DESAIN DAN SIMULASI PERANGKAT KERAS *MULTIOPERAND MSB-FIRST ADDER* BILANGAN BERTANDA MENGGUNAKAN VHDL

Arief Budijanto
Program Studi Teknik Elektro, Universitas Widya Kartika Surabaya
ariefbudijanto@widyakartika.ac.id

ABTRAKS

Msb-First Adder merupakan metode penjumlahan bilangan bulat yang dilakukan mulai dari bobot bit yang paling signifikan terlebih dahulu atau dari bit paling kiri menuju kekanan. Metode pemjumlahan ini mempunyai kelebihan jika proses penjumlahan dibatasi oleh waktu (*deadline*), karena yang dijumlahkan terlebih dahulu adalah bit yang mempunyai bobot paling signifikan. Selain itu metoda ini juga menunjukan kinerja yang lebih baik ketika digunakan untuk menjumlahkan bilangan banyak (*multioperand*). Dalam penelitian ini dibuat suatu arsitektur *MSB-First Adder* yang diaplikasikan untuk komputasi Transformasi Fourier Diskrit (TFD). Tahapan yang dilakukan dalam peneltian ini adalah desain diagram blok arsitektur MSB-First Adder dan Arsitektur TFD yang menggunakan MSB-First Adder, kemudian memodelkan dalam bentuk *VHDL Code* . Tahapan yang terakhir yaitu melakukan verifikasi dan analisis dari hasil komputasi TFD yang menggunakan MSB-First adder dibandingkan dengan TFD menggunakan LSB-First Adder dan MATLAB. Hasil akhir komputasi TFD *Multioperand MSB-First* dengan TFD *LSB-First* memperlihatkan hasil yang sama. Selain itu pada komputasi TFD *Multioperand MSB-First* tersedianya hasil-antara (*intermediate-result*) pada awal proses yang mendekati hasil akhirnya. Hasil ini tidak terjadi pada komputasi prosesor DFT *LSB-First*. Waktu yang dibutuhkan untuk proses komputasi TFD *Multioperand MSB-First* adalah 158,506 μS, sedangkan TFD *LSB-First* adalah 55,308 μS. Agar waktu komputasi TFD *Multioperand MSB-First* akan mendekati sama dengan waktu komputasi TFD *LSB-First*, jika pada bagian output *memory* (RAM dan ROM) dirubah menjadi 16 saluran output (tiap saluran 16 bit) dan *multiply* nya disusun paralel sebanyak 16 buah.

Kata kunci: MSB-First Adder, LSB-First Adder, Hasil-Antara, VHDL, TFD

PENDAHULUAN

Metoda penjumlah konvensional (*LSB-First Adder*) adalah metoda yang sering kali dipakai dalam keseharian untuk menjumlahkan suatu bilangan, yaitu dengan cara menjumlahkan bit yang bobotnya paling ringan terlebih dahulu (paling kanan), sedangkan menuju bobot paling berat (paling kiri). Metoda ini akan mempunyai kelemanhan jika proses dibatasi oleh waktu. Jika batas-waktu (*deadline*) kurang dari waktu yang digunakan untuk melakukan proses maka hasil penjumlahan mempunyai tingkat akurasi yang rendah, ini merupakan konsekuensi dilakukannya penjumlahan dari LSB[5,6].

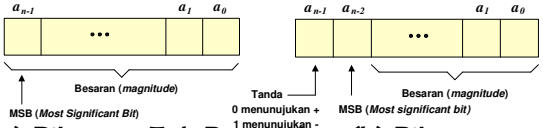
Metoda *Multioperand MSB-First Adder* adalah metoda penjumlahan yang dilakukan dengan menjumlahkan bit yang bobotnya paling berat terlebih dahulu (paling kiri/paling signifikan) dengan jumlah *operand* yang banyak. Metoda ini mempunyai kinerja yang baik, terutama jika batas-waktunya lebih kecil dari waktu yang dibutuhkan oleh proses, sebagaimana telah dibahas pada makalah[5,6] , Metoda ini juga menunjukan kinerja yang lebih baik ketika digunakan untuk menjumlahkan bilangan banyak (*multioperand*).

Kinerja yang telah dibuktikan dalam makalah [5,6,10] masih terbatas pada bilangan biner tidak bertanda sedangkan *Multioperand MSB-First Adder* telah didesain sebatas desain arsitektur[7] belum sampai tahap implementasi dalam *VDHL code*.
Dalam penelitian ini akan dibuat desain arsitektur *Multioperand MSB-First Adder* pada aplikasi arsitektur DFT (*Descrete Fourier Transform*) dan simulasinya menggunakan VHDL.

Bilangan Bulat Bertanda

Dalam sistem bilangan desimal, tanda dari suatu bilangan ditunjukkan dengan

simbol + atau – sebelah kiri digit yang paling berarti (*most-significant digit*). Dalam sistem biner tanda dari suatu bilangan ditunjukkan dengan bit yang letaknya paling kiri (*left-most bit*). Untuk bilangan positif bit paling kiri sama dengan '0', dan untuk suatu bilangan negatif sama dengan '1'. Dalam bilangan bertanda bit paling kiri merepresentasi tanda dan bit-bit sisanya $n-1$ merepresentasikan besaran. Seperti digambarkan dalam gambar 1(b). Lokasi bit yang paling kiri (*most significant bit (MSB)*) iniliah yang membedakan antara bilangan bertanda (*signed number*) dengan bilangan tidak bertanda (*unsigned number*). Dalam *unsigned number* semua bit merepresentasikan besaran dari suatu bilangan, sedangkan untuk *signed number* bit paling kiri (*MSB*) adalah a_{n-1} dan sisanya $n-1$ adalah besarannya ($a_{n-2},\ldots a_0$)[12].



(a) Bilangan Tak Bertanda (b) Bilangan Bertanda

Gambar 1. Format Representasi Bilangan Bulat[12]

Bilangan Negatif

Bilangan negatif dapat direpresentasi dalam tiga cara yang berbeda, yaitu: tanda-dan-besaran (*sign-and-magnitude*), komplement 1 (*1's complement*), dan komplemen 2 (*2's complement*).

a. Representasi *Sign-and-Magnitude*

Didalam representasi bilangan desimal yang sudah kita kenal, besaran dari kedua bilangan positif dan negatif diekspresikan dengan cara yang sama. Hanya simbol tanda yang membedakan suatu bilangan menjadi positif atau negatif. Cara ini disebut representasi bilangan *sign-and-magnitude*. Dengan cara yang sama dapat digunakan dalam bilangan biner pada bagian bit tandanya, untuk '0' berati bilangan positif dan '1' bilangan negatif. Sebagai contoh, jika kita menggunakan bilangan 4 bit, maka +5 = 0101 dan -5=1101. Karena kesamaanya dengan bilangan desimal *sign-and-magitude*, representasi ini mudah dimengerti[12]. Secara umum representasi *sign-and-magitude* diekspresikan sebagai[17]:

$$A = \begin{cases} \sum_{i=0}^{n-2} 2^i a_i & \text{Bila } a_{n-1} = 0 \\ -\sum_{i=0}^{n-2} 2^i a_i & \text{Bila } a_{n-1} = 1 \end{cases} \quad \ldots(1)$$

Terdapat beberapa kekurangan dalam representasi *sign-and-magitude*. Salah satunya adalah terdapat dua representasi untuk bilangan '0' :

$+0_{10}$ = 0000

-0_{10} = 1000 (*sign-and-magnitude*)

Hal ini tidak sesuai untuk digunakan, karena akan menyulitkan pemeriksaan bilangan '0' saat suatu operasi yang sering dilakukan oleh komputer jika dibandingkan dengan representasi tunggal[12].

b. Representasi *1's Complement*

Didalam cara *1's complement* sebuah bilangan negatif *K n*-bit, didapatkan dengan mengurangkan ekivalen bilangan positfnya *P*, dari 2^n-1, maka $K=(2^n-1)-P$. Sebagai contoh, jika $n=4$ maka $K=(2^4-1)-p=(15)_{10}-P$ $=(1111)_2-P$. Jika bilangan +5 dikonversikan menjadi suatu bilangan negatif, maka . Untuk lebih jelasnya, *1's complement* dapat dilakukan dengan mudah yaitu dengan cara mengkomplemenkan tiap bit dari bilangan, termasuk bit tandanya (*sign bit*)[12].

c. Representasi *2's Complement*

Representasi *2's complement* adalah suatu cara yang digunakan untuk mengatasi kekurangan yang terdapat pada representasi *sign-and-magnitude* dan *1's complement*. Pada representasi *sign-and-magnitude* mempunyai dua nilai nol yaitu +0 (0000) dan -0 (1000), seperti yang sudah dijelaskan diatas. Untuk representasi *1's complement* juga mempunyai dua buah nilai nol, yaitu +0 (0000) dan -0 (1111). Cara *2's complement* dapat dilakukan dengan menambah 1 dari bilangan yang telah di komplemenkan (*1's complement*). Misal bilangan desimal $-7_{(10)}$, untuk mencari nilai binernya maka $-7_{(10)}$ harus

dikonversikan kedalam biner positif, yaitu "0111". Kemudian biner "0111" dikomplemenkan sehingga menjadi "1000". Selanjutnya "1000" tambahkan dengan '1', sehingga didapatkan bilangan biner dari $-7_{(10)}$ adalah "1001".

Metoda Pemjumlah *MSB-Fisrt*

Dalam sub bab ini dijelaskan beberapa metoda pemjumlah MSB-First untuk banyak operand (*multioperand*) yang telah dilakukan oleh beberapa penelti sebelumnya.

a. Pemjumlah *Multioperand* Metoda *MSB-Fisrt* Bilangan Tak Bertanda

Metoda ini telah di kemukan pada makalah[5] . Argoritma tersebut diperlihatkan pada gambar 2.

b. Pemjumlah *Multioperand* Metoda *MSB-Fisrt* Bilangan Bertanda

Algoritma Pemjumlah *Multioperand MSB-Firs*t untuk Bilangan Bertanda yang dimodifikasi dapat dilihat pada gambar 3. Algoritma ini yang diterapkan pada penyelesain perihitungan Transformasi Fourier Diskrit.

Untuk semua operand di dalam register

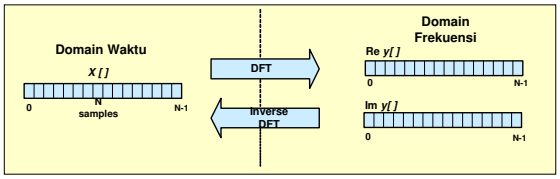
$R[N-1\ldots 0] < n-1\ldots 0 >; j := 1; ACC := 0$

1. Hitung: $\sum_{i=0}^{N-1} R[i] < n-1 >;$
2. $A \leftarrow \sum_{i=0}^{N-1} R[i] < n-1 > \#0 \,@\, n-j;$
3. $ACC \leftarrow A + ACC;$
4. $n \leftarrow n-1; j \leftarrow j+1;$ kembali ke langkah no.1,ulangi sampai $n < 0$

Gambar 2. Algoritma Pemjumlah *Multioperand* MSB-First Bilangan Tak Bertanda[5]

Trasnformasi Fourier Diskrit

Trasnformasi Fourier Diskrit (TFD) digunakan untuk mengubah sinyal diskrit dari domain waktu kedalam domain frekuensi. Ilustrasi TFD dapat dilihat pada gambar 4.



Gambar 4. Ilustrasi Transformasi Fourier Diskrit[9]

Persamaan untuk menghitung TFD adalah sebagai berikut[9]:

$$y[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \qquad k = 0 \le k \le N-1 \qquad \ldots(1)$$

$$W_N^{kn} = \cos\left(\frac{2\pi kn}{N}\right) - j\sin\left(\frac{2\pi kn}{N}\right) \qquad \ldots(2)$$

$$y[k] = \sum_{n=0}^{N-1} x[n]\{\cos(2\pi kn)/N) - j\sin(2\pi kn/N)\} \qquad \ldots(3)$$

Dari persamaan (3) spektrum dipisahkan ke dalam :

Bagian Real : $\mathrm{Re}\, y[k] = \sum_{n=0}^{N-1} x[n]\cos(2\pi kn/N)$

Bagian Imajiner : $\mathrm{Im}\, y[k] = -\sum_{n=0}^{N-1} x[n]\sin(2\pi kn/N)$

Magnitude : $|y[k]| = \{\mathrm{Re}\, y[k]^2 + \mathrm{Im}\, y[k]^2\}^{1/2}$

Phase : $\theta_k = \arctan\frac{\mathrm{Im}\, y[k]}{\mathrm{Re}\, y[k]}$

Gambar 5. komputasi TFD secara l

Input biner $\longrightarrow \quad D = D_{n-1}D_{n-2}\ldots D_1D_0$

2. Cek, Bit MSB input biner?

   jika MSB = '1' $\quad R_{PISO(positif)}[N-1\ldots0] < n-1\ldots0 > \leftarrow 2^{'S}\ comp(D[N-1\ldots0]) < n-1\ldots0 >;$

   jika MSB = '0'

   $R_{PISO(negatif)}[N-1\ldots0] < n-1\ldots0 > \leftarrow (D[N-1\ldots0]) < n-1\ldots0 >;$

3. $ACC < p-1\ldots0 > ; i = 0; k = p; ACC = 0;$
4. Untuk pencacah '1' bilangan positif dan negatif, Hitung:

   $X \leftarrow \sum_{i=0}^{N-1} R_{PISO(negatif)}[i] < (n-1) >; \quad Y \leftarrow \sum_{i=0}^{N-1} R_{PISO(positif)}[i] < (n-1) >;$

5. Hitung: $Z \leftarrow Y - X;$ (ket: Z adalah bilangan bertanda *2'S comp*)
6. $A \leftarrow Z_{n-1} @ (p-k)Z\#0 @ n-1;$
7. $ACC \leftarrow A + ACC;$
8. $k \leftarrow k-1; n \leftarrow n-1;$ ulangi ke langkah no.3 sampai $n < 0$

Gambar 3. Algoritma Pemjumlah *Multioperand MSB-First* untuk Bilangan Bertanda yang dimodifikasi.

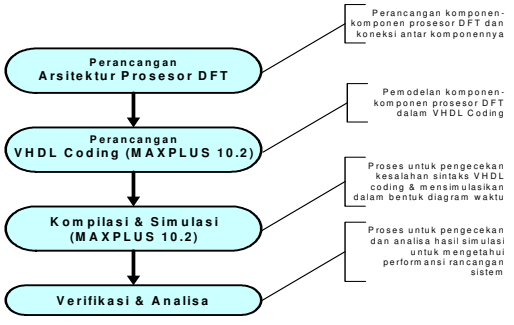
Dalam menghitung TFD dalam penelitian ini menggunaka algoritma TFD dengan metoda komputasi secara langsung (*Direct Computation of the DFT*) . Algoritma tersebut dapat dilihat pada gambar 5.

```
var  x: array[0..N-1]
var  y: array[0..N-1]
for k := 0 to N-1 do
   begin
      y[k] := x[0];
      for n := 1 to N-1 do
         y[k] := y[k] + Wnk * x[n];
end
```

Gambar 5. komputasi secara TFD langsung

METODOLOGI PENELITIAN

Metodologi perancangan sistem dalam penelitian ini ditunjukkan dengan diagram alir yang dapat dilihat pada gambar 5.



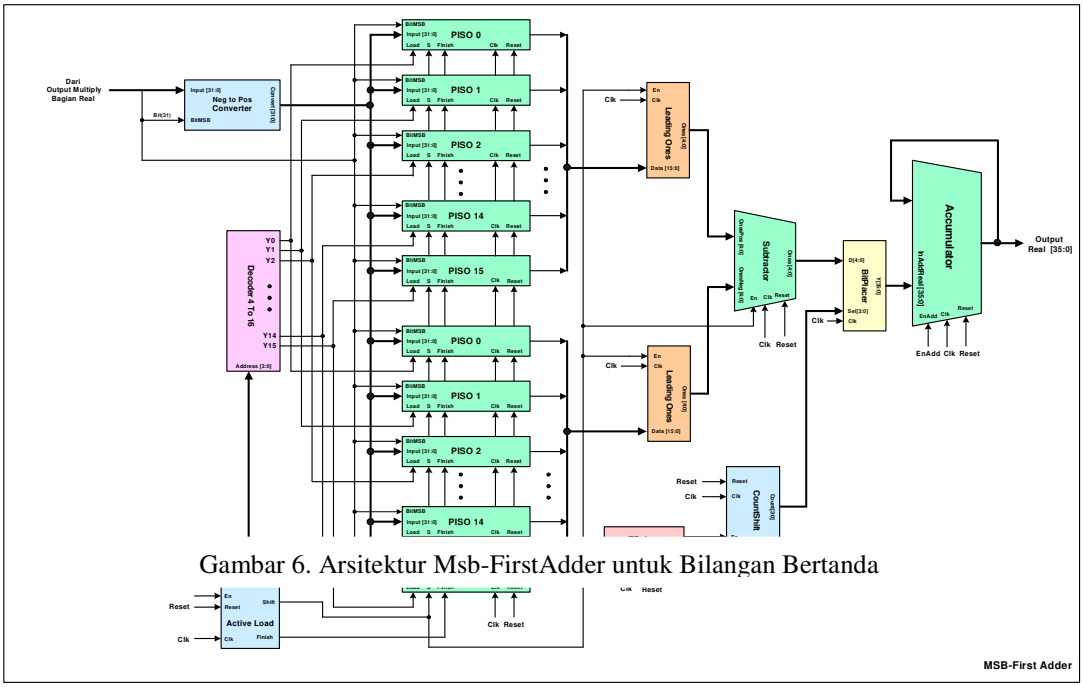
Arsitektur MSB-First Adder

Desain arsitektur *Msb-First Adder* untuk bilangan bertanda ditunjukkan pada gambar 6. Arsitektur tersebut terdiri d yari beberapa blok komponen yang dintegrasikan jadi satu. Fungsi dari tiap-tiap komponen dapat dilihat dalam tabel 1.

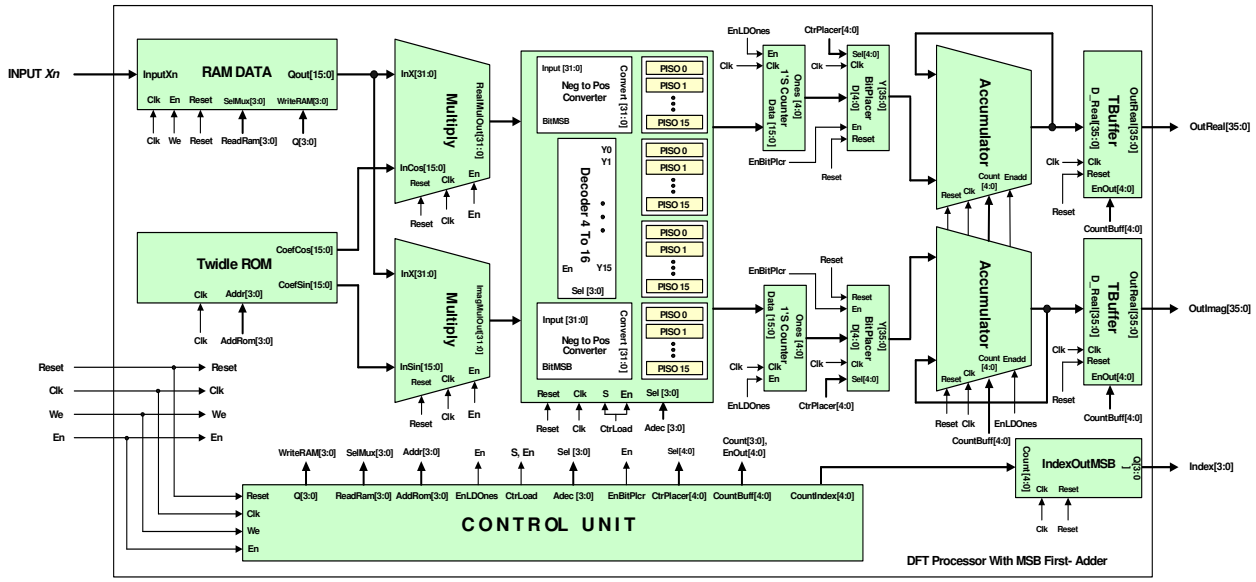
Tabel 1. Fungsi Komponen Pendukung Arsitektur MSB-First Adder

Nama Komponen	Fungsi
Converter Neg to Pos	Mengubah bilangan negatif menjadi positif dengan cara mendeteksi sign bit bialangan biner
Dekoder 4 to 16	Sebagai dekoder alamat PISO
Register PISO	Register geser untuk menggeser data dari masukan paralel 32 bit dengan keluaran bit per bit
Leading One	Berfungsi sebgai pencacah banyaknya bilangan '1' negatif dan positif
Bit Placer	Digunakan untuk menempatkan posisi data (D= D4 D3 D2 D1 D0) yang merupakan keluaran dari dari rangkaian Leading One
Accumulator	Untuk mengakumulasi dari hasil perkalian bilangan imajiner (koefisien sin) dan bilangan real (koefisien cos)

Sedangkan penerapan arsitektur Msb-First Adder untuk komputasi TFD ditunjukkan pada gambar 6.



Gambar 6. Arsitektur Msb-FirstAdder untuk Bilangan Bertanda

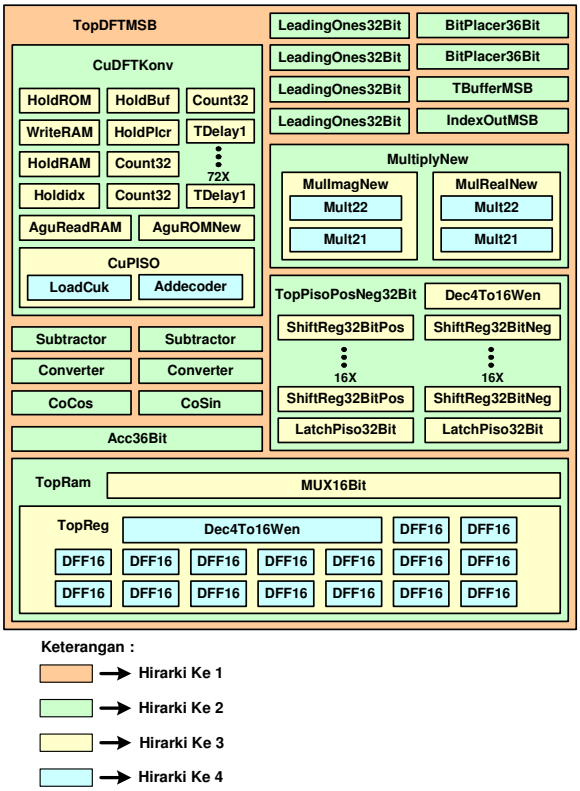


Gambar 7. Arsitektur Msb-First Adder untuk komputasi TFD

Implemen

VHDL Code dibuat dengan metoda pendeskripsian perilaku dari suatu rangakian tingkat tinggi, menganalisa dan menyempurnakannya sebelum melakukan implementasi rangkaian pada tingkat yang lebih rendah (*top-down modelling*).

Diagram blok implementasi VHDL code berdasarkan hirarkinya ditunjukkan pada gambar 8.



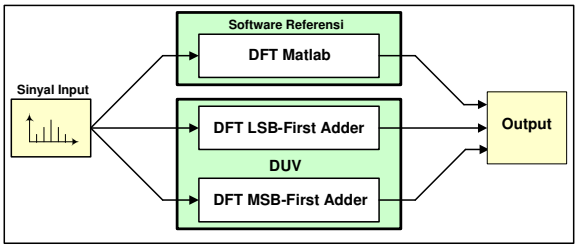
Gambar 8. Implementasi VHDL code Msb-First Adder untuk Komputasi TFD

HASIL DAN PEMBAHASAN

Verifikasi

Verifikasi dilakukan untuk menguji kebenaran rancangan arsitektur komputasi TFD yang menggunakan Multioperand MSB-First Adder dibandingkan dengan komputasi LSB-First Adder dan matlab.

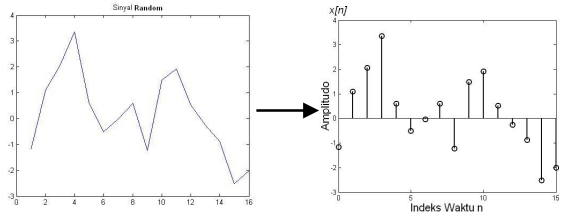
Diagram blok verifikasi sistem ditunjukkan pada gambar 9.



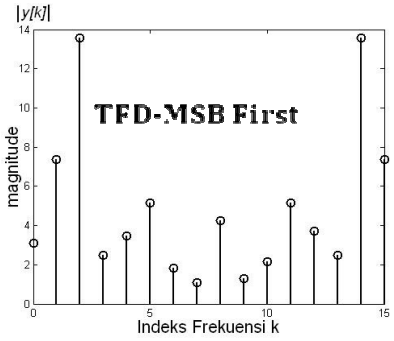
Gambar 9. Diagram Blok Verifikasi Arsitektur Komputasi TFD menggunakan *Multioperand MSB-First Adder*

Hasil Simulasi

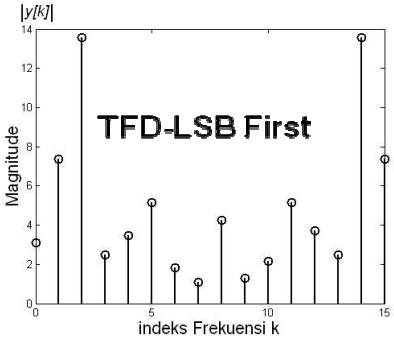
Hasil simulasi dari pengujian komputasi TFD dari sinyal random yang sampling 16 point dengan frekuensi sampling 10000 Hz ditunjukkan pada gambar 9.



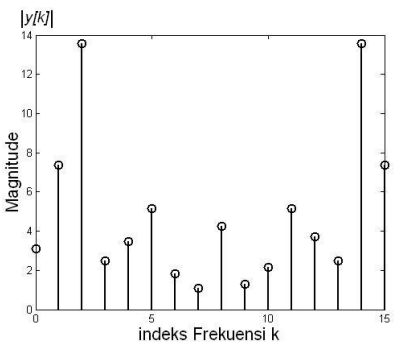
Gambar 9. Sinyal Random disampling 16 point dengan frekuensi sampling (fs=10000Hz)



(a)

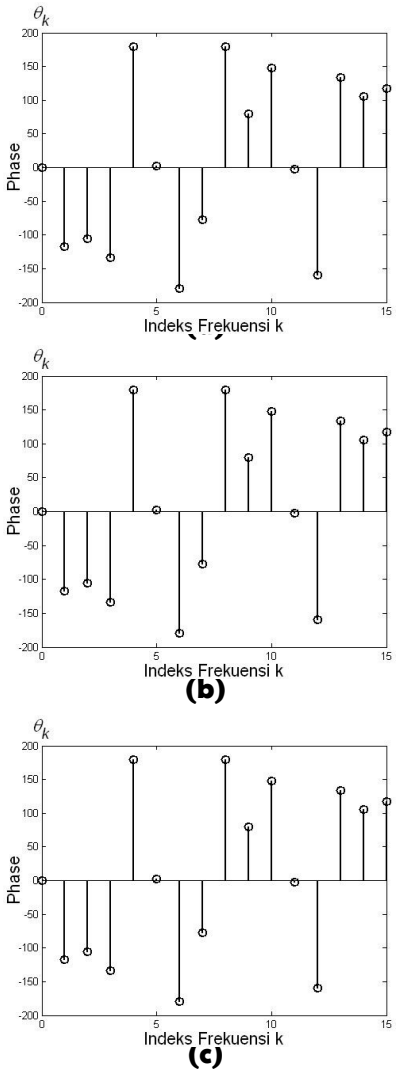


(b)



(c)

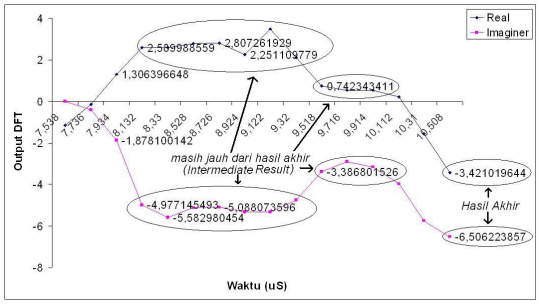
Gambar 10. Grafik *Magnitude* dari komputasi DFT 16 point untuk sinyal random



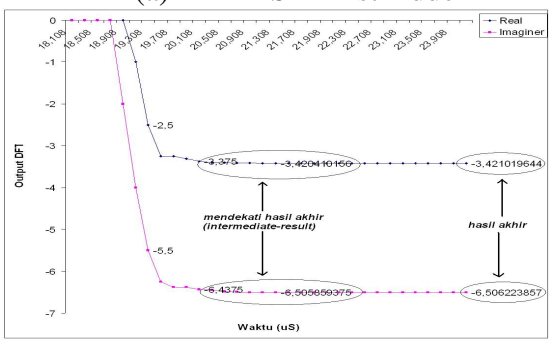
Gambar 11. Grafik *phase* dari komputasi DFT 16 point untuk sinyal random

Perbandingan Waktu Komputasi TFD

- Waktu Komputasi TFD 16 point
  Dengan LSB-First Adder = 55,308 µS
  Dengan MSB-First Adder = 158,506 µS
- Komputasi DFT MSB-First dengan TFD LSB-First mempunyai hasil yang sama
- DFT MSB-First pada awal proses tersedia *intermediate result* yang mendekati hasil akhir



(a) TFD LSB-First Adder



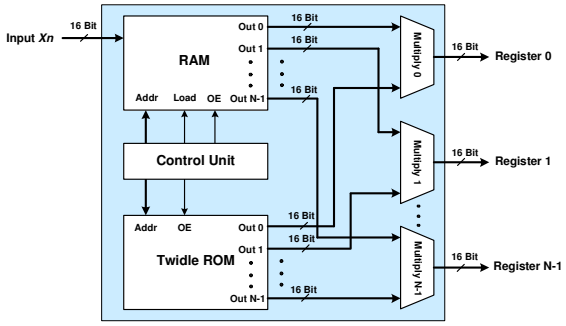
(b) TFD MSB-First Adder

Gambar 12. *Intermediate-Result* output *y[1]* dari komputasi DFT sinyal *random*

Saran Pengembangan

Dari hasil perbandingan waktu TFD Msb-First Adder dan Lsb-First Adder

- TFD MSB-First dalam menyelesaikan komputasi akan mempunyai waktu yang sama atau mendekati sama dengan TFD LSB-First, jika dilakukan modifikasi rancangan arsitekturnya pada bagian *memory* (ROM dan RAM) dan *multiply*
- Gambar rancangan arsitektur bagian memory (ROM dan RAM) dan multiply yang dimodifikasi sbb:



Gambar 13. Modifikasi Arsitektur Memori

KESIMPULAN

Berdasarkan hasil dari verifikasi dan analisa dapat ditarik kesimpulan sebagai berikut:

1. Hasil akhir komputasi prosesor DFT *Multioperand MSB-First* dengan prosesor DFT *LSB-First* memperlihatkan hasil yang sama.
2. Pada komputasi prosesor DFT *Multioperand MSB-First* tersedianya *intermediate-result* pada awal proses yang mendekati hasil akhirnya. Hail ini tidak terjadi pada komputasi prosesor DFT *LSB-First*
3. Waktu yang dibutuhkan untuk proses komputasi pada prosesor DFT *Multioperand MSB-First* adalah 158,506 μS, sedangkan prosesor DFT *LSB-First* adalah 55,308 μS.
4. Waktu komputasi prosesor DFT *Multioperand MSB-First* akan mendekati sama dengan waktu komputasi DFT *LSB-First*, jika pada bagian output *memory* (RAM dan ROM) dirubah menjadi 16 saluran output (tiap saluran 16 bit) dan *multiply* nya disusun paralel sebanyak 16 buah.

DAFTAR PUSTAKA

1. Heuring P. Vincent., H.F. Jordan,(1997), *Computer System Design and Architecture*, Addison-Wesley,New York.
2. Joseph J. F. Cavanagh.(1985), *Digital Computer Arithmetic: Design and Implementatio*n McGraw-Hill, Singapore.
3. James O. Hamblen & Michael D. Furman. (2001), *Rapid Prototyping of Digital Systems: A Tutorial Approach*, Kluwer Academic Publisher, USA.
4. Janick B.,(2000), *Writing Testbenches Fuctional Verification of HDL Models*, Kluwer Academic Press,New York.
5. Kuspriyanto, Kerlooza, Y.Y., (2004), *Toward New Real-Time Processor: The Multioperand MSB-First Real-Time Adder*, Proceedings of DSD'2004 Euromicro Symposium on Digital System Design, Rennes-France, IEEE Computer Society.
6. Kuspriyanto, Totok B., (2005), *Penjumlah Bilangan Biner Bertanda Menggunakan*
7. *Metode MSB-First*, Proceedings 6th Seminar on Intellegent Technology and Its Applications Surabaya Indonesia.
8. Lars Wanhammars.(1999). *DSP Integrated Circuits*, Academic Press, USA.
9. Stephen B. & Zvonko G.Vrinesic, (2000), *Fundamental of Digital Logic With VHDL Design*, McGraw-Hill, Singapore.
10. Steven W.Smith,(1999), *The Scientist and Engineer's Guide to Digital Signal Processing*,2nd, California Technical Pub.,San Diego, USA.
11. Totok Budioko.(2006), *Perancangan dan Simulasi Penjumlah Multioperand MSB-First Pada Filter Digital*, Tesis Magister, Institut Teknologi Bandung.
12. Volnei A. Pedroni.(2004), *Circuit Design With VHDL*, MIT Press.Massahuset.
13. William Stalling.(1996), *Computer Organization and Architecture, 4e: Designing for Performance*, Prentice Hall, Upper Saddle River, New Jersey.