struktur *filterbank lattice*2, format data *fixed point*2 dan dengan tingkat dekomposisi tetap tingkat ke-6 mampu memberikan akurasi yang sangat baik yaitu 98,90% atau tingkat kesalahan hanya 1,10%. Kedua macam implementasi cukup efisien dalam pernggunaan *resources* sebab hanya memerlukan 5% dari *slice* yang tersedia di FPGA.

Saran

Penelitian selanjutnya dapat dilakukan dengan menambah ketelitian data yaitu dengan menggunakan format data dengan range yang lebih lebar. Selain itu juga dapat dilakukan proses optimasi penggunaan resources FPGA lebih lanjut dengan melakukan proses *mapping* FPGA yang lebih baik.

REFERENSI

[1] M. Angelopoulou, K. Masselos, P. Cheung and Y. Andreopoulos, "A Comparison of 2-D Discrete Wavelet Transform Computation Schedules on FPGA," 2006.

[2] Q. Huang, Y. Wang and S. Chang, "High Performance FPGA Implementation of Discrete Wavelet Transform for Image Processing," 2011.

[3] M. Guarisco, X. Zhang, H. Rabah and W. Serge, "An Efficient Implementation of Scalable Architecture for Discrete Wavelet Transform On FPGA," in *IEEE*, 2007.

[4] M. Katona, A. Pizurica, N. Teslic, V. Kovacevic and W. Philips, "FPGA Des*ign* and Implementation of a Wavelet Domain Video Denoising System".

[5] P. Gupta and S. K. Lenka, "A Novel VLSI of Architecture of High Speed 1D Discrete Wavelet Transform," *International Journal of Electrical and Electronic Engineering (IJEEE)*, vol. 3, no. I, pp. 79-85,

2013.

[6] H. Sahoolizadeh and A. Keshavarz, "A FPGA Implementation of Neural / Wavelet Face Detection System," *Australian Journal of Basic and Applied Science*, pp. 379-388, 2010.

[7] A. Benkrid, K. Benkrid and D. Crookes, "Des*ign* and Implementation of a Generic 2-D Orthogonal Discrete Wavelet Transform on FPGA," in *11th Annual IEEE Symposium of Field Programmable Custom Computing Machine (FCCM'03)*, 2003.

[8] K. H. Talukder and K. Harada, "Haar Wavelet Based Approach for Image Compression and Quality Assessment of Compressed Image," *IAENG International Journal of Applied Mathematics*, 2006.

[9] R. M. Jiang and D. Crookes, "FPGA Implementation of 3D Discrete Wavelet Transform for Real-Time Medical Imaging," IEEE, 2007.

[10] A. Ahmad, P. Nicholl and B. Krill, "Dynamic Parial Reconfiguration of 2-D Haar Wavelet Transfrom (HWT) for Face Recognition System".

[11] J. Chilo and T. Lindblad, "Hardware Implementation of 1D Wavelet Transform on an FPGA for Infrasound *Sign*al Classification," in *IEEE Transaction on Nuclear Science Vol. 55*, 2008.

[12] M. Vetterli and J. Kovacevic, Wavelet and Suband Coding, New Jersey: Prentice Hall PTR, 2007.

[13] M. Steinbuch and M. Van de Molengraft, Wavelet Theory and Application a Literature Study, Eindhoven: Eindhoven University Technology Department of Mechanical Engineering Control System Group, 2005.

[14] R. Wood, J. McAllister, G. Lightbody and Y. Yi, FPGA-based Implementation of *Sign*al Processing Systems, Wiltshire, United Kingdom: John Wiley & Sons, 2008.

[15] P. P. Chu, FPGA Prototyping By Verilog Examples (Xilinx Spartan-3 Version), New Jersey: John Willey & Sons, 2008.