ACCELERATION OF MOLECULAR DYNAMICS SIMULATION FOR TERSOFF2 POTENTIAL THROUGH RECONFIGURABLE HARDWARE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BİLGİN VARGÜN

IN PARTIAL FULLFILMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
MICRO AND NANOTECHNOLOGY DEPARTMENT

SEPTEMBER 2012

Approval of the thesis:

ACCELERATION OF MOLECULAR DYNAMICS SIMULATION FOR TERSOFF2 POTENTIAL THROUGH RECONFIGURABLE HARDWARE

Submitted by BİLGİN VARGÜN in partial fulfillment of the requirements for the degree of **Master of Science in Micro and Nanotechnology Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Mürvet Volkan _____
Head of Department, **Micro and Nanotechnology Dept., METU**

Prof. Dr. Şakir Erkoç _____
Supervisor, **Micro and Nanotechnology Dept.**, **METU**

Dr. Selim Eminoğlu _____
Co-Supervisor, **CTO, Mikrosens A.Ş.**

Examining Committee Members:

Prof. Dr. Şakir Erkoç _____
Micro and Nanotechnology Dept., METU

Prof. Dr. Kemal Leblebicioğlu _____
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Mehmet Kadri Aydınol _____
Metallurgical and Materials Engineering Dept., METU

Assoc. Prof. Dr. Ali Hikmet Doğru _____
Computer Engineering Dept., METU

Assoc. Prof. Dr.Veysi İşler _____
Computer Engineering Dept., METU

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Bilgin Vargün

Signature:

ABSTRACT

ACCELERATION OF MOLECULAR DYNAMICS SIMULATION FOR TERSOFF2 POTENTIAL THROUGH RECONFIGURABLE HARDWARE

Vargün, Bilgin
Msc. Department of Micro and Nanotechnology
Supervisor : Prof. Dr. Şakir Erkoç
Co-Supervisor : Dr. Selim Eminoğlu

September 2012, 72 pages

In nanotechnology, Carbon Nanotubes systems are studied with Molecular Dynamics Simulation software programs investigating the properties of molecular structure. Computational loads are very complex in these kinds of software programs. Especially in three body simulations, it takes a couple of weeks for small number of atoms. Researchers use supercomputers to study more complex systems. In recent years, by the development of sophisticated Field Programmable Gate Array (FPGA) Technology, researchers design special purpose co-processor to accelerate their simulations. Ongoing researches show that using application specific digital circuits will have better performance with respect to an ordinary computer.

In this thesis, a new special co-processor, called TERSOFF2, is designed and implemented. Resulting design is a low cost, low power and high performance computing solution. It can solve same computation problem 1000 times faster. Moreover, an optimized digital mathematical elementary functions library is designed and implemented through thesis study. All of the work about digital circuits

and architecture of co-processor will be given in the related chapter. Performance achievements will be at the end of thesis.

Keywords: Molecular Dynamics, FPGA, Tersoff2, Hardware Co-processor

ÖZ

TERSOFF2 POTANSİYELİNİN MOLECÜLER DİNAMİK BENZETİMİNİN PROGRAMLANABİLEN ÖZEL SAYISAL DEVRE DONANIMLARIYLA HIZLANDIRILMASI

Vargün, Bilgin
Yüksek Lisans, Mikro ve Nanoteknoloji Bölümü
Tez Yöneticisi: Prof. Dr. Şakir Erkoç
Ortak Tez Yöneticisi: Dr. Selim Eminoğlu

Eylül 2012, 72 sayfa

Nanoteknoloji alanında, Karbon nanotüp sistemlerinin moleküler yapılarının ve özelliklerinin araştırılmasında moleküler dinamik benzetim programları kullanılmaktadır. Bu tip programların içerisinde hesaplama yükü çok yüksek yapılar bulunmaktadır. Özellikle üçlü yapıların benzeşim hesapları bir kaç yüz atom için haftalar almaktadır. Bu durumda kalan araştırmacılar daha fazla atoma sahip sistemleri araştırırken süper bilgisayarları kullanmaktadırlar. Son yıllarda programlanabilen sayısal devre teknolojisindeki önemli gelişmeler sayesinde araştırmacılar özel sayısal devrelerden oluşmuş çözümler geliştirerek hesaplamalarını hızlandırmaktadırlar. Devam eden araştırmalar bu tarz devrelerin normal bilgisayarlara göre daha iyi başarım gösterdiğini söylemektedir.

Bu tez çalışmasında TERSOFF2 adında özel bir sayısal işlemci geliştirilmiştir. Tasarlanmış yapının düşük maliyetli, düşük güç tüketimli ve yüksek başarımlı oldu-

ğu görülmüştür. Aynı bilimsel hesaplama için, yüzlerce kat hızlanma sağlanmıştır. Bu tez çalışması yapılırken moleküler dinamik benzeşim problemleri için iyileştirilmiş matematik temel fonksiyonlarının sayısal devre kütüphanesi tasarlanmış ve üretilmiştir. Bunlarla alakalı detaylı bilgiler tezin ilgili bölümlerinde verilecektir. Geliştirilen sayısal işlemcinin başarımı tezin sonunda gösterilecektir.

Anahtar Kelimeler: Moleküler Dinamik, FPGA, Tersoff2, Özel İşlemciler

This thesis dedicated to humans who can read the universe

ACKNOWLEDGMENTS

I want to express my gratitude to my supervisor Prof. Dr. Şakir Erkoç and co-supervisor Dr. Selim Eminoğlu for their guidance, advice, criticism, encouragements throughout the research.

Also I want to express my special thanks to TÜBİTAK İLTAREN Institute and Kıvanç Günalp.

TABLE OF CONTENTS

ABSTARCT.....iv

ÖZ vi

ACKNOWLEDGMENTS.....ix

LIST OF FIGURES.....xii

LIST OF TABLES.....xv

CHAPTERS

1-INTRODUCTION.....1
1.1Molecular Dynamics Simulations.....1
1.2 Examples on Hardware Acceleration of MD Simulations.....3
2-TERSOFF 2.....6
2.1TERSOFF2 Empirical Many Bodies Potential.....6
2.2TERSOFF2 Processor Main Finite State Machine.....12
3-LIBRARY.....14
3.1 Molecular Dynamics Elementary Functions Library.....14
3.2 Molecular Dynamics Simulation Library and
Tersoff2 Co-Processor Architecture 17
3.2.1Elementary Mathematics Library Design.....18
3.2.1.a Exponential Function.....18
3.2.1.b Square Root Inverse Square Root Function.....20
3.2.1.c Power Function.....24
3.2.1.d Log-adder Function.....26
3.2.1.e Logarithm Function.....26
3.2.2 Complex Mathematics Library Design.....28
3.2.2.a Radius Computation Function.....30
3.2.2.b Cache Structure for Tersoff2.....31
3.2.2.c Carbon Pair Generator Function.....35
3.2.2.d Verlet Update Function.....36
3.2.2.e Thermostat Function.....38
3.2.2.f Intelligent Reset Function.....40
3.2.2.g Cosine Function.....40
3.2.2.h Projection Function.....42
3.2.2.i Handshaking Strategy.....43
3.2.2.j Synchronization strategy.....43
4-TERSOFF2 Hardware Acceleration Kernel.....45
4.1Kernel.Clerify.....47

4.2Kernel.Coordinate……..47
4.3Kernel.Energy-Pair……48
4.4Kernel.GTETA Function……..52
4.5Kernel.Projection……..54
4.6Kernel.TwoPairsSummation……54
4.7Kernel.ThreePairsSummation……...56
4.8Kernel.PowerSystem……58
4.9Kernel.ForceThreePairs……61
4.10Kernel.ForceM……63
4.11Kernel.EnergyM……..65
5-CONCLUSIONS……66

REFERENCES……...69

LIST OF FIGURES

FIGURES

Figure 1 Basic structure of MD calculations.....2
Figure 2 Basic atomic structure.....8
Figure 3 Main FSM diagram for Tersoff2 processor.....12
Figure 4 Black box input output diagram of exponential module.....18
Figure 5 Schematic exponential computation diagram.....20
Figure 6 Black box diagram of square root inverse square root module.....20
Figure 7 Abstract initial single precision Rs computation diagram.....22
Figure 8 Starting with single precision Rs and modified Goldschmidt structure the both result can be obtained at the same time.....23
Figure 9 Double precision floating point black box diagram.....24
Figure 10 We can reduce power computation resources by eliminating the second natural logarithm module.....25
Figure 11 Accumulation of 8 terms with 7 adders in 3 stage addition.....26
Figure 12 Accumulation of 8 terms with 3 adders and 2 simple delay units.....26
Figure 13 Approximation architecture of logarithm function.....27
Figure 14 Schematic diagram of logarithm function.....28
Figure 15 The radius square black box diagram with square root black box diagram.....29
Figure 16 Schematic for computing distance and inverse distance.....30
Figure 17 Schematic for basic memory for xi, xj and xk.....31
Figure 18 Black Box Diagram for coordinates memory structure.....32
Figure 19 Black Box Diagram for neighbor list module.....33
Figure 20 Black Box Diagram for all of the memory blocks.....34
Figure 21 Black Box Diagram for carbon pair generator.....35
Figure 22 Black Box Diagram for Verlet Update.....36
Figure 23 Schematic diagram of main Verlet computation.....37
Figure 24 Black box diagram of three axis Verlet update.....37
Figure 25 Black box diagram of Thermostat.....38

Figure 26 Schematic diagram for Thermostat.....39
Figure 27 The black box representation of intelligent reset circuit.....40
Figure 28 The black box diagram of cosine angle module.....40
Figure 29 Abstract diagram of cosine angle module.....41
Figure 30 Schematic diagram of cosine angle module.....42
Figure 31 Schematic diagram of Projection module.....43
Figure 32 The black box representation of Kernel of Tersoff2.....45
Figure 33 Coordinate module diagram.....47
Figure 34 Energy Pair module diagram.....48
Figure 35 Schematic diagram for FCR cut off function.....49
Figure 36 Schematic diagram of single energy pairs.....50
Figure 37 Schematic diagram for energy pairs and their derivatives.....51
Figure 38 Schematic diagram of DerivativeForce module.....52
Figure 39 The black box representation of GTETA device.....52
Figure 40 Schematic diagram of GTETA function.....53
Figure 41 The back box representation of Twopairssummation module.....54
Figure 42 Schematic diagram of Twopairssummation module.....55
Figure 43 Schematic diagram of Force from two pair energy.....56
Figure 44 The black box diagram of ThreePairssummation Device.....56
Figure 45 schematic diagram of ThreepairsSummation Device.....57
Figure 46 Schematic diagram of DWIJKm Device.....58
Figure 47 The black box diagram for powersystem module.....58
Figure 48 Schematic diagram for power calculation.....59
Figure 49 Detailed Schematic diagram for WIJKn, WIJKn1,
PARAN and PARAN1power calculation.....60
Figure 50 The black box diagram for Forcethreepairs device.....61
Figure 51 Schematic diagram for Forcethreepairs device.....62
Figure 52 The Black box diagram ForcetM device.....63
Figure 53 Schematic diagram ForcetM device.....64
Figure 54 Tersoff2 Main Architecture.....66
Figure 55 Speedup due to pipelined architecture of Tersoff2.....67

Figure 56 Computation time comparison……......67
Figure 57 Tersoff2 time budget ……......68

LIST OF TABLES

TABLES

Table 1 Tersoff 2 Potential Constants……...7

CHAPTER 1

INTRODUCTION

1.1MOLECULAR DYNAMICS SIMULATIONS

Molecular Dynamics (MD), which based on Newtonian Mechanics, is a technique for analysis and design of molecular system. In physics, chemistry and biology researchers utilize special simulation techniques to solve dynamics of the molecular systems. Detailed information about the theory of MD is given in [1], [2].

This method is used with Empirical Potential Functions for Many-Body system given in [1]. System potential is based on Bonn-Oppenheim approximation and for N related particle total energy is;

$$E = \emptyset_{1+}\emptyset_{2+}\emptyset_{3+}\emptyset_{1+} \dots \emptyset_n \quad (1.1)$$

If we subtract the non interacting particle's energy $\emptyset_1$, the total energy of interacting particles with respect to their position is;

$$\emptyset = \emptyset_2 + \emptyset_3 + \cdots \emptyset_n \quad (1.2)$$

Where;

$$\emptyset_2 = \sum_{i<j} U2(r_i, r_j), \quad (1.3)$$

$$\emptyset_3 = \sum_{i<j<k} U3(r_i, r_j, r_k) \quad (1.4)$$

U_2 in 1.3 and U_3 in 1.4 represent two, three body interactions [1].

According to [1], [2] MD Simulation with respect to its potential can be done with following procedure in Figure 1.

1. Empirical Many Body Potential has to be computed,
2. The force 1.5 on each atom in the system have to be computed from derivative of 1.2

$$F = -\nabla.\emptyset \quad (1.5)$$

3. From 1.5 the new velocities and positions have to be computed with respect to time difference between MD steps.
4. The velocity redistribution has to be controlled over system with respect to its thermal stability conditions.
5. All steps have to be recomputed for each MD steps until system reaches desired time limit.

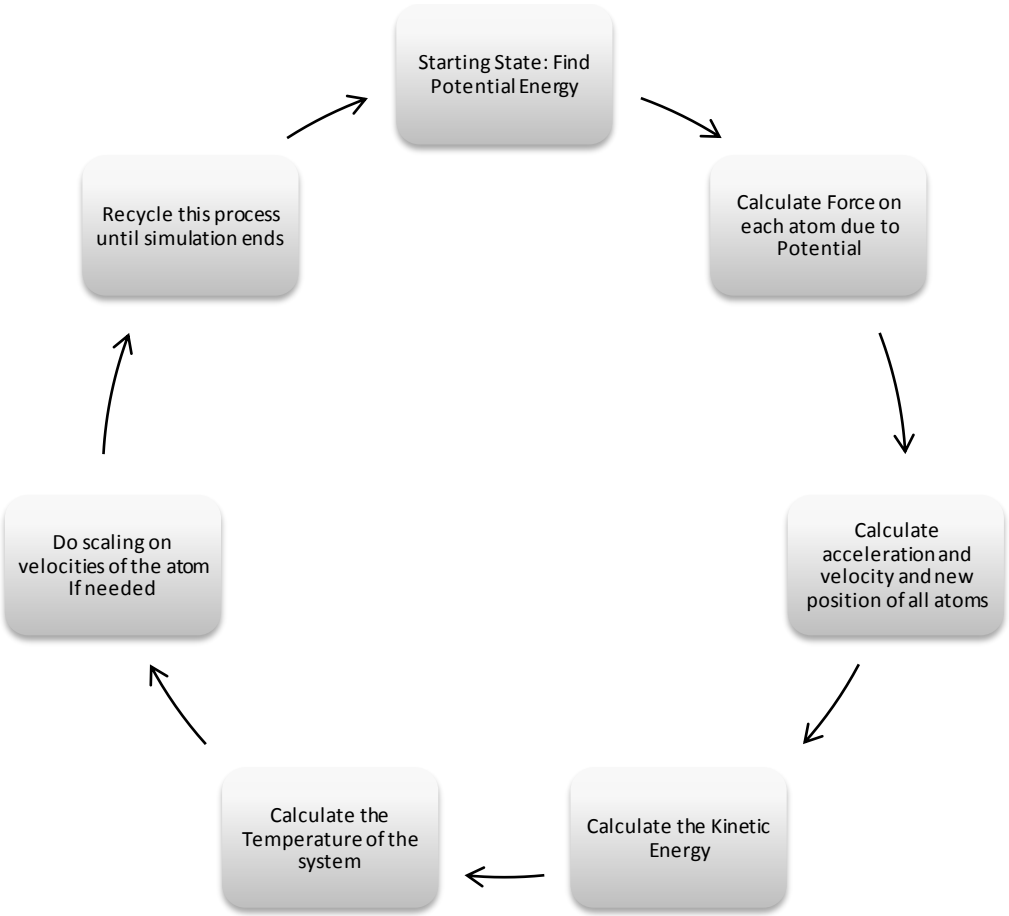


Figure 1: Basic structure of MD calculations.

With this brief introduction, we must state the computation problem. When we look at the computation structure we have N^2 computation complexity for two body interaction system and N^3 computation complexity for three body interaction system. Since most of the potentials has complex and difficult computation functions, simulations take very long times (weeks and months).

For the problem of computation time, researchers use super computer system to get a reasonable simulation time cost. However, super computers are not easily reachable tools to use. With the recent development in FPGA technology, researchers study on reprogrammable hardware system to design special digital circuits. When we review such kind of works we understand that, there will be a promising solution for computation problems by FPGA technology. Most of these kinds of designs are in the bioinformatics area. After giving some examples on reprogrammable hardware accelerator for MD simulation system, we are going to explain our potential and design.

1.2 EXAMPLES ON HARDWARE ACCELARATION OF MD SIMULATIONS

Reconfigurable Molecular Dynamics Simulator:

This design is given by Azizi, N. Kuon, I. Egier A. The suggested work is accelerating the non-bonded interaction by Lennard-Jones Potential [9].

$$\emptyset_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad (1.6)$$

$$F = -\nabla_r \emptyset_{LJ}(r) \quad (1.7)$$

In this work, a pair generator, Lennard-Jones calculator, Acceleration update, Verlet Update, three memory controllers and System controller modules designed. According to paper, they speed up computation with different precisions up to X21. When we investigate the paper, their work didn't consist of double precision floating point structure. And they reach acceleration with a reasonable error. Better performance can be achieved by more sophisticated pair generator block and memory structure for 1.6.

FPGA-Acceleration Molecular Dynamics simulations An Overview:

This design is given by Yang, X. Shengmei, M. Yong, D [10]. The design is also deal with accelerating Lennard-Jones Potential [9].

$$U_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad (1.8)$$

Force Calculator, Verlet Update, Pair Generator and Memory access system are designed. According to paper, they reach 2 times acceleration of computation. They also have 31 bit time resolution. A linked list method is utilized to for more efficient computation.

Preliminary Investigation of Advanced Electrostatics in Molecular Dynamics on Reconfigurable Computers:

This design is given by Scrofano, R. Prasanna, V.[11]. This design is accelerate Lennard-Jones Potential [9] and Coulomb potential.

$$\emptyset_{LJ}(r) = 4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad (1.9)$$

$$F = -\nabla_r \emptyset_{LJ}(r) \quad (1.10)$$

With reconfigurable computation technique, they reach almost 3 times speedup over software solution. Neighbor list technique is used and off-chip memories are utilized.

Efficient and Accurate FPGA-based simulator for Molecular Dynamics:

This work is done by Cho, E. and Bourgeois, A. [12]. It also investigates Lennard-Jones Potential [9] and Coulomb potential. They propose a new architecture for Coulomb force named Multi Level Charge assignment method. They use high level programming (Simulink and SysGen) structure to generate digital circuit. Accelerator has X10 to X100 better performances without a loss of accuracy.

FPGA-Based Three-Body Molecular Dynamics Simulator:

In this paper [13], Pottathuparambil and Sass designed a pipelined accelerator for Stillinger-Weber Potential [14].

$$\varphi = \sum_{i<j} U_{ij} + \sum_{i<j<k} W_{ijk} \quad (1.11)$$

Where;

$$U_{ij} = \varepsilon x f_2(r) \quad (1.12)$$

$$f_2(r) = \begin{cases} A(Br^{-p} - r^{-q})e^{(r-a)^{-1}} & r < a \\ 0 & r \geq a \end{cases} \quad (1.13)$$

$$W_{ijk} = \varepsilon f_3\left(\frac{r_i}{\sigma}, \frac{r_j}{\sigma}, \frac{r_k}{\sigma}\right) \quad (1.14)$$

$$f_3(r_i, r_j, r_k) = h(r_{ij}, r_{ik}, \theta_{jik}) + h(r_{ji}, r_{jk}, \theta_{ijk}) + h(r_{ki}, r_{kj}, \theta_{ikj}) \quad (1.15)$$

$$h(r_{ij}, r_{ik}, \theta_{jik}) = \lambda e^{[\gamma(r_{ij}-a)^{-1}+\gamma(r_{ik}-a)^{-1}]} x(\frac{1}{3} + \cos\theta_{jik})^2 \quad (1.16)$$

For benchmarking, reference computer has Dual AMD core running at 2.0 GHz. At the end, they have 1.5 times faster solution for the computation of Stillinger-Weber. The design is consist of an off-chip DDR memory system for I/O applications.

FPGA-Accelerated Molecular Dynamics Simulation System:

In this work [15] Lennard-Jones Potential 1.9 is investigated. The design includes Cell-List Method, Pair Generator and Lennard-Jones force calculator. Reference system is a PC with 2.66 GHz Pentium 4 CPU and resulting performance is 12 times speedup. Since designers use fixed-point number system, they have %1 error with their results.

Accelerating molecular dynamics simulations with configurable circuits:

In this work [16], Lennard-Jones Potential and Coulomb force are investigated. They have three memory architectures for Position, Velocity and Force datas. The forces are computed with look-up tables and linear interpolation.

Most of the researches on Accelerating Molecular Dynamics Simulation with FPGA are deal with Lennard-Jones and Coulomb force [36], [37], [38,], [39], [40].

CHAPTER 2

TERSOFF 2

2.1TERSOFF2 EMPIRICAL MANY BODIES POTENTIAL

This potential energy function is developed for silicon and carbon covalent systems [4]. Empirical potential of Tersoff2, which express the total energy of the atomic system as an explicit function of the atomic positions are very useful [4] for carbon nanotube systems. With Tersoff2 many research on carbon nanotube system have been done for investigating different properties [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29]. In this thesis, the analysis of Tersoff2 is out of scope by means of crystal stability, phonon dispersion and other properties [30], [31], [32], [33], [34], [35]. Instead of the properties of Tersoff2 we are going to analyze computation of it.

Tersoff 2 has the following sets of equations;

$$\emptyset = \emptyset_{2,3} = \sum_{i<j:k} U_{ij:k} \tag{2.1}$$

2.1 is the sum of two and three body potentials. The potential on a single atom due to two and three body effects is;

$$U_{ij;k} = f_c(r_{ij}) \times \left[f_R(r_{ij}) + b_{ij}\, x f_A(r_{ij})\right] \tag{2.2}$$

Where;

$f_c(r_{ij})$ is the cut-off function,

$$f_c(r) = \begin{cases} 1, & r < R - D \\ \frac{1}{2} - \frac{1}{2} x \sin\left(\frac{1}{2} x \pi \frac{(r-R)}{D}\right) & R - D < r < R + D \\ 0 & r > R + D \end{cases} \tag{2.3}$$

$f_R(r_{ij})$ is the repulsive part of the potential in 2.2,

$$f_R(r) = Axe^{-\lambda_1 r} \tag{2.4}$$

$f_A(r_{ij})$ is the attractive part of the potential in 2.2,

$$f_A(r) = -Bxe^{-\lambda_2 r} \tag{2.5}$$

b_{ij} is the three pair potential in 2.2,

$$b_{ij} = (1 + \beta^n \zeta^n)^{\frac{-1}{2n}} \quad (2.6)$$

Where ζ_{ij} in 2.6;

$$\zeta_{ij} = \sum_{k \neq i,j} f_c(r_{ik}) x g(\theta_{ijk}) x e^{(\lambda_3 x (r_{ij} - r_{ik}))^3} \quad (2.7)$$

Where $g(\theta)$ in 2.7;

$$g(\theta) = 1 + \frac{c^2}{d^2} + \frac{c^2}{d^2 + (h - \cos\theta)^2} \quad (2.8)$$

The constants values are in the Table1.

Table1: The constants values

A(Ev)	**1393.6**
B(eV)	**346.74**
λ1(1/Å)	**3.4879**
λ2(1/Å)	**2.2119**
λ3(1/Å)	**0**
β	**1.57240E-07**
n	**0.72751**
h	**-0.57058**
c	**3.80490E+04**
d	**4.3484**
R(Å)	**1.95**
D(Å)	**0.15**

We have to compute all of the above equations and also derivative of them because force on each atom due to system's potential is;

$$F = -\nabla . \emptyset \quad (2.9)$$

Design of Tersoff2 processor computation kernels start with decomposition of computation structure. In this study, decomposition of computations starting point is the FORTRAN implementation of Tersoff2 potential which is written by Prof. Şakir Erkoç in 2000.

We start with pseudo code of Tersoff2 and decomposition of the equations;

The computation structure has three for loops;

for .....i.....

% Do all of the computation for all atoms

for.....j.....

%Do computation due to ij two pair energy

for.....k.....

%Do computation depends on ijk three pair energy

end

end

end

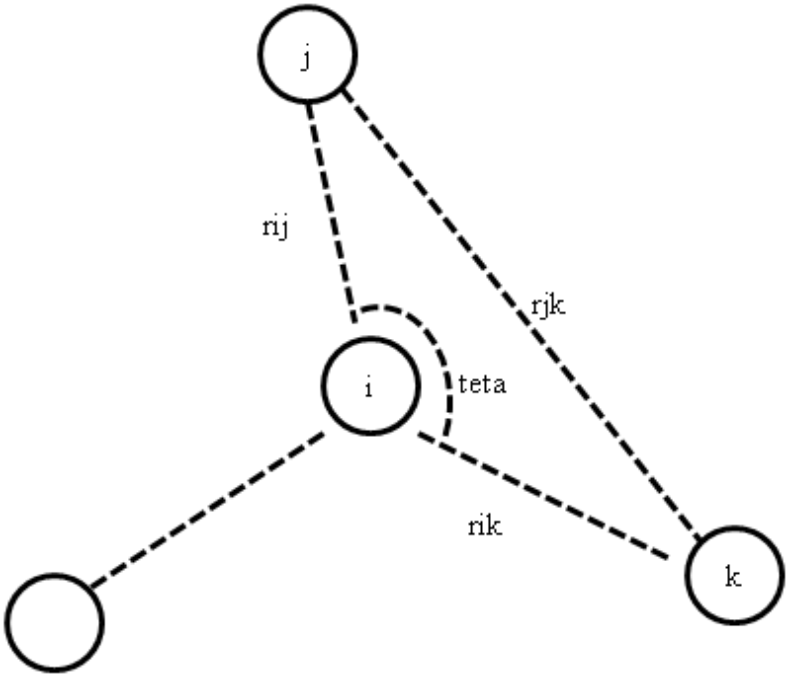


Figure 2: Basic atomic structure.

With the following pseudo code;

for i=1:1:NA %NA.....Number of Atoms

for j=1:1:NA

%.. find distances for all i!=j

% i.e. RIJ the distance between i and j atom,

% i.e. XIJ, YIJ, ZIJ the distance between i and j atom,

%Find the cut off values

```
    if (RIJ<R-D)
            FCIJ=1/2-sin(pi*(RIJ-R)/(2*D))/2;
            DFCIJ=-pi/(4*D)*cos(pi*(RIJ-R)/(2*D));
    else
            FCIJ=1;
            DFCIJ=0;
     end

    UIJ1=FCIJ*exp(-LAMDA1*RIJ); % Find the two pair repulsive energy
    E2=E2+UIJ1;              % accumulate for all i!=j
    DUIJ1=(DFCIJ-LAMDA1*FCIJ)*exp(-LAMDA1*RIJ); % Find the derivative
    XRIJ=XIJ/RIJ;            % find the projections
    YRIJ=YIJ/RIJ;            % find the projections
    ZRIJ=ZIJ/RIJ;            % find the projections

    FX2=FX2+DUIJ1*XRIJ;      % Repulsive Force accumulated due to j body
    FY2=FY2+DUIJ1*YRIJ;      % Repulsive Force accumulated due to j body
    FZ2=FZ2+DUIJ1*ZRIJ;      % Repulsive Force accumulated due to j body
    UIJ2=FCIJ*exp(-LAMDA2*RIJ); % Find the two pair attractive energy
    DUIJ2=(DFCIJ-LAMDA2*FCIJ)*exp(-LAMDA2*RIJ);

    FX32=DUIJ2*XRIJ;         % attractive Force accumulated due to j body on X
    FY32=DUIJ2*YRIJ;         % attractive Force accumulated due to j body on Y
    FZ32=DUIJ2*ZRIJ;         % attractive Force accumulated due to j body on Z

    WIJK=0;
       % Initilize Three pair computation components
    XDWIJK=0;
    YDWIJK=0;
    ZDWIJK=0;



for k=1:1:NA % Do the three pair computation for k!=i,j
    if (RIJ<R-D)
       FCIK=1/2-sin(pi*(RIK-R)/(2*D))/2;
       DFCIK=-pi/(4*D)*cos(pi*(RIK-R)/(2*D));
    else
       FCIK=1;
       DFCIK=0;
```

```
        end

        XRIK=XIK/RIK;
        YRIK=YIK/RIK;
        ZRIK=ZIK/RIK;

        CTIJK=(RIJ2+RIK2-RJK2)/(2*RIJ*RIK); % compute the cosine between ijk
        GTETA=1+c^2/d^2-c^2/(d^2+(h-CTIJK)^2); % g (teta ) defined in terosff2

        WIJK=WIJK+FCIK*GTIJK; % accumulation three pair energy

        DGTIJ=c^2*(h-CTIJK)/(d^2+(h-CTIJK)^2)^2*
                                    (1/RIK-RIK/RIJ2+RJK2/(RIJ*RIJ*RIK));
        DGTIK=c^2*(h-CTIJK)/(d^2+(h-CTIJK)^2)^2*
                                    (1/RIJ-RIJ/RIJ2+RJK2/(RIK*RIK*RIJ));
        DWIJ=FCIK*DGTIJ;
        DWIK=DFCIK*GTIJK+FCIK*DGTIK;
        XDWIJK=XDWIJK+DWIJ*XRIJ+DWIK*XRIK;
        YDWIJK=YDWIJK+DWIJ*YRIJ+DWIK*YRIK;
        ZDWIJK=ZDWIJK+DWIJ*ZRIJ+DWIK*ZRIK;
      End

        WIJKN=WIJK^n;
        WIJKN1=WIJK^(n-1);
        PARAN=(1+BETAN*WIJKN)^US;
        PARAN1=(1+BETAN*WIJKN)^US1;
        FX3=FX3+FX32*
              PARAN-0.5*BETAN*UIJ2*PARAN1*WIJKN1*XDWIJK;
        FY3=FY3+FY32*
              PARAN-0.5*BETAN*UIJ2*PARAN1*WIJKN1*YDWIJK;

        FZ3=FZ3+FZ32*
              PARAN-0.5*BETAN*UIJ2*PARAN1*WIJKN1*ZDWIJK;
        E3=E3+UIJ2*PARAN;
    end
        FX=-A*FX2+B*FX3;
        FY=-A*FY2+B*FY3;
        FZ=-A*FZ2+B*FZ3;
end
        EPOT=(A*E2-B*E3)/2;
```

To have a clear view on computations we can use the following logical way;

1. First of all, we calculate the distances rij, rij2, xij, yij, zij…
2. Find the cut off result (fcr)
3. Find repulsive two pair energy component uij1 and accumulate it.
4. Find attractive two pair energy uij2.
5. Find derivative of uij1, uij2 and projection parameters
6. Find cut off value fcr for ik pair.
7. Find projections for ik.
8. Find cosine of angle between ijk and g (Θ).
9. Find derivative of g(Θ).
10. Find three pair energy and accumulate it.
11. Find derivative of three pair energy and accumulate it.
12. Find force from F2 and F3.
13. Find potential energy from E2 and E3.
14. Do same procedure for all atoms in the system.
15. At the end compute kinetic energy and temperature
16. Calculate the scaling factor and do scaling for every two MD step.
17. Return the beginning of code until the simulation end.

When we analysis the code we see following properties;

- Computation has N^3 computation complexity due to 3 for loops.
- It will take very long time when we increase the atom number.
- Computations include very complex mathematical process.
- Computation has operations which consists of 3, 4, 5 operands.

To overcome these problems we can use pipelining, parallel computation and special digital circuit techniques. In this work, we follow some rules to get maximum computation performance.

1. Use pipeline structure to overcome the complexity problem.
2. At the end get a linear architecture to get minimum computation over cost in bigger atomic system.

3. Decompose all of the mathematical expression in two operant structure and design fully pipelined, single result per cycle for complex mathematical functions.
4. Design efficient memory architecture to achieve best timing performance.

2.2 TERSOFF2 PROCESSOR MAIN FINITE STATE MACHINE

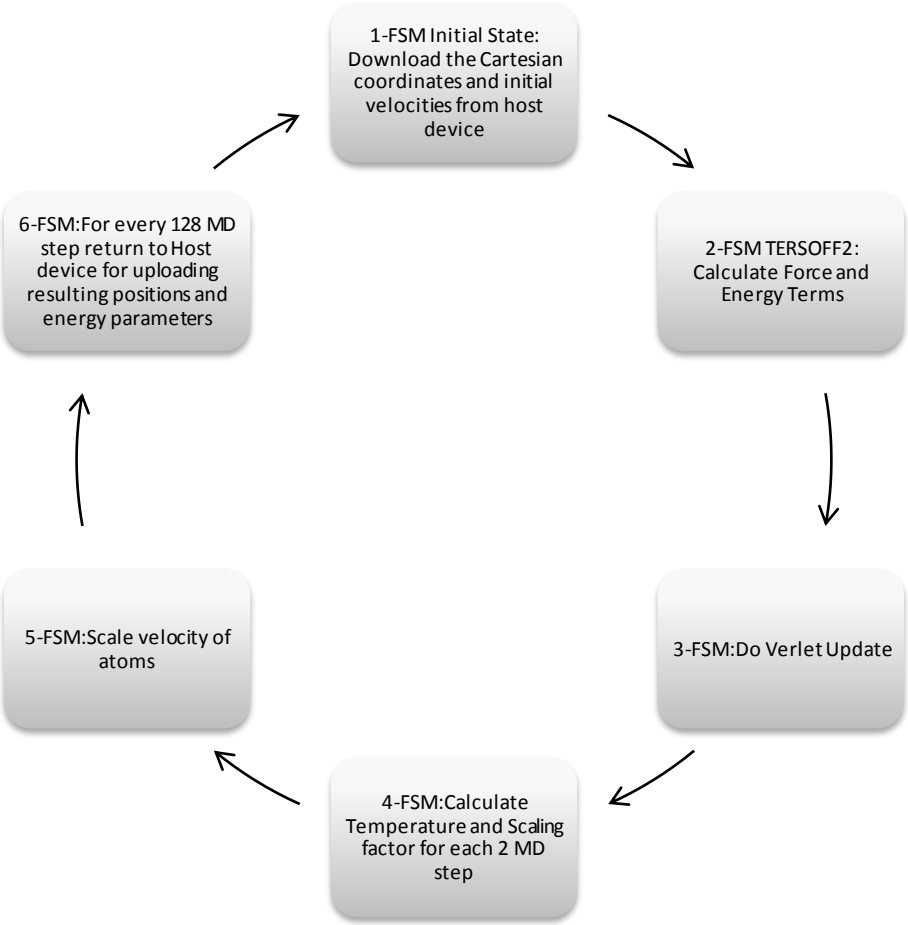


Figure 3: Main FSM diagram for Tersoff2 processor.

In Tersoff2 the main FSM design controls the all architecture. In real implementation, there are more states for design simplicity. For example, uploading state includes more than 20 states for each part of Cartesian coordinates. i.e. xi downloading, xj downloading….

In Verlet Updating state, when we finish the all updating sequence we do not step the Temperature and Scaling Factor state, instead we return the force and energy state for every two MD steps. Moreover, we directly turn to Force and energy state unless we reach the 128 MD step.

Host device can be any system that includes Ethernet ports. The interface of TERSOFF2 processor is Ethernet hardware instance of utilized FPGA.

CHAPTER 3

LIBRARY

3.1 MOLECULAR DYNAMICS ELEMENTARY FUNCTIONS LIBRARY

In Tersoff2 potential, we have to deal with some basic elementary mathematical functions such as; exponential, power, square root, and reciprocal of square root with double precision floating point. In this library the elementary functions are optimized for molecular dynamics and FPGA technology. In the case of computation optimization we have to change the structure of computation in accordance with digital circuit and we have to change elementary functions with respect to FPGA resource technology. The optimizations will be explained in detail in the library instances. All of library implemented in Verilog HDL. All of the functions are in double precision floating point arithmetic. Library is consisting of elementary functions and complex functions which are related to Tersoff2. A brief explanation is given with following part. This library consists of following elementary functions;

Exponential Function:

This function is designed with respect to [5]. In this paper, pipelined, well suited architecture for exponential computation is proposed.

Square Root Inverse Square Root Function:

This function is optimized for produce sqrt(x) and 1/sqrt(x) at the same time and it is designed with respect to [6]. In this paper fully pipelined independent sqrt and 1/sqrt function proposed, in the thesis study this paper work is optimized and modified for double result. This structure is proposed during thesis study.

Power Function:

This function is combination of log(x) and exp(x). Since power function result is

used both in potential and force computations, it is optimized for generate $\propto^{\partial}$ and $\propto^{\partial-1}$ at the same time. This structure is proposed during thesis study.

Logarithm Function:

In fact, we do not need logarithm function Tesoff2 computation, to compute double precision power we need logarithm. This function is designed with respect to [7]. In this paper, pipelined, well suited architecture for logarithm computation is proposed.

Log adder Function:

Tersoff2 co-processor is fully pipelined digital circuit. In accumulation parts, we need very big adders to accumulate energy portions of the system. Unfortunately, adder trees are very big structures to design. As a solution for this problem given in thesis work, is a special asymmetric log adder tree. This structure is proposed during thesis study.

The complex functions due to Tersoff2 are;

Radius computation function:

This function is designed and implemented for computing distance between two atoms. With the help of well known Cartesian distance formula;

$$r_{ij} = \sqrt{\left(\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2\right)} \qquad (3.1)$$

It also includes of Cartesian computation and sqrt(x) and 1/sqrt(x) functions. This structure is proposed during thesis study.

Cache Structure for Tersoff2:

This library instance is the memory architecture designed for molecular dynamics. This part is designed for feeding, loading and serving elements for computation functions. It has the ability of feeding all computation pairs at the same time. This structure is proposed during thesis study.

Carbon Pair Generator Function:

This instance is designed for address decoding service for cache structure. It serves consecutive pairs for every cycle. This structure is proposed during thesis study.

Verlet Update Function:

Verlet computation is a very important part of position and velocity update of computed potential. It is fully pipelined. This structure is proposed during thesis study.

Thermostat Function:

Since the thermal stability of carbon nanotube system is investigated; a controller is designed and implemented in this study.

Intelligent Reset Function:

Since Tersoff2 potential function has two and three particle potential computation in every atomic computation stage there has to be a circuit which is going to clear the accumulation parts. This structure is proposed during thesis study.

Cosine Function:

In Tersoff2, there is a three particle potential computation, so we need to calculate the cosine of the angle between three atoms. It is a well known identity in cosine theory in a triangle form;

$$Cos(\emptyset) = \frac{({r_{ij}}^2 + {r_{ik}}^2 - {r_{jk}}^2)}{(2 \times r_{ij} \times r_{ik})} \qquad (3.2)$$

Projection Function:

In Tersoff2, we need to find Force vectors in x, y and z bases. These are called Fx, Fy, Fz in the design. To find these vectors, three division circuits have to be utilized. This solution is going to consume lots of resources. Instead of that solution we can use multiplication of reciprocal of square root result because, multiplication consumes much less resources. This structure is proposed during thesis study.

In the following part, each of the library instance and main part of processor is going to be explained.

3.2 MOLECULAR DYNAMICS SIMULATION LIBRARY AND TERSOFF2 CO-PROCESSOR ARCHITECTURE

In this part, elementary functions and architecture of Tersoff2 is going to be explained.

Following List of Symbols are used through this part.

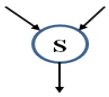
In general, this symbol indicates addition family. In each schematic, there will be a note to explain type of addition i.e. double float, fixed, signed, unsigned.

In general, this symbol indicates subtraction family. In each schematic, there will be a note to explain type of subtraction i.e. double float, fixed, signed, unsigned.

In general, this symbol indicates multiplication family. In each schematic, there will be a note to explain type of multiplication i.e. double float, fixed, signed, unsigned.

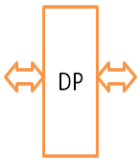
In general, this symbol indicates division family. In each schematic, there will be a note to explain type of division i.e. double float, fixed, signed, unsigned.

In general, this symbol indicates comparison family. In each schematic, there will be a note to explain type of comparison i.e. double float, fixed, signed, unsigned.

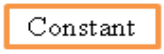


In general, this symbol indicates square root family. In each schematic, there will be a note to explain type of square root i.e. double float, fixed, signed, unsigned.

In general, this symbol indicates inverse square root family. In each schematic, there will be a note to explain type of inverse square root i.e. double float, fixed, signed, unsigned.



In general, this symbol indicates block ram family. In each schematic, there will be a note to explain type of block ram i.e. dual port, single port, prom with different port sizes.



In the design, there are lots of constants whose are used in the computations.

In general, this symbol indicates FIFO family. In each schematic, there will be a note to explain type of FIFO.

3.2.1ELEMENTARY MATHEMATICS LIBRARY DESIGN

3.2.1.a Exponential Function:

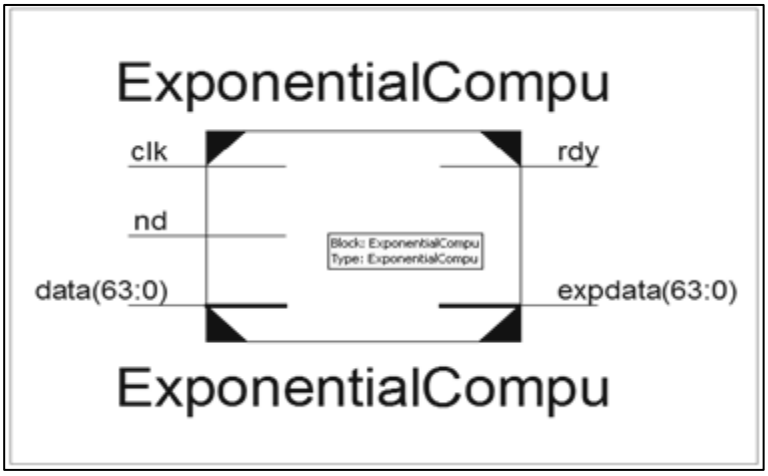


Figure 4: Black box input output diagram of exponential module.

To design and implement this library instance, Jamro and Wielgosz [5] architecture is used due to it's pipelined structure. First, there is a mathematical explanation of solution is given then the circuit schematics and the resources which are used explained.

In the computation of Tersoff2 double precision floating point exponential is used, so we need to implement it with these properties;

a- Result per clock cycle is 1,

b- Fully pipelined architecture,

c- Result within one iteration.

We start with the well known mathematical identities:

$$e^{x} = 2^{x.\log_2 e} = 2^{x_i}.e^{(x-x_i).\log_2 e} \quad (3.3)$$

$$e^{x+y} = e^{x}.e^{y} \quad (3.4)$$

x_i in 3.3 is the integer part of x and

$$x_i = x.\log_2 e \quad (3.5)$$

x_f is the fractional part of x.

$$x_f = x - x_i.\log_2 e^{-1} \quad (3.6)$$

This fractional part in 3.5 can be decomposed in to four parts for turning exponential computation into four multiplications.

i.e. $$x_f = x_m.x_d.x_l.x_t \quad (3.7)$$

where;

$$x_m = x_{f\ most\ \ significant\ \ \ 9\ bits\ [1:9]} \quad (3.8)$$

$$x_d = x_{f\ middle\ \ \ significant\ \ \ 9\ bits\ [10:18]} \quad (3.9)$$

$$x_l = x_{f\ least\ \ significant\ \ \ 9\ bits\ [19:27]} \quad (3.10)$$

$$x_t = x_{f\ rest\ \ of\ \ \ least\ \ significant\ \ \ [28:51]} \quad (3.11)$$

Then

$$e^{x} = 2^{x_i}.e^{x_m}.e^{x_d}.e^{x_l}.(1 + x_t) \quad (3.12)$$

Remember that

$$e^{x} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots,\ -\infty < x < \infty \ldots. \quad (3.13)$$

Taylor expansion series will be replaced for 3.10 and only first two terms of 3.12 are enough to have right precision. Then we need stored pre-computed results of e^{x_m}, e^{x_d}, e^{x_l} and three small multipliers to get result. This architecture is designed with Verilog HDL and implemented in ISE 14.2 for V7485T device. In the [5], their design consumes 71 dsp units and operates at 166 MHz at most, while in this thesis work it consumes 48 units and operates at 400 MHz.

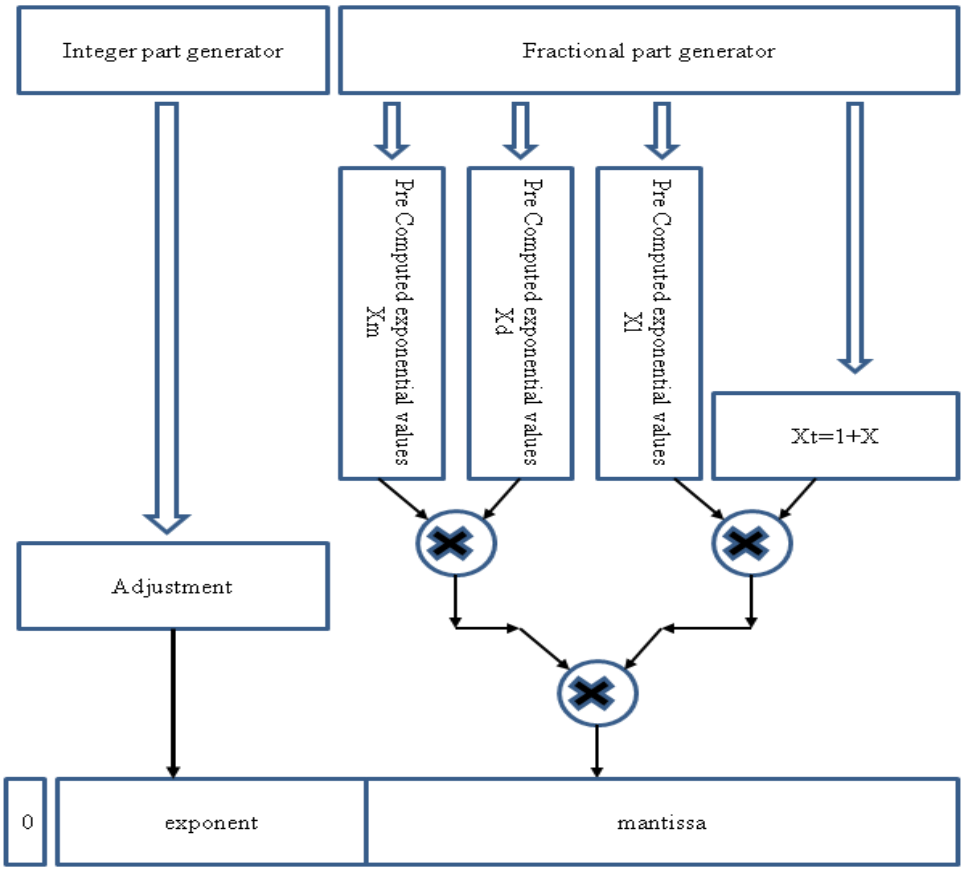


Figure 5: Schematic exponential computation diagram.

3.2.1.b Square Root Inverse Square Root Function:

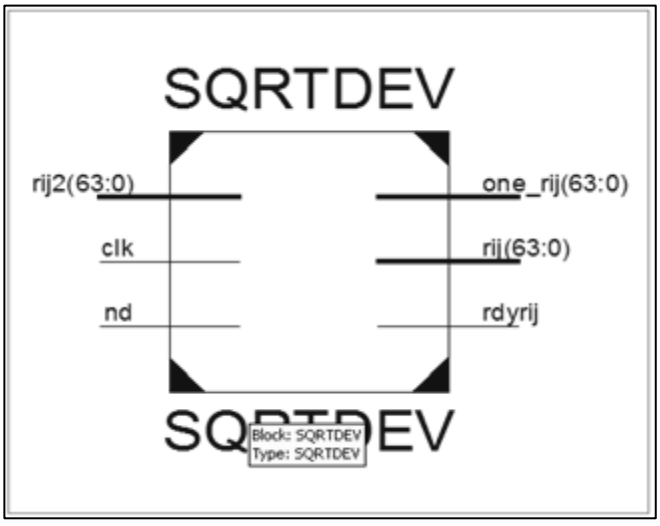


Figure 6: Black box input output diagram of square root and inverse square root module.

In Tersoff2 computation problem we need double precision floating point in two stages. The first one is atomic distance calculation and the second one is projection part. Since these two parts are consecutive parts a new computation method is proposed and designed in this thesis work "Generate both square and inverse square root result at the same time". The main idea of each result is based on Pineiro works [6]. Like the other instances, we need same properties with this computation: a-Result per clock cycle is 1, b-Fully pipelined architecture, c-Result within one iteration.

We begin with mathematical explanation; to find the solution in single iteration, force us to start a lucky iteration point. As explained in [6] this can be achieved by single precision initial guess. For single precision result of square root [6] utilize second order minimax approximation with Remez algorithm.
For single precision initial point, decompose fractional part into following order to find the pre-computed coefficient of approximation.

$$X_1 = [1.x_1x_2x_3 \dots x_{m1}] \quad (3.14)$$

$$X_2 = [.x_{m1+1}x_{m1+2}x_{m1+3} \dots x_{m2}] \times 2^{-m1} \quad (3.15)$$

$$X_3 = [.x_{m2+1}x_{m2+2}x_{m2+3} \dots x_{m3}] \times 2^{-m3} \quad (3.16)$$

m1 and m2 are the border bits of the separation.

When we sent these bits to coefficients lookup tables we can get the input vector set of first initial guess from Remez algorithm [6]. Then we can compute the following polynomial to reach single precision square root solution.

$$R_s = C_0 + C_1.X_2 + C_2.X_2^2 \quad (3.17)$$

Up to this point we use small multipliers and adders and lookup tables. Second computation stage of square root and inverse square root problem will be solved with Goldschmidt algorithm [8]. For floating point computations the mantissa part is in the range of [1, 2). The detailed analysis of method is out of this thesis work.

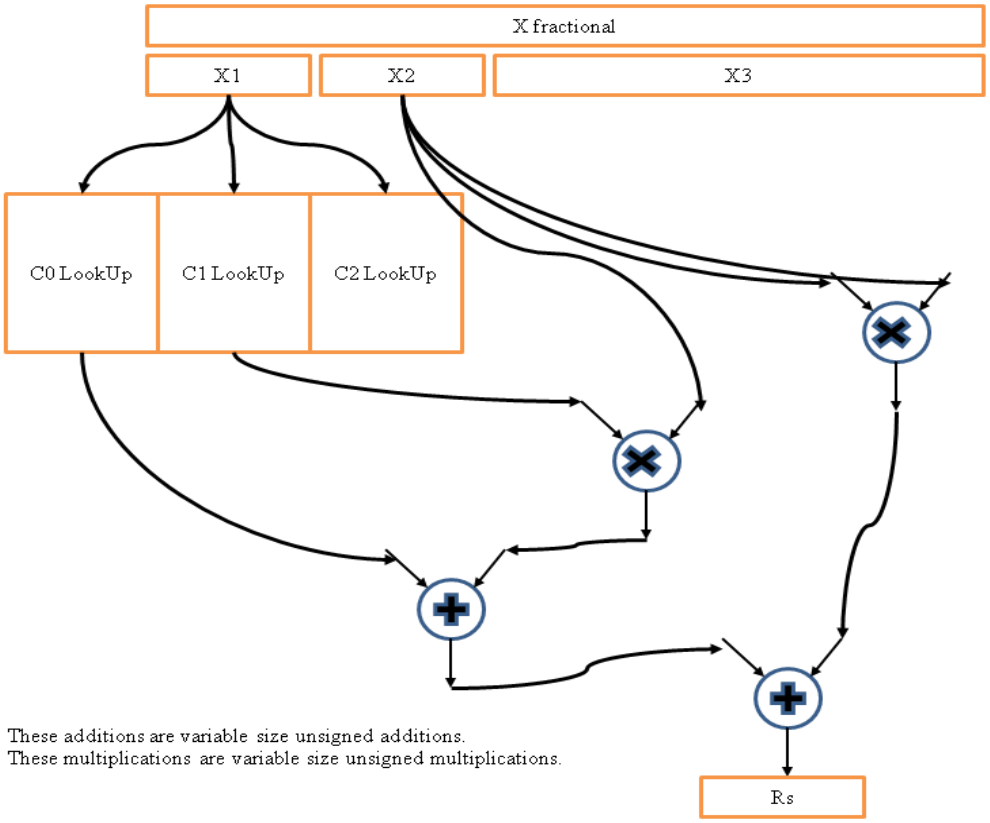


Figure 7: Abstract initial single precision Rs computation diagram.

The computation of Goldschmidt procedure for square root and inverse square root are:

- First compute the Goldschmidt coefficient for square root and inverse square root.

$$G_{s\ square} = X.R_s \tag{3.18}$$

$$G_{s\ inverse\ \ square} = R_s \tag{3.19}$$

- Second find the bound of solution to guarantee the result.

$$V_s = 1 - R_s.G_s \tag{3.20}$$

- Finally, compute the following equation to get the final result.

$$Z = G_s.\left(1 + \frac{V_s}{2}\right) \tag{3.21}$$

In the [6] proposed architecture does not compute both function at the same time. It utilizes same resources for each function. In this thesis work this structure is modified to get both results at the same time.

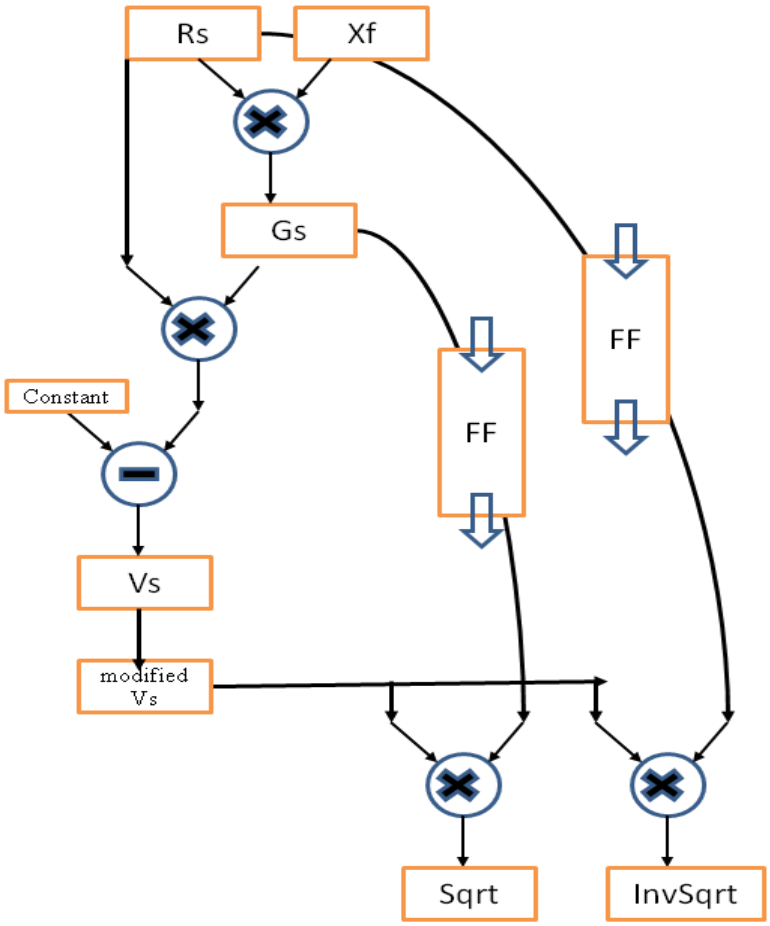


Figure 8: Starting with single precision Rs and modified Goldschmidt structure the both result can be obtained at the same time.

Starting coefficient for inverse square root is R_s. Replacement G_s with R_s in the Final step will give us the inverse square root solution. To handle both equations at the same time we have to use parallel resources.

$$Z_{sqrt} = G_s.\left(1 + \frac{V_s}{2}\right) \quad (3.22)$$

$$Z_{inverse\ \ sqrt} = R_s.\left(1 + \frac{V_s}{2}\right) \quad (3.23)$$

By using small number of resources, in this work we make improvement on result. This architecture is designed with Verilog HDL and implemented in ISE 14.2 for V7485T device. In this thesis work it consumes 61 DSP units and operates at 400 MHz.

3.2.1.c Power Function:

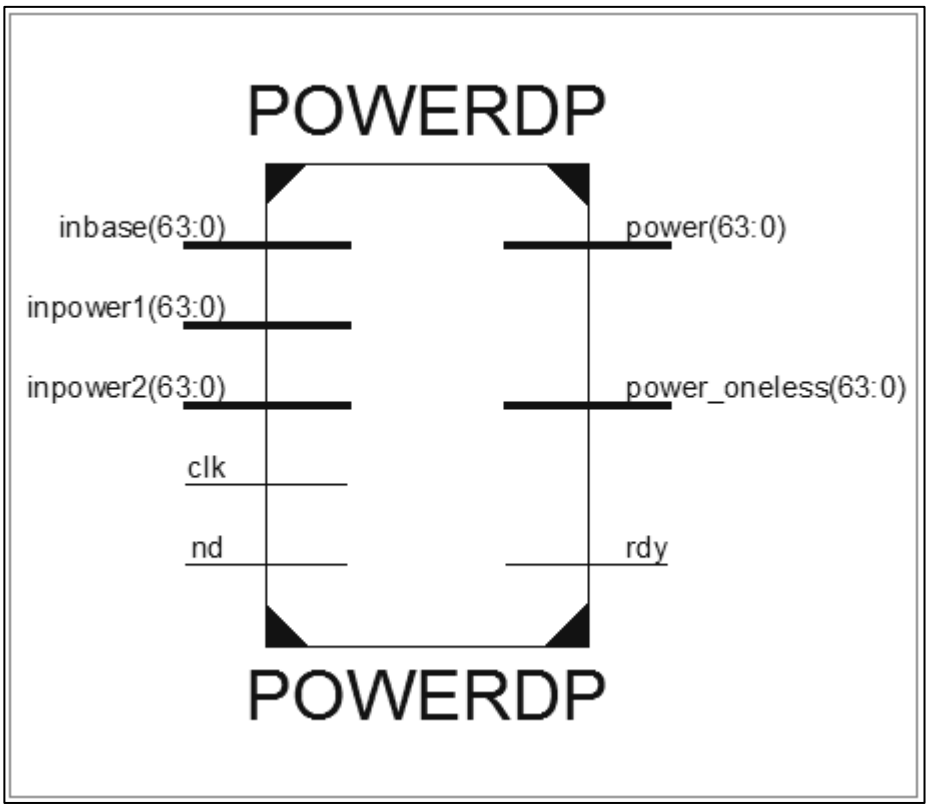


Figure 9: Double precision floating point black box diagram.

In Tersoff2 potential, we have double precision power computations. For α^{β} both of base α and power β are double precision floating points. There is not a direct computation in floating point for power, we use following well known mathematical identity to compute power function.

$$\alpha^{\beta} = e^{\log_e \alpha^{\beta}} \tag{3.24}$$

Then, this identity is equal to;

$$\alpha^{\beta} = e^{\beta . \log_e \alpha} \tag{3.25}$$

Since we have exponential and logarithm functions separately, we can compute power function with following procedure;

- First calculate the natural logarithm of base α,
- Then, do floating point multiplication with power β,
- Finally, compute the exponential of result.

With this structure we can compute double precision floating point power function.

In Tersoff2, we have to compute derivative of potential. We have one more power computation for derivative. Since derivative for power computations are one less power of same base, we can integrate both power computation to reduce resource consumption in the following way;

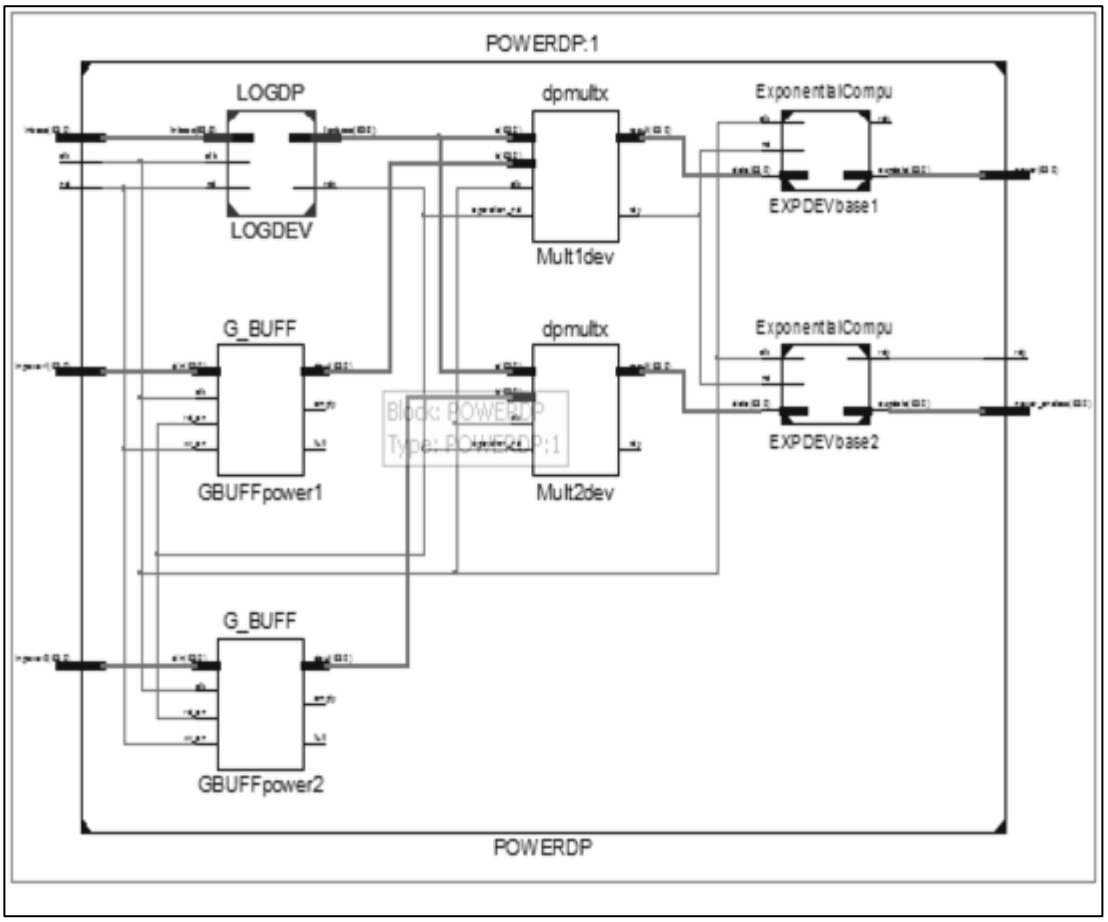


Figure 10: We can reduce power computation resources by eliminating the second natural logarithm module.

We need to compute both of α^{β}, $\alpha^{\beta-1}$. Our power computation strategy starts with taking natural logarithm of base α. Since, starting point of power computation of both terms are same. Then we don't need to us one more logarithm function. With this strategy we can reduce the resource consumption % 25.

3.2.1.d Log adder Function:

In floating point computations, accumulation takes lots of resources. Assume we need to take 8 pairs of terms, to do these additions we need to construct following circuit;

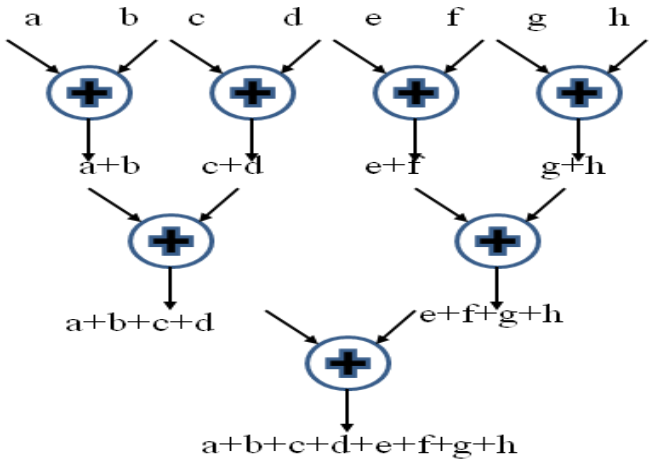


Figure 11: Accumulation of 8 terms with 7 adders in 3 stage addition.

This technique will consume too much resource in FPGA. Instead of this we can utilize log adder technique to reduce resource consumption.

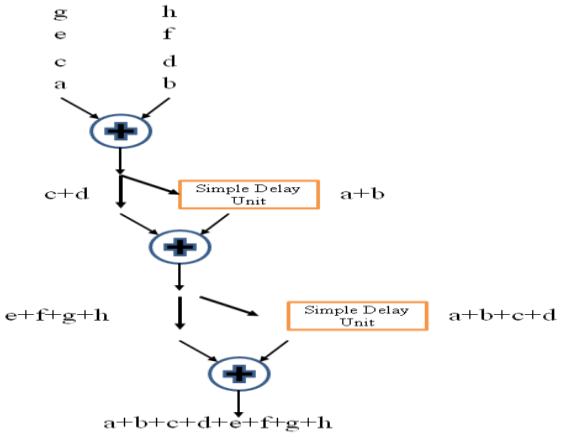


Figure 12: Accumulation of 8 terms with 3 adders and 2 simple delay units.

3.2.1.e Logarithm Function:

For double precision floating point logarithm function, this library follows [7] methodology. It is based on Taylor approximation. The procedure for computation is;

1-Reduction; deduce Y in n bits to A for producing simpler design where; $-2^{-k} < A < 2^{k}$ and k is 14 bit for double precision. For a given f(A) the function logarithm is computed.

2-Evaluation; compute approximate value of B from given approximate function.

3-Postprocessing; since we reduce Y to A, we have to update the result B which is generated in step 2.

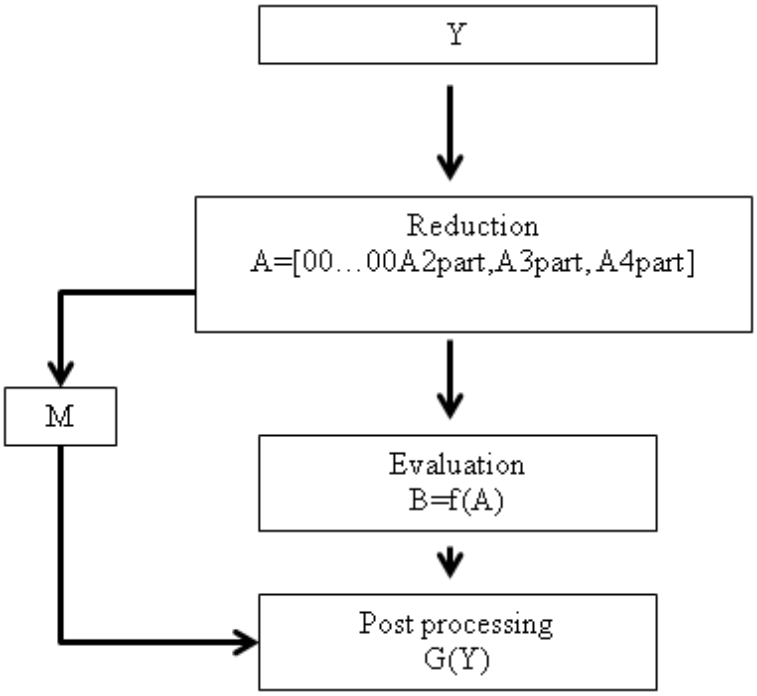


Figure 13: Approximation architecture of logarithm function.

Where ;

$$A = YxR - 1 \qquad (3.26)$$

R in 3.26 is 15 bits approximation of 1/Y.

In Evaluation step, we use following approximation;

$$\ln(1 + A) \cong A - \frac{1}{2}A_2^2 z^4 - A_2 A_3 z^5 + \frac{1}{3}A_2^3 z^6 \qquad (3.27)$$

Where

$$z = 2^{-14} \qquad (3.28)$$

The design of 3.27 is consist of small multipliers and a big adder.

For post processing step;

$$G(Y) = B - \ln R \qquad (3.29)$$

We consume a large prom for pre-computed –ln(R) in 3.29 and small multipliers for computing logarithm function.

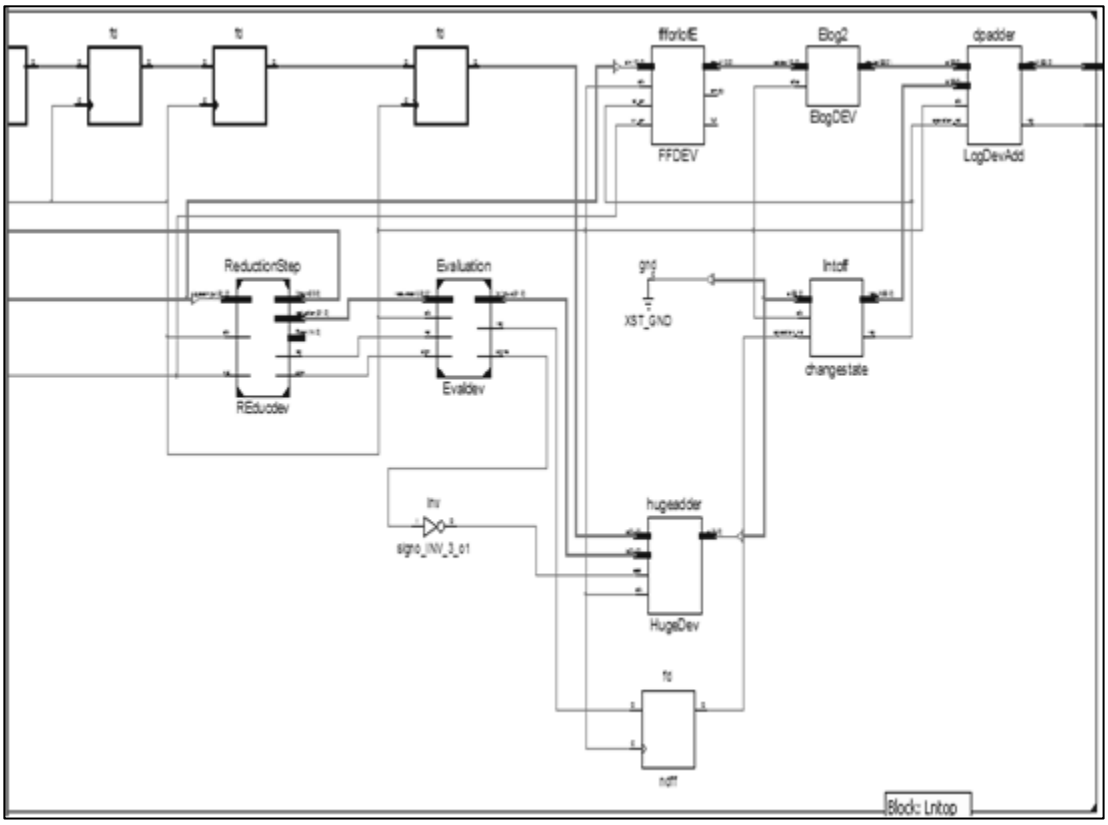


Figure 14: Schematic diagram of logarithm function.

The design consumes 8 X 36 K bits block rams and 60 DSP units and can operate up to 400MHz.

3.2.2 COMPLEX MATHEMATICS LIBRARY DESIGN

3.2.2.a Radius Computation Function:

Tersoff2 starts with computing the Radius computation function. In Cartesian coordinates system; the distance can be computed from well known identity;

$$r_{ij} = \sqrt{\left(\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2 + \left(z_i - z_j\right)^2\right)} \qquad (3.30)$$

The strategy to compute distance r_{ij};

- First find the distance between each bases i.e. x_{ij} , y_{ij} , z_{ij}

- Then find the square of them,
- Add the three terms,
- Finally send the resulting value to square root and inverse square root module.

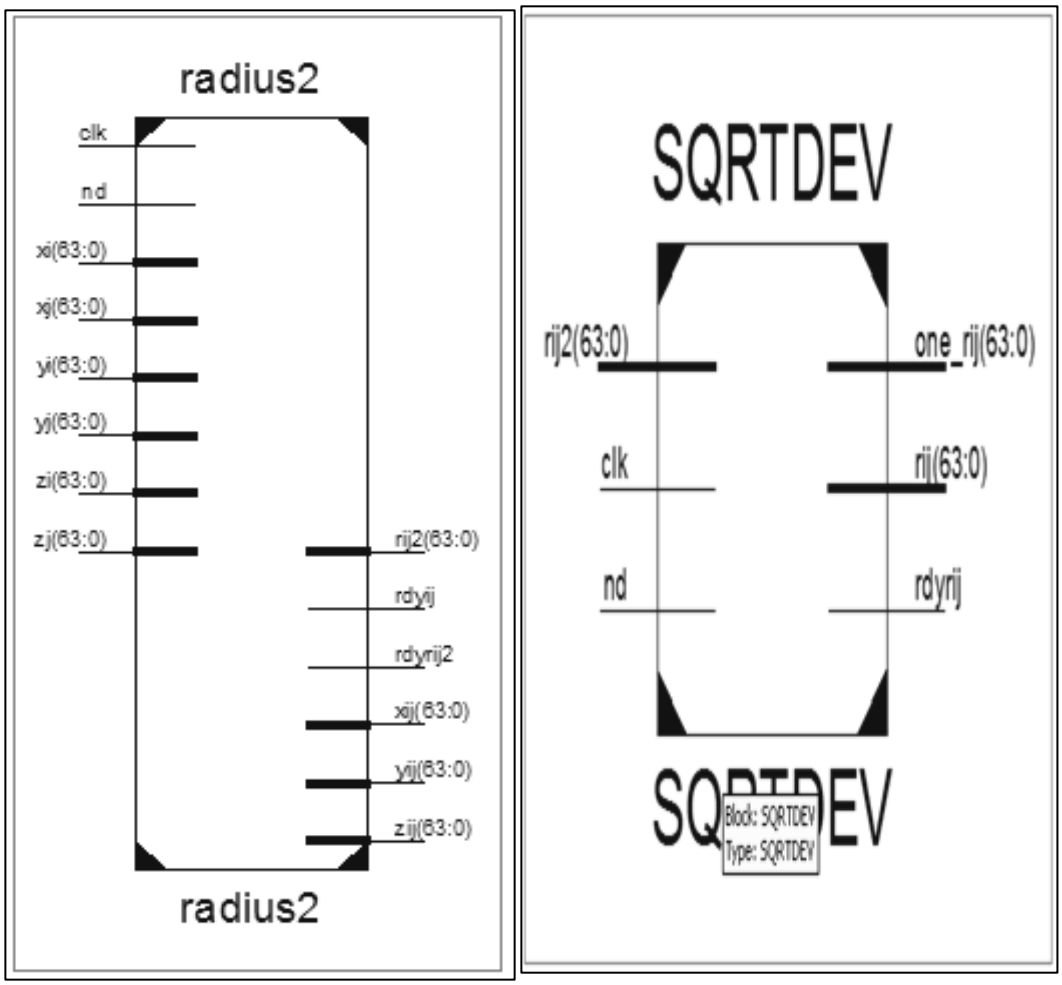


Figure 15: The radius square black box diagram with square root black box diagram.

In this design in every clock cycle we can feed all terms of Cartesian coordinates of three neighbor atoms. After a determined latency, distance square feed through SQRTDEV module and we can find both distance and reciprocal distance of three pair atomic system.

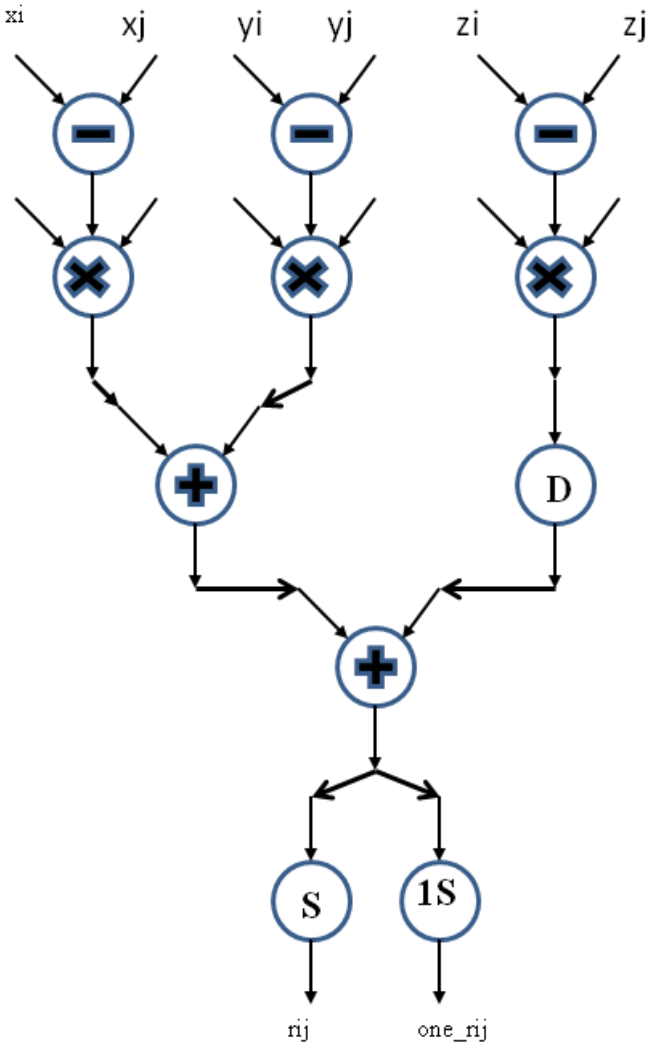


Figure 16: Schematic for computing distance and inverse distance.

In this design, all of the computation elements are double precision floating points. The symbol D indicates the synchronization elements. In the first stage of computation we get the distances between for two atoms in the x, y, z bases;

$$x_{ij} = x_i - x_j \quad (3.31)$$

$$y_{ij} = y_i - y_j \quad (3.32)$$

$$z_{ij} = z_i - z_j \quad (3.33)$$

In the second stage we multiple terms 3.31, 3.32, 3.33 with themselves to get square of terms.

$$x_{ij}^2 = x_{ij} \times x_{ij} \quad (3.34)$$

$$y_{ij}^2 = y_{ij} \times y_{ij} \quad (3.35)$$

$$z_{ij}^2 = z_{ij} \times z_{ij} \quad (3.36)$$

In third and fourth stage we add 3.34, 3.35 and 3.36.

$$r_{ij}^2 = x_{ij}^2 + y_{ij}^2 + z_{ij}^2 \quad (3.37)$$

In the final stage we take the square root and the inverse square root to get distance and inverse distance.

3.2.2.b Cache Structure for Tersoff2:

In Tersoff2, to get high performance result special memory structure for computation kernels has to be designed. Having fully pipelined digital circuits does not mean we get the best performance at the end, without an optimized memory cache structure. So we need to classify how to design and locate the single cycle reachable memory elements.

For calculating distances we need to reach $x_i, x_j, x_k, y_i, y_j, y_k, z_i, z_j, z_k$ at the same cycle, moreover we need $v_{xi}, v_{yi}, v_{zi}, F_{xi}, F_{yi}, F_{zi}$ for Verlet update module to compute new velocity and force values. By definition each of these terms are 64 bits. Following memory design is going answer all of constraints that we need. First, by using three dual-port block rams we construct the basic memory element instances.

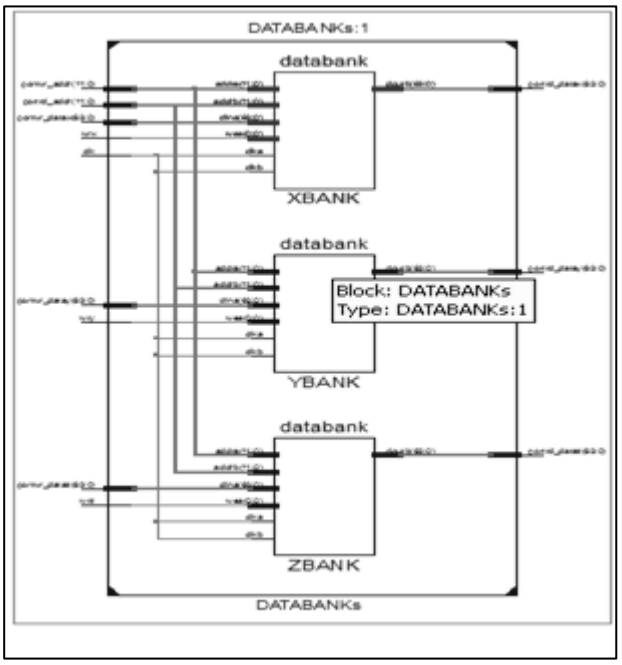


Figure 17: Schematic for basic memory for xi, xj and xk.

Second, use three modules of basic memory elements to construct single coordinate data bank. If we connect to same address information to all of address input port of databanks we can get single cycle 9 X 64 bits memory cache.

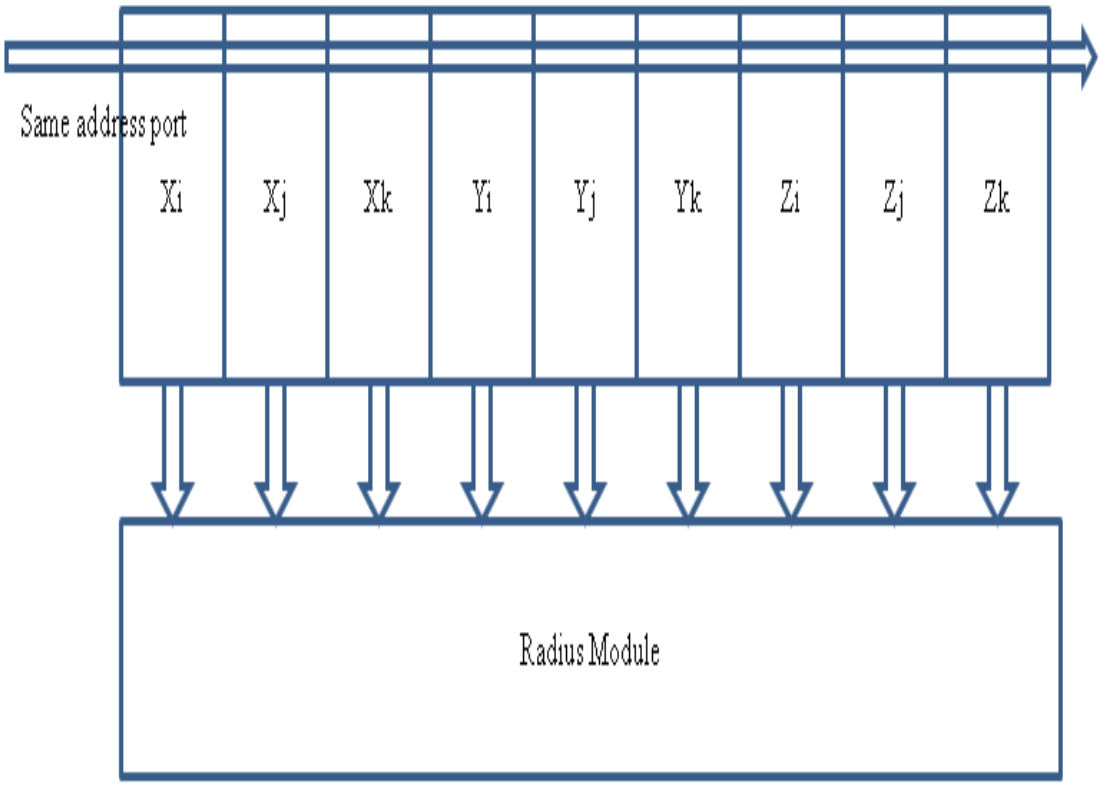


Figure 18: Black Box Diagram for coordinate's memory structure.

In each computation cycle, with this architecture we can feed all of the Cartesian terms to radius module and best performance going to be achieved.

Moreover, in this design we use neighbor list concept, we also need some memory blocks to store neighbor information for each atom. By using neighbor list method we do not need to calculate every atom cutoff range. We are going to use Neighbor list for each atom to reach the neighbors. After gathering all of the memory structures, following cache architecture is designed.

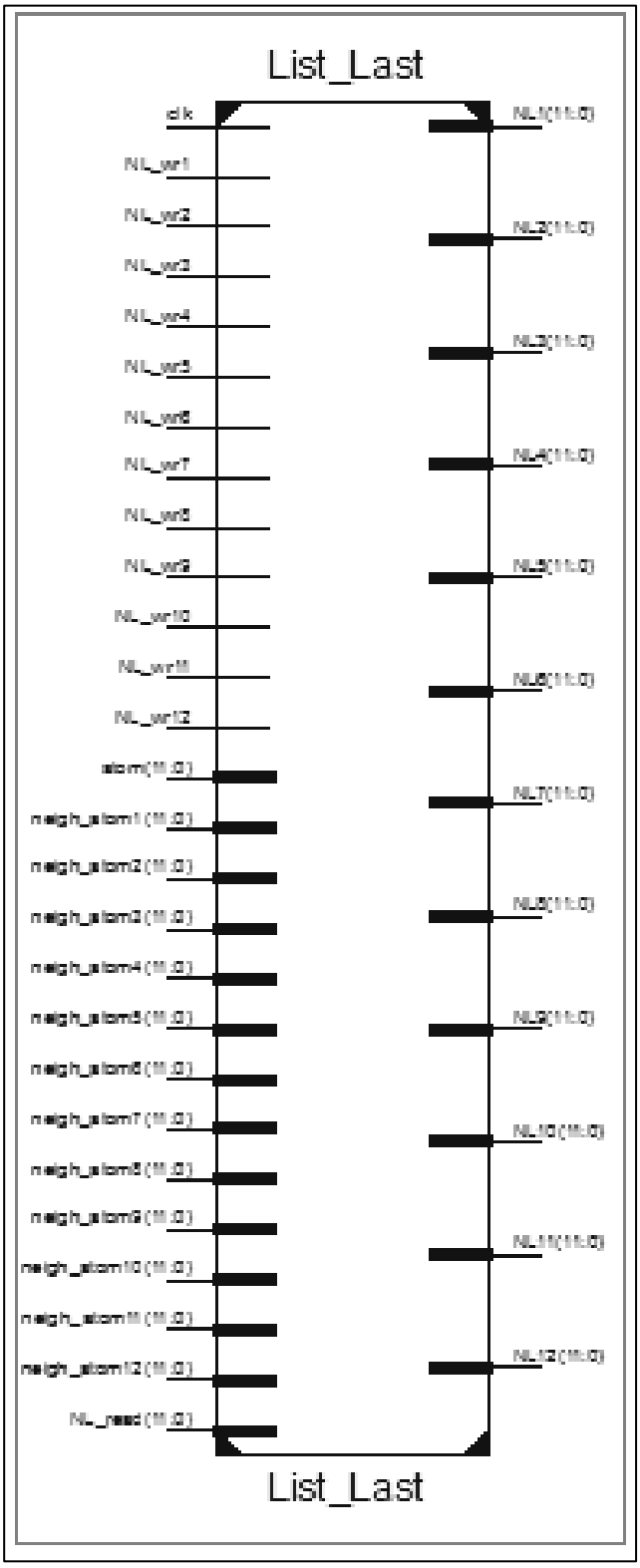


Figure 19: Black Box Diagram for neighbor list module.

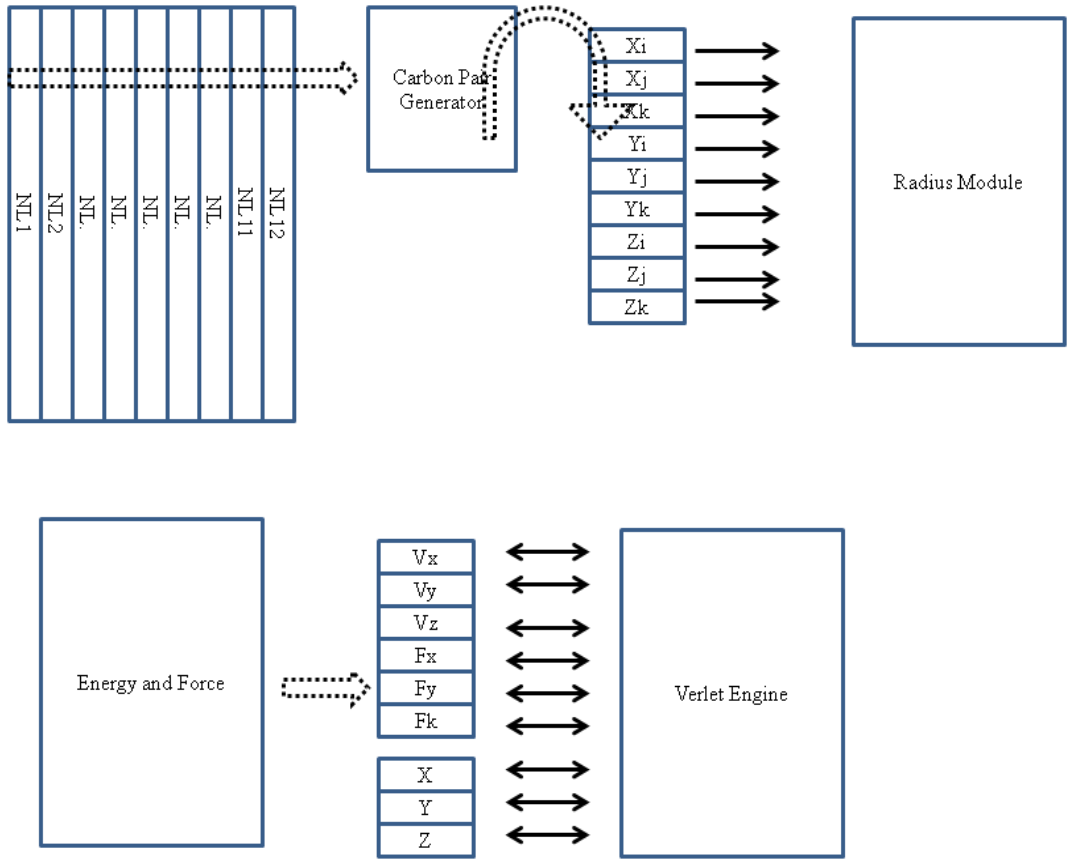


Figure 20: Black Box Diagram for all of the memory blocks.

With this architecture, following memory stream strategy followed.

First, after selecting current atom neighbor list of it, send them to Carbon pair generator. Second, after generating related atom address in the carbon pair generator module, appropriate addresses of three neighbor atom selects the memory location and sends the Cartesian coordinates to radius (distance) module. Third, after calculating the energy and force results, new values of velocity and forces with old ones send to Verlet update module. Finally, new computed values transferred to appropriate locations.

Cache structure of tersoff2 consumes; Neighbor List 18 block-rams (18 K bits), Databanks 3 X 23 block-rams (36 K bits). With consumption of these resources we can reach best performance due to single cycle memory feed system.

3.2.2.c Carbon Pair Generator Function:

In the selection atoms, there is a need to having intelligent pair selection system. In this processor design we follow the design rule of pipelining. Since all other parts of processor are pipelined, to have best performance we need pairs at every cycle otherwise, pipelined design is going to be useless.

For this constraint, this design includes a carbon pair generator system.

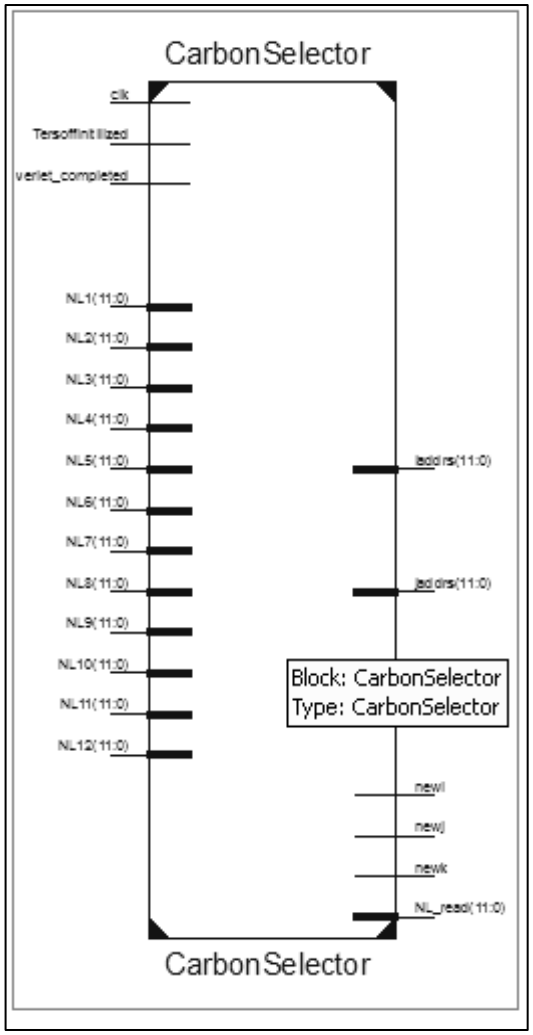


Figure 21: Black Box Diagram for carbon pair generator.

The module gets the Neighbor List and generates appropriate address in single cycle with a determined latency. Module consumes 400 registers to accomplish this constraint.

3.2.2.d Verlet Update Function:

After calculating the energy and force of each atom, we need to update position and velocity of each atom in the system.

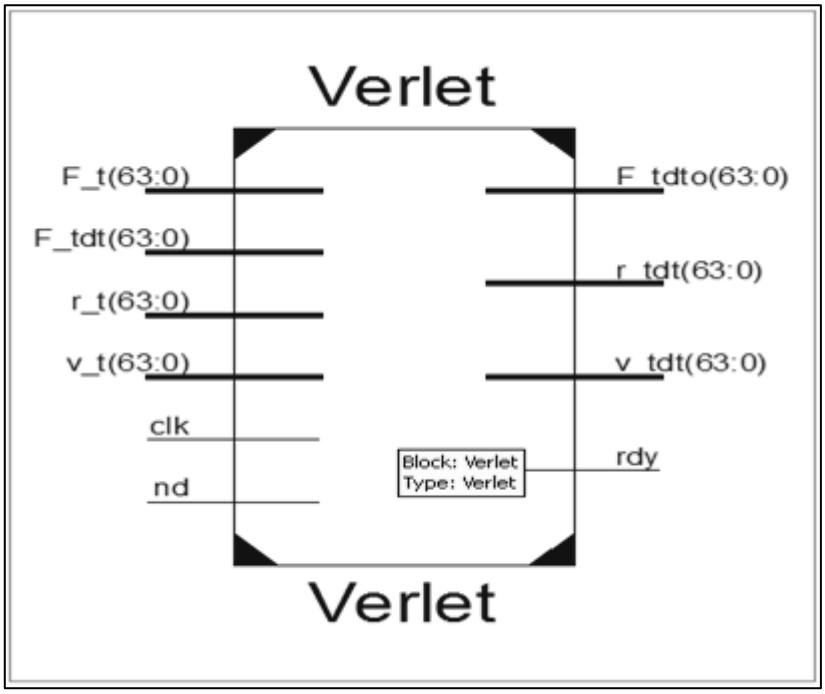


Figure 22: Black Box Diagram for Verlet Update.

New position is equal to;

$$r_{n+1} = r_n + h \times v_n + (1/2 \times m) \times h^2 \times F_n \qquad (3.38)$$

New velocity is equal to;

$$v_{n+1} = v_n + \left(\frac{h}{2m}\right) \times (F_{n+1} + F_n) \qquad (3.39)$$

Where;

r_{n+1} is the next MD step location of the atom,

r_n is the current location of the atom,

h is the time constant, m is the mass constant,

v_n is the current MD step velocity,

v_{n+1} is the next MD step velocity,

F_n is the current MD step force,

F_{n+1} is the next MD step force.

These equations are computed for all of three Cartesian axes.

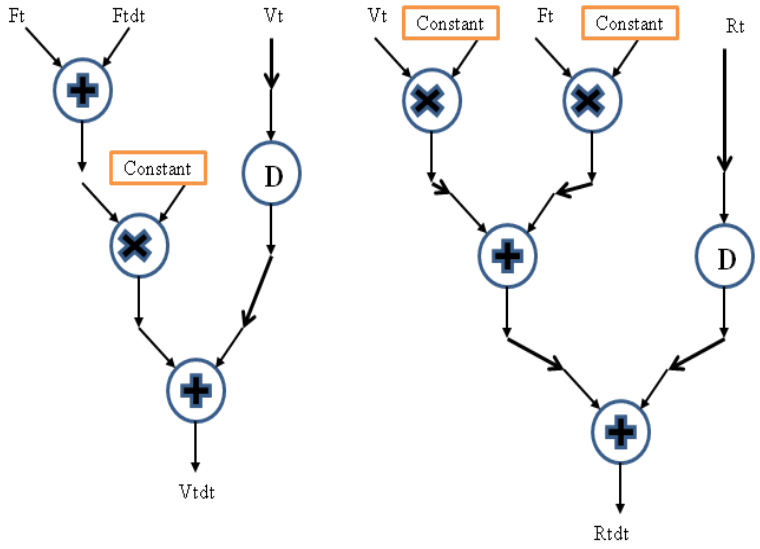


Figure 23: Schematic diagram of main Verlet computation for one Cartesian axe.

Since we need this circuit for all axes, resulting structure is in Figure 24.

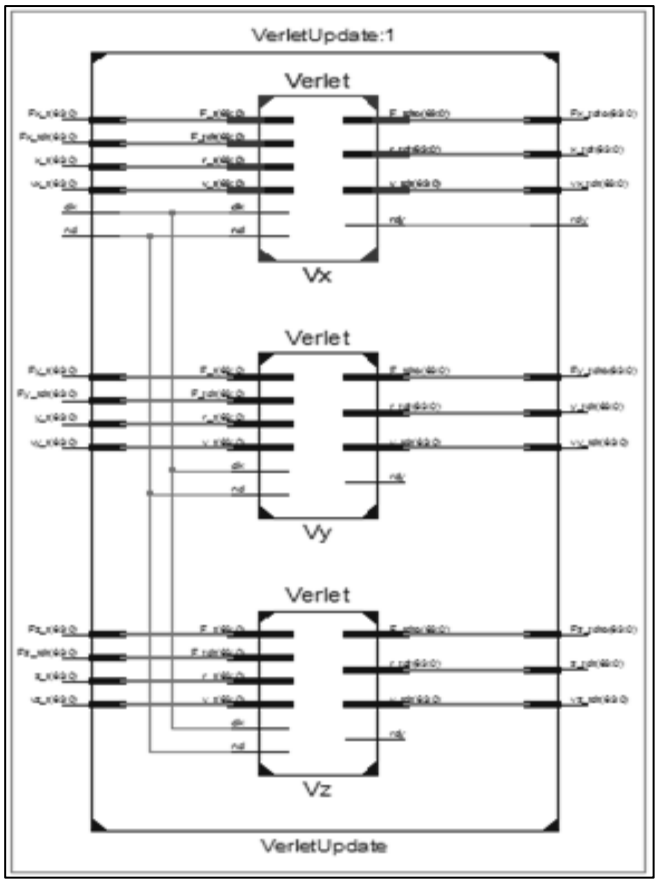


Figure 24: Black box diagram of three axis Verlet update.

This circuit generates velocity and position updates at every clock cycle with fully pipelined mode. Design consumes 117 DSP bocks. Each computation symbol represents a floating point unit and can operate up to 500 MHz.

3.2.2.e Thermostat Function:

In tersoff2 computation, there is a thermal control of the system. After completing a MD step, system's kinetic energy is calculated and according to following equations its temperature is found. In every 2 MD step according to temperature of the system, a velocity scaling factor is calculated and velocities of the atoms are recomputed with this scaling factor. This process ensures the thermal stability of the system.

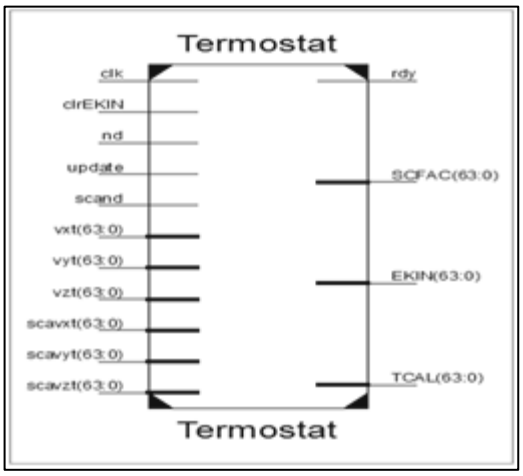


Figure 25: Black box diagram of Thermostat.

$$Temperature = \frac{RM \times pKinetic \quad energy}{3 \times NA \times BK} \qquad (3.40)$$

RM, NA and BK are constants.

$$ScalingFactor = \sqrt{\frac{TE}{Temperature}} \qquad (3.41)$$

TE is the target temperature has to be scaled.

In Thermostat Function this strategy is followed;

First calculate the system kinetic energy by accumulating kinetic energy of each atom

$$V_T = v_x^2 + v_y^2 + v_z^2 \qquad (3.42)$$

$$pKineticEnergy = \sum V \qquad (3.43)$$

$$KineticEnergy = RM\ x\ pKineticEnergy \qquad (3.44)$$

Then find the temperature of the system. Finally, calculate the Scaling Factor and rescale the velocity with this factor.

In this thesis work, thermostat part is the most problematic part of all design. Because, there is a division and square root for only one term. This processor uses the power of pipelining and parallelism. For one operation like Temperature calculation or scaling factor calculation, these properties are not going to be used. At the end, all system has to wait the scaling factor computation. This result is inevitable. This design consumes 100 DSP units. It can operate up to 400 MHz.

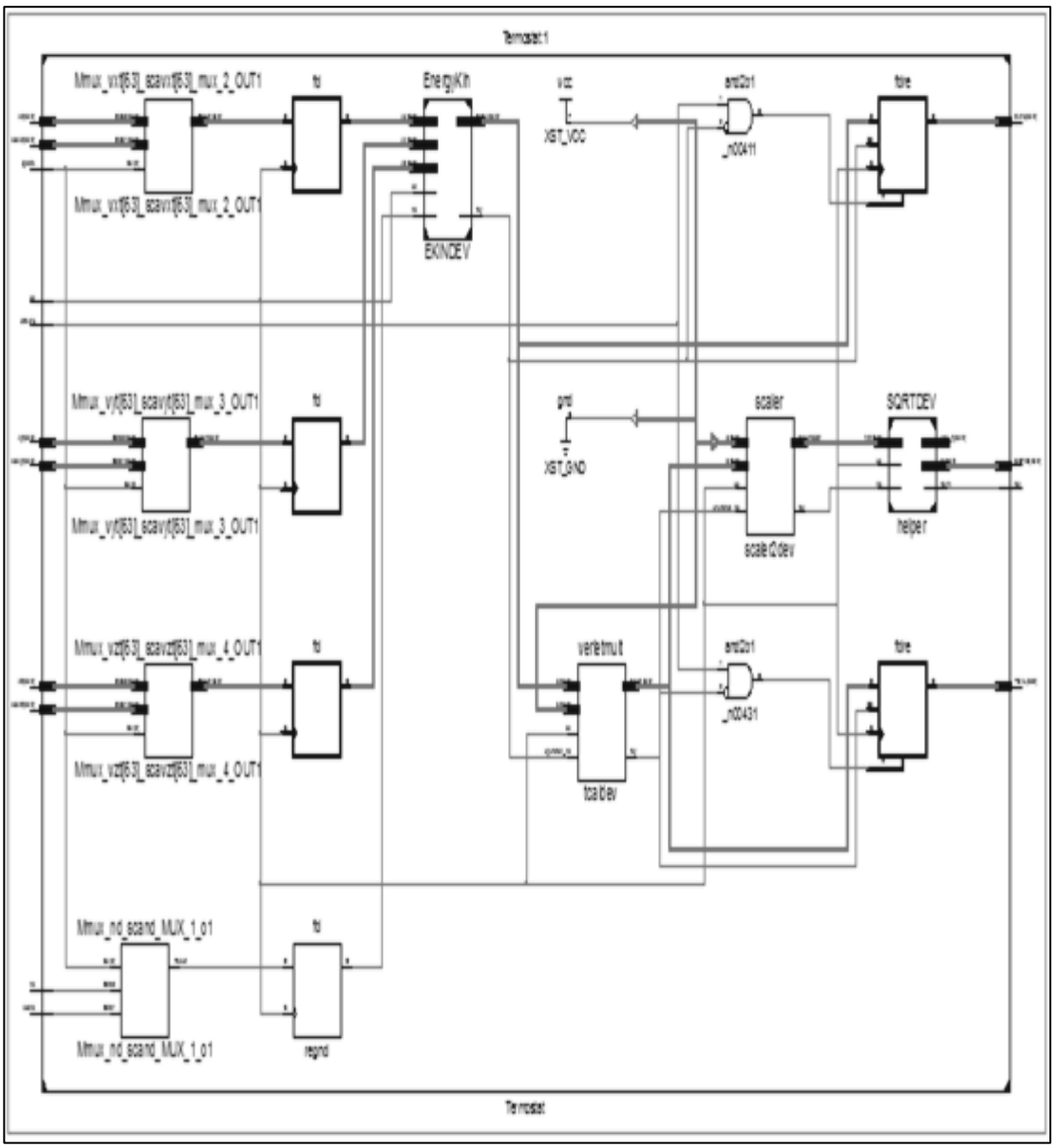


Figure 26: Schematic diagram for Thermostat.

3.2.2.f Intelligent Reset Function:

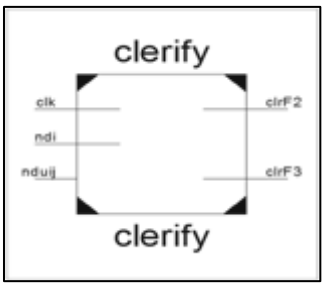


Figure 27: The black box representation of intelligent reset circuit.

This part of the design is a special reset circuit for Accumulators. Since system generates forces consecutively, we have to clear accumulation modules for consistency. It utilizes simple shift registers and generates reset flags through all processor when it is needed. It consumes only 30 registers and can operate up to 700 MHz.

3.2.2.g Cosine Function:

In thesoff2, we have to find the cosine of the angle between three neighbor atoms. We can utilize well known mathematical identity;

$$Cos(\emptyset) = \frac{({r_{ij}}^2 + {r_{ik}}^2 - {r_{jk}}^2)}{(2 \times r_{ij} \times r_{ik})} \qquad (3.45)$$

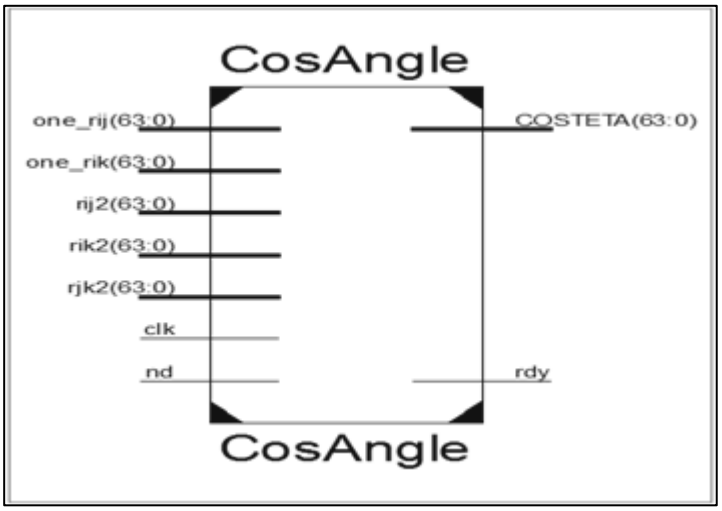


Figure 28: the black box diagram of cosine angle module.

There are two divisions in this calculation if we rearrange this equation we can get a more efficient circuit for digital design.

$$\cos \emptyset = \left(r_{ij}{}^2 + r_{ik}{}^2 - r_{jk}{}^2\right) x \frac{1}{2x\, r_{ij}} x \frac{1}{r_{ik}} \qquad (3.46)$$

In the square root and inverse square root module, we have already found $\frac{1}{r_{ik}}$ and $\frac{1}{r_{ij}}$.

As a result, we don't have to use division which will consume lots of system resources. Note that, we don't divide by 2 with division; instead of this we subtract 1 from the exponent part of floating point.

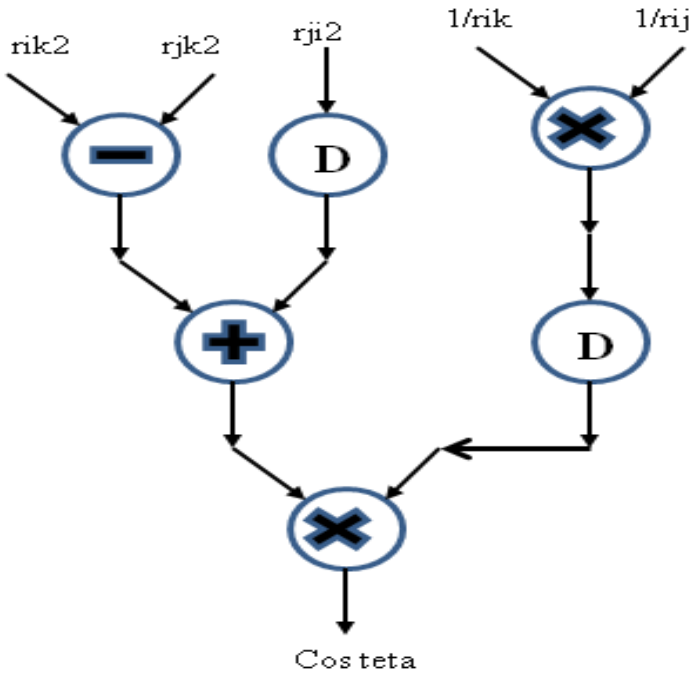


Figure 29: Abstract diagram of cosine angle module.

To calculate the Cosine of the Angle, design uses following strategy;

First, buffer $r_{ij}{}^2$ and calculate

$$r_{ik}{}^2 - r_{jk}{}^2 \qquad (3.47)$$

At the same time do the multiplication $\frac{1}{r_{ij}} x \frac{1}{r_{ik}}$ and buffer the result.

Second, do addition

$$r_{ij}{}^2 + r_{ik}{}^2 - r_{jk}{}^2 \qquad (3.48)$$

Finally, after finishing addition, multiply first and second stage outputs with deducting exponent of the result.

This module consumes 24 DSP units and can operate 330 MHz. It is fully pipelined.

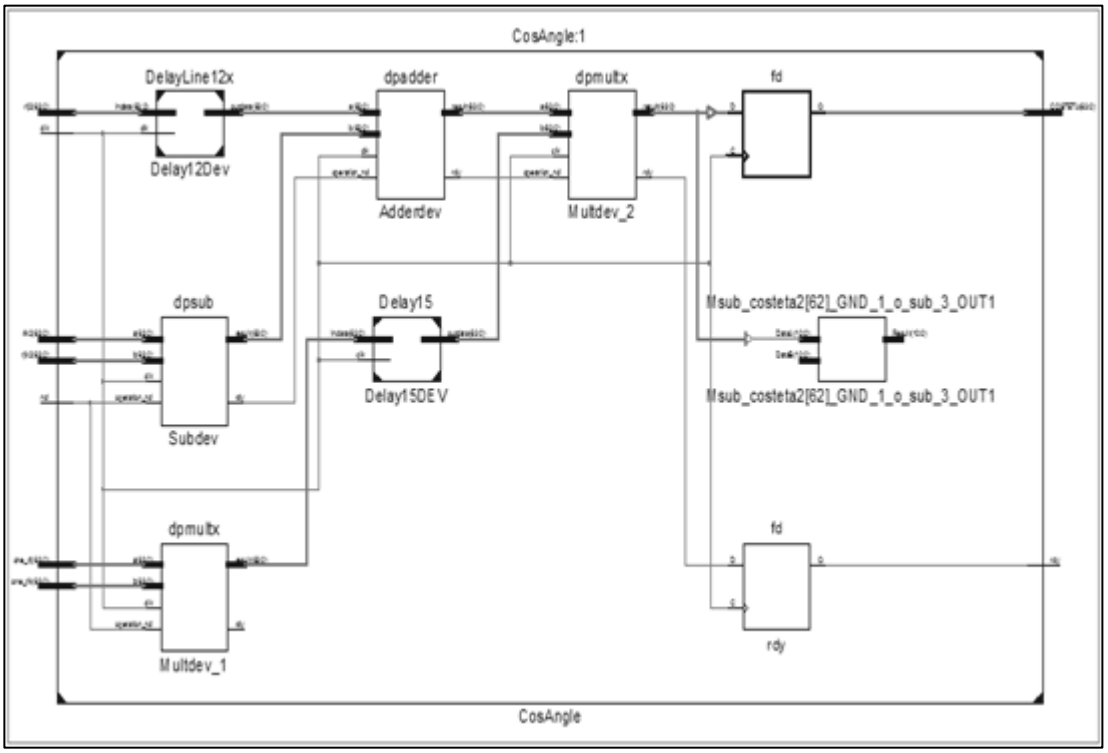


Figure 30: Schematic diagram of cosine angle module.

3.2.2.h Projection Function:

We have to find projection of force onto Cartesian system in all MD calculations. In theory we have to use 3 division modules to get the right result. The projection can be done by following equations;

$$XR_{ij} = \frac{X_{ij}}{R_{ij}}, YR_{ij} = \frac{Y_{ij}}{R_{ij}}, ZR_{ij} = \frac{Z_{ij}}{R_{ij}} \quad (3.49)$$

However, we generate inverse square root of R_{ij} when we calculate the distance between two atoms. So, in the design multiplication is going to be used instead of divisions in the following manner.

$$XR_{ij} = X_{ij}\, x \frac{1}{R_{ij}}, YR_{ij} = Y_{ij}\, x \frac{1}{R_{ij}}, ZR_{ij} = Z_{ij}\, x \frac{1}{R_{ij}} \quad (3.50)$$

Since there are two locations where projection exists, in the processor design we use 6 multiplications instead of 6 divisions to get great benefit by means of resource usage.

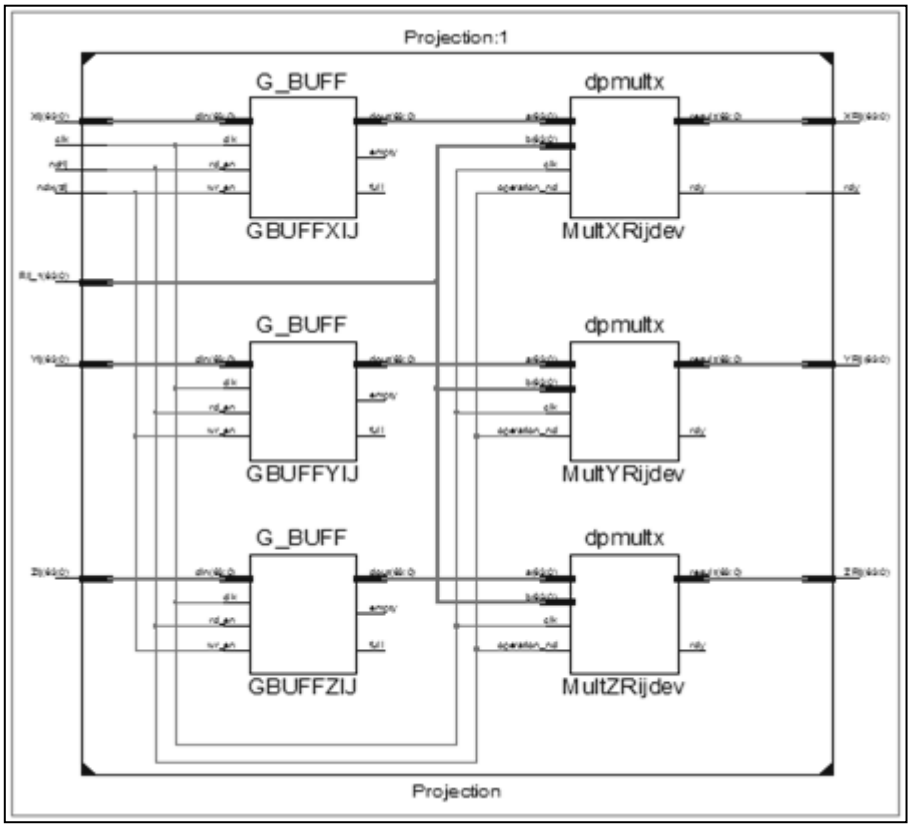


Figure 31: Schematic diagram of Projection module.

3.2.2.i Handshaking Strategy:

All of the modules have handshaking ability to save time consistency. This ability is location free. Each module has **nd** new data input flag and **rdy** result ready flag. A new input supplied to any module has also informed with **nd** input. Also, it generates **rdy** signal when its result is ready.

3.2.2.j Synchronization strategy:

In this thesis work, designed modules have different latency, in other words different functions generate results in different times. To be consistent in calculations, we have to synchronize the result of modules and buffer the result until when it is needed for another module. Moreover, some results are needed by different time and different modules. So, sometimes we need to replicate results in different locations.

Two elements of Synchronization:

1. Shift registers: For time less than 32 cycles we can utilize shift registers to synchronization. We can use these elements to delay in pipelined fashion.
2. FIFOs: For time bigger than 32 cycles we need to buffer results in a FIFO until it is ready to serve in other equations.

CHAPTER 4

TERSOFF2 HARDWARE ACCELERATION KERNEL

For the beginning, we give a hierarchical design structure of main computation kernel. This kernel is the main computation part of Tersoff2 processor and it includes all of the computation that we need for energy and force calculations.

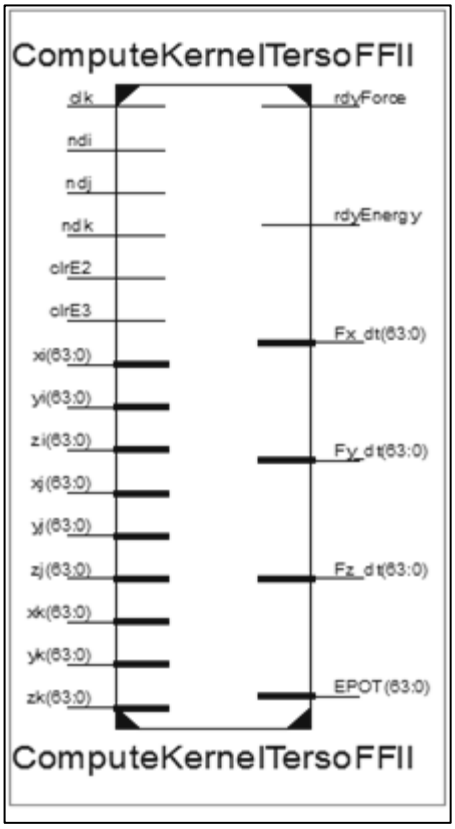


Figure 32: the black box representation of Kernel of Tersoff2

Tersoff2 kernel has Cartesian coordinates input ports and Force and Energy output ports this structure is fed by coordinate module for single cycle operation. It generates force components and potential components at every process cycle at the same time. We introduce kernel structure with lower level modules.

Kernel has the following hierarchical architectural structure;

- Tersoff2KernelDevice
    - Clarify-Device

- Coordinate-Device
  - Radius-Dev
  - SQRT-Device
  - Global Buffer Device
- Energy- Pair-Device
  - FCRIJ-Device
  - EXP-Device
  - UIJK-Device
  - DERIVATIVEFORCE-Device
  - Global Buffer Device
- GTETA-Device
  - FLEVEL-Device
  - SLEVEL-Device
  - TLEVEL1-Device
  - TLEVEL2-Device
  - Global Buffer Device
- ProjectionIJ-Device
- ProjectionIK-Device
- TwoPairsSummation-Device
  - Accumulation-Device
  - FX2-Device
  - FY2-Device
  - FZ2-device
  - Global Buffer Device
- ThreePairsSummation-Device
  - FCRIK-device
  - WIJKm-Device
  - DWIJm-Device
  - DWIKm-Device
  - XDWIJK-Device
  - YDWIJK-Device

        - ZDWIJK-Device
    - PowerSystem
        - POWER-Device
        - PARAN-Device
    - ForceThreePairs-Device
    - ForceM-Device
        - FORCEX-Device
        - FORCEY-Device
        - FORCEZ-Device
    - EnergyM-Device

4.1KERNEL.Clerify:

This module is explained in the library.

4.2KERNEL.Coordinate:

This module consists of radius module and square root and inverse square root module. Fundamental duty of this module is supplying related distance vectors to other modules. It generates rij, one_rij, rik, one_rik, rij2, rjk2, rik2, xij, yij, zij, xik, yik, zik.

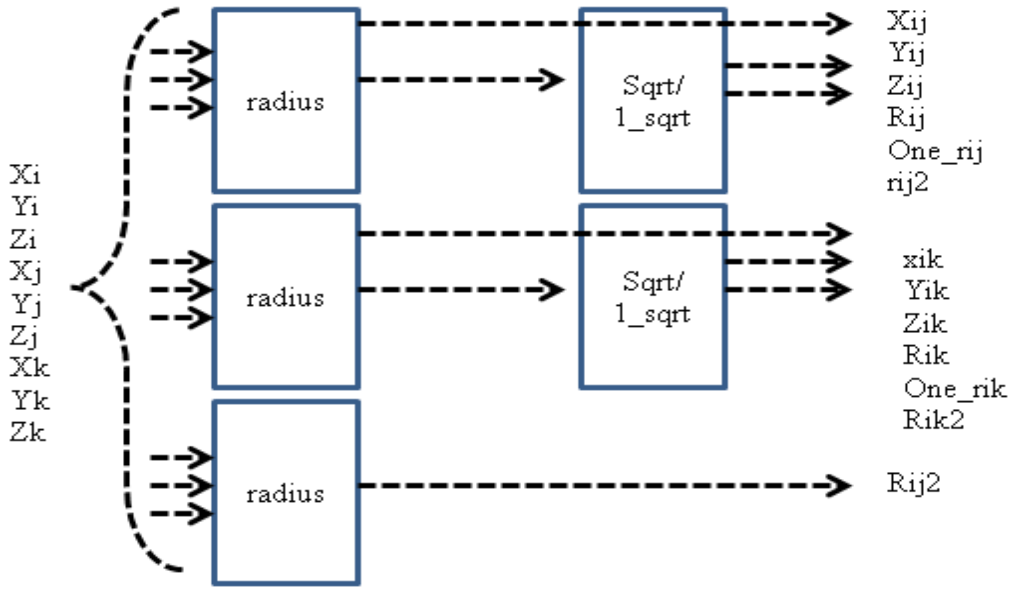


Figure 33: Coordinate module diagram.

4.3Kernel.Energy-Pair:

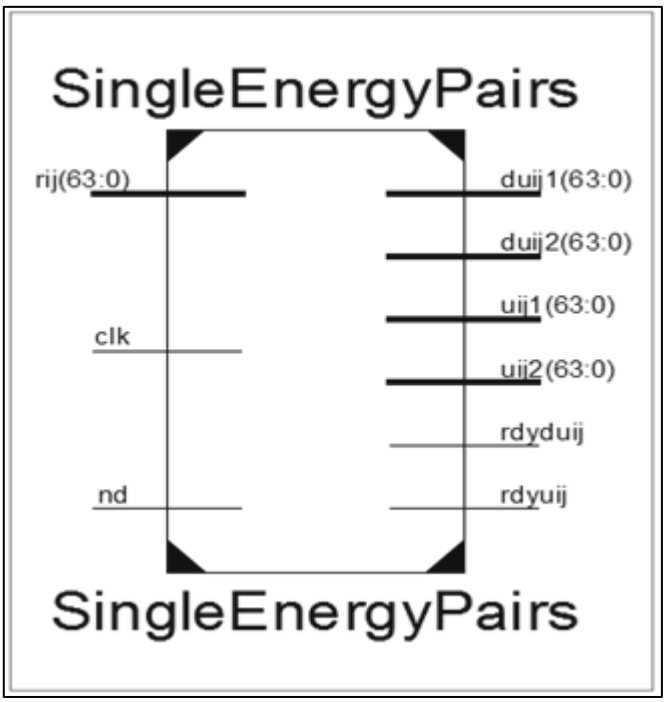


Figure 34: Energy Pair module diagram.

This module generates the main potential calculations for uij1, uij2 duij1 and duij2. It consists of cut-off function and exponential function.

The computation strategy is;

First, find the cut off function and derivative of cut- off function

if $(RIJ < R - D)$

$$fcr = \frac{1}{2}x(1 - sin(\frac{\pi x(RIJ - R)}{2D}) \qquad (4.1)$$

$$\text{dfcr} = -\frac{\pi}{4D}xcos(\frac{\pi x(RIJ - R)}{2D}) \qquad (4.2)$$

else

$$fcr = 1$$

$$dfcr = 0$$

End

Since, we discard the atoms which are not in the range of R+D, we check

R-D condition. This module checks it and store the result in to a buffer. At same time the computation of sin (θ) and cos (θ) argument is calculated. We use single CORDIC module to generate sin (θ) and cos (θ) of arguments. We transform floating points to fixed point because, CORDIC [41] accepts only fixed point

numbers. Finally, selection according to R-D condition is done by multiplexers and we have both fcr and dfcr at the same time.

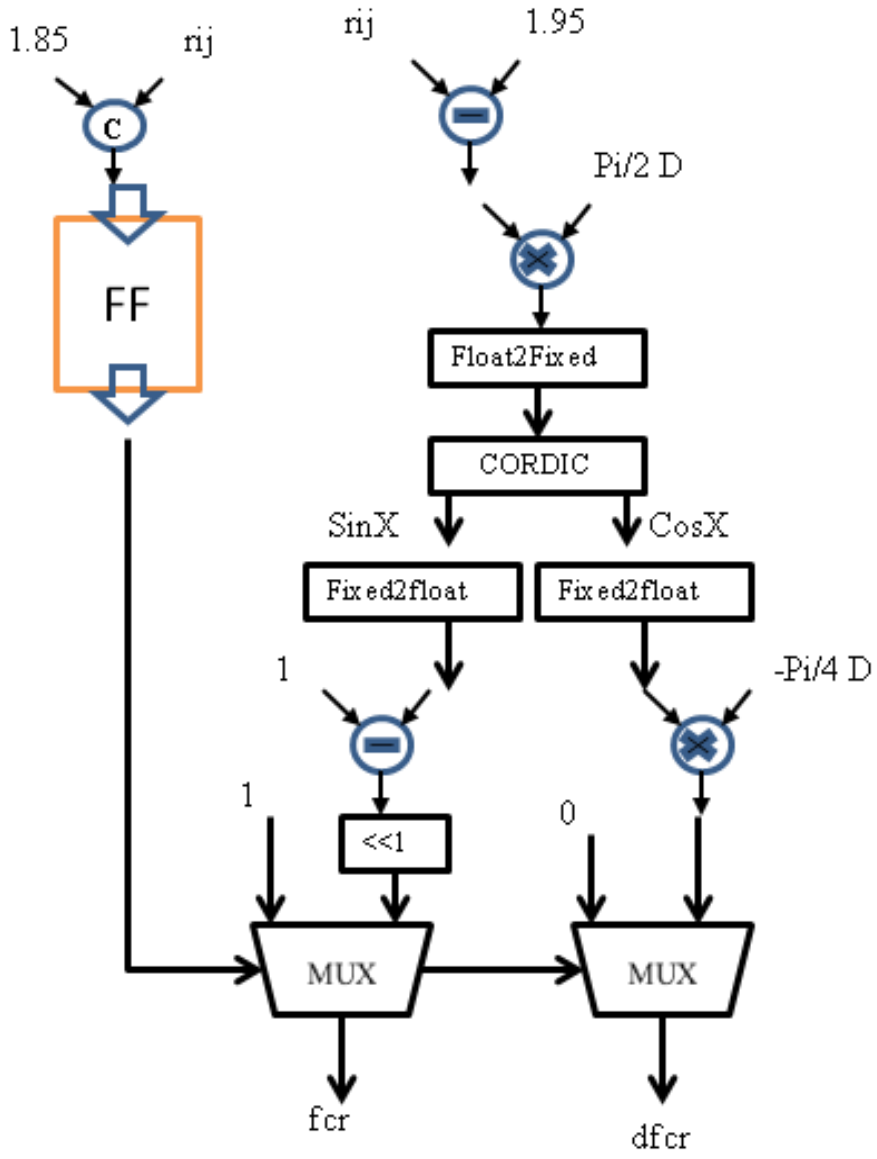


Figure 35: Schematic diagram for FCR cut off function.

Then, find exponential decays and their derivatives simultaneously,

$$UIJ1 = FCIJxe^{(-\lambda_1 xRIJ)} \quad (4.3)$$

$$UIJ2 = FCIJxe^{(-\lambda_2 xRIJ)} \quad (4.4)$$

$$DUIJ1 = (DFCIJ - \lambda_1 xFCIJ)xe^{(-\lambda_1 xRIJ)} \quad (4.5)$$

$$DUIJ2 = (DFCIJ - \lambda_2 xFCIJ)xe^{(-\lambda_2 xRIJ)} \quad (4.6)$$

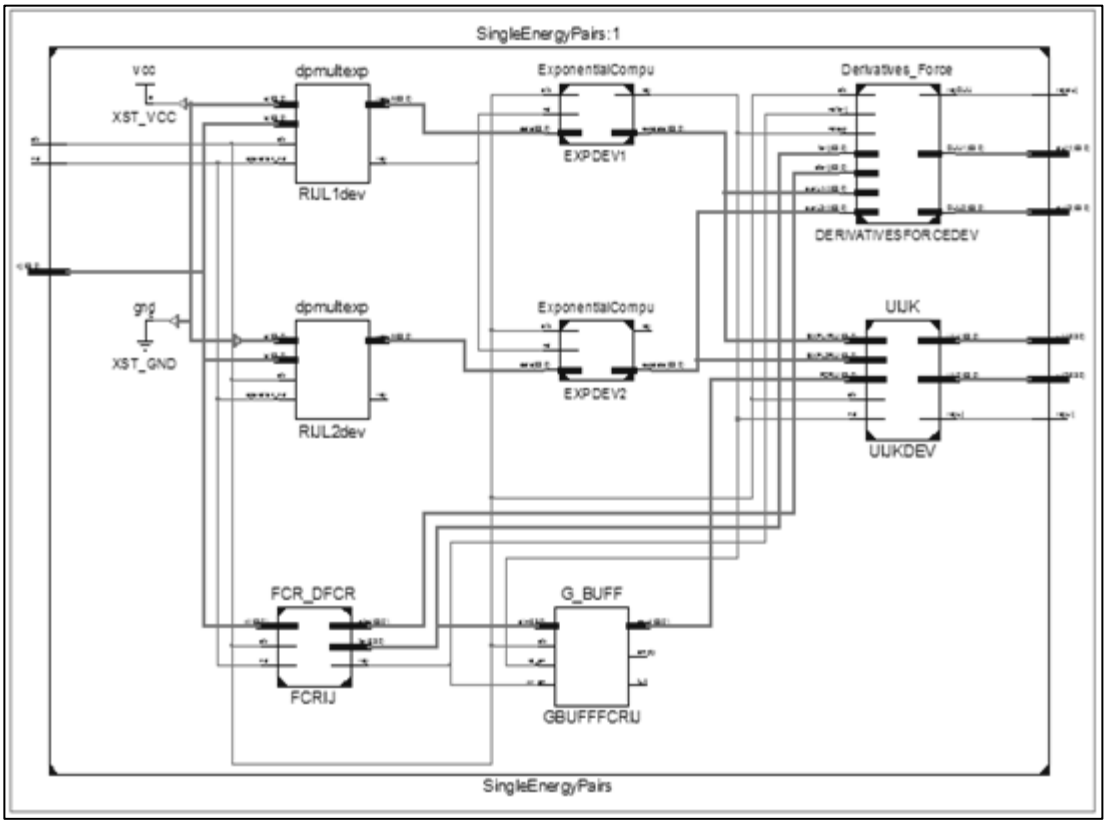


Figure 36: Schematic diagram of single energy pairs.

The exponential function explained in the library.

Derivative_Force function solves following equation in pipelined mode;

$$DUIJ1 = (DFCIJ - \lambda_1 xFCIJ)xe^{(-\lambda_1 xRIJ)} \quad (4.7)$$

$$DUIJ2 = (DFCIJ - \lambda_2 xFCIJ)xe^{(-\lambda_2 xRIJ)} \quad (4.8)$$

There are subtraction and multiplication of floating point cores in it. It simultaneously generates derivative of potentials 4.4 and 4.5.

In this digital circuit, first we calculate cut-off function 4.1, derivative of cut-off function 4.2, rijx$(-\lambda_1)$ and rijx$(-\lambda_2)$ values simultaneously. Then, we calculate exp (rijx$(-\lambda_1)$), exp (rijx$(-\lambda_2)$) in parallel while buffering dfcr term in 4.2. By following schematic flow finally, we compute 4.7, 4.8, 4.3 and 4.2 at the same time. At the end, send all of the result at the same time.

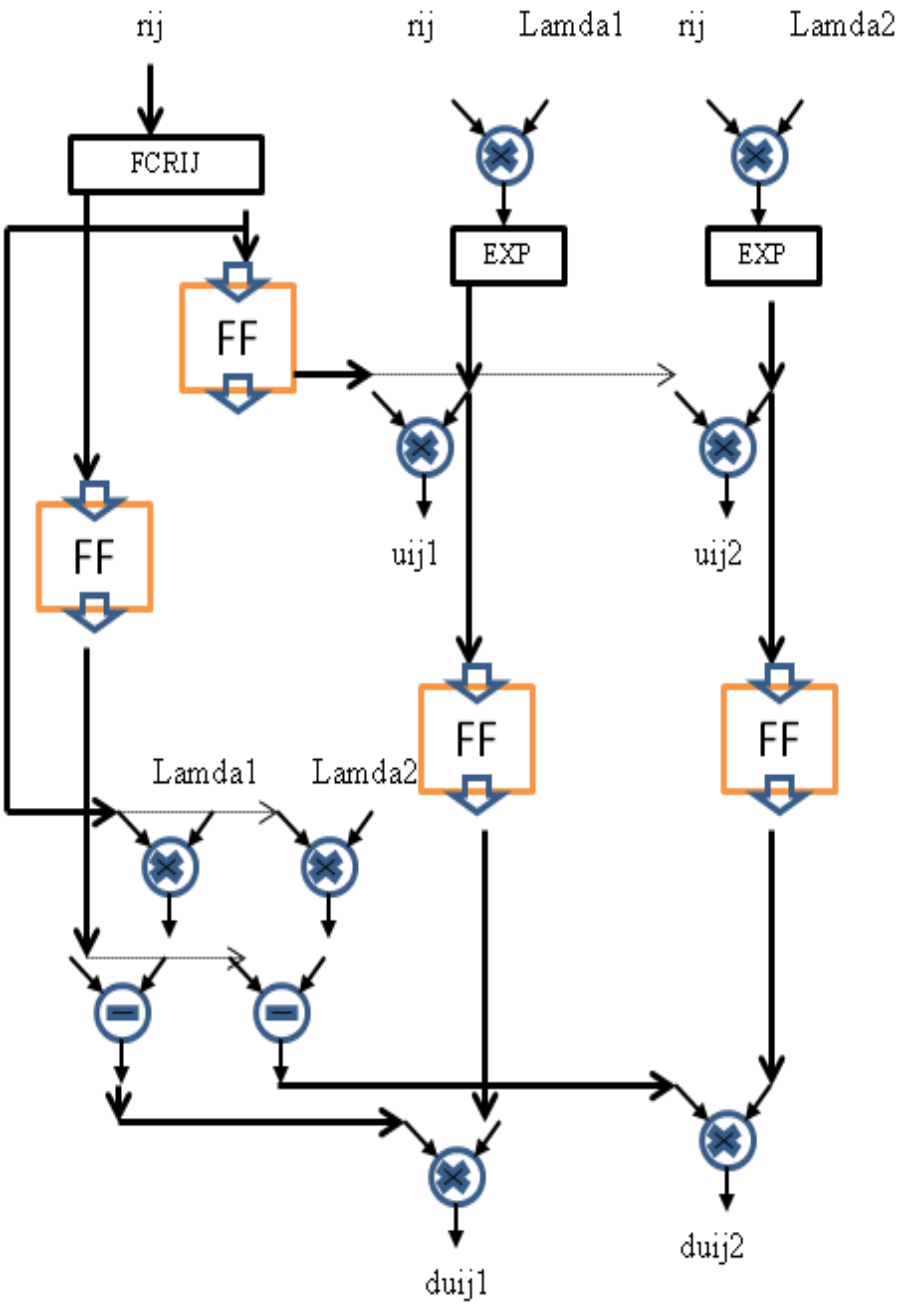


Figure 37: Schematic diagram for energy pairs and their derivatives.

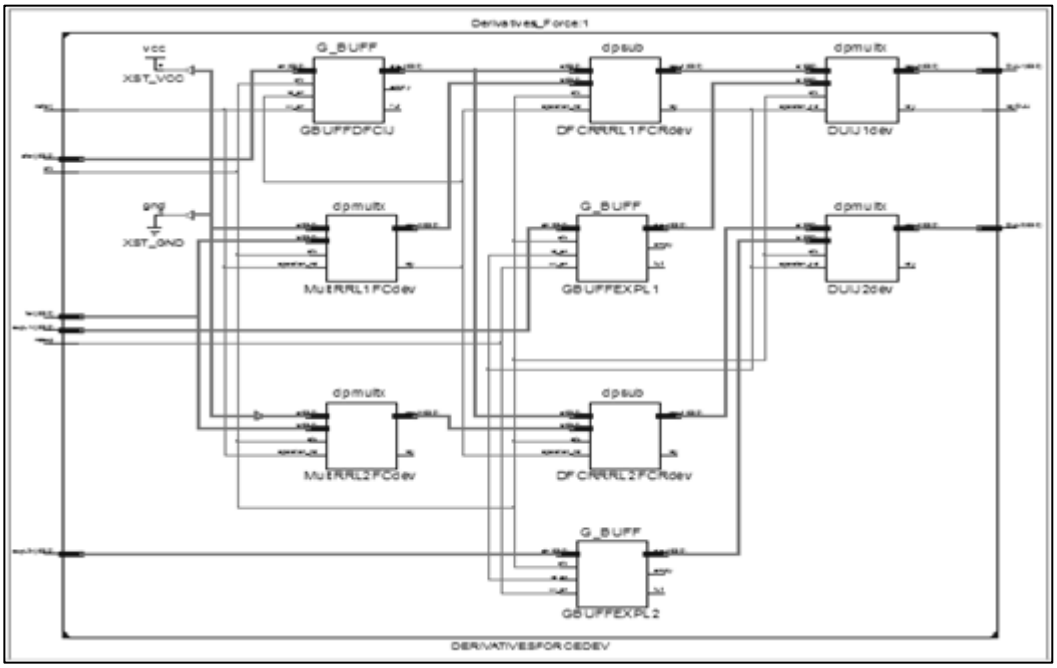


Figure 38: Schematic diagram of DerivativeForce module.

4.4 Kernel.GTETA function:

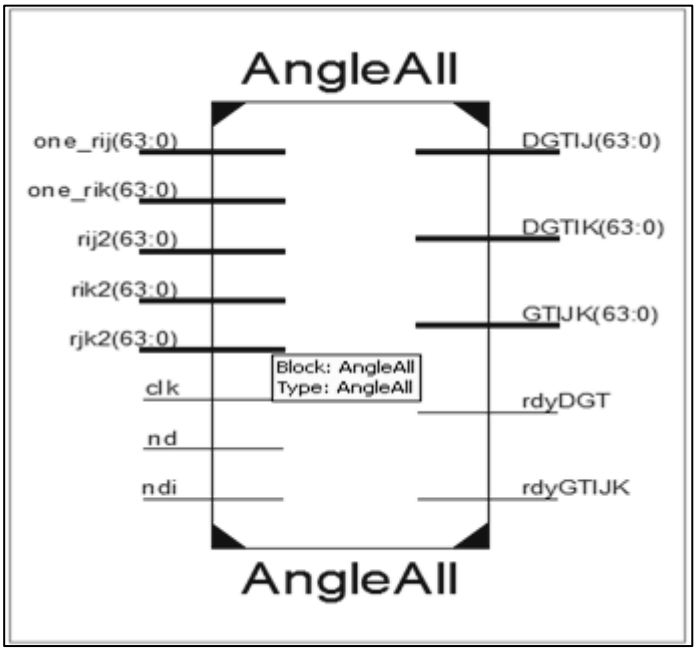


Figure 39: The black box representation of GTETA device.

This module is responsible to generate cosine of angle between ijk and three pairs sub parameters of g(θ) and dg(θ). It solves following equations simultaneously.

$$CTIJK = \frac{(RIJ^2 + RIK^2 - RJK^2)}{(2xRIJxRIK\ )} \quad (4.8)$$

$$g(\theta) = 1 + \frac{c^2}{d^2} - \frac{c^2}{(d^2 + (h - CTIJK)^2)} \quad (4.9)$$

$$DGTIJ = \frac{c^2 x(h - CTIJK)}{(d^2 + (CTIJK)^2)^2} x(\frac{1}{RIK} - \frac{RIK}{RIJ^2} + \frac{RJK^2}{RIJxRIJxRIK}) \quad (4.10)$$

$$DGTIK = \frac{c^2 x(h - CTIJK)}{(d^2 + (CTIJK)^2)^2} x(\frac{1}{RIJ} - \frac{RIJ}{RIJ^2} + \frac{RJK^2}{RIJxRIKxR\ IK}) \quad (4.11)$$

The Cosine function explained in the library.

The module start to calculate the distance vector related parts at the same time with Cosine function to reduce computation time. When the result of Cosine function is ready, the related computations of it are calculated. Finally, all of gtijk, dgtij and dgtik are generated.

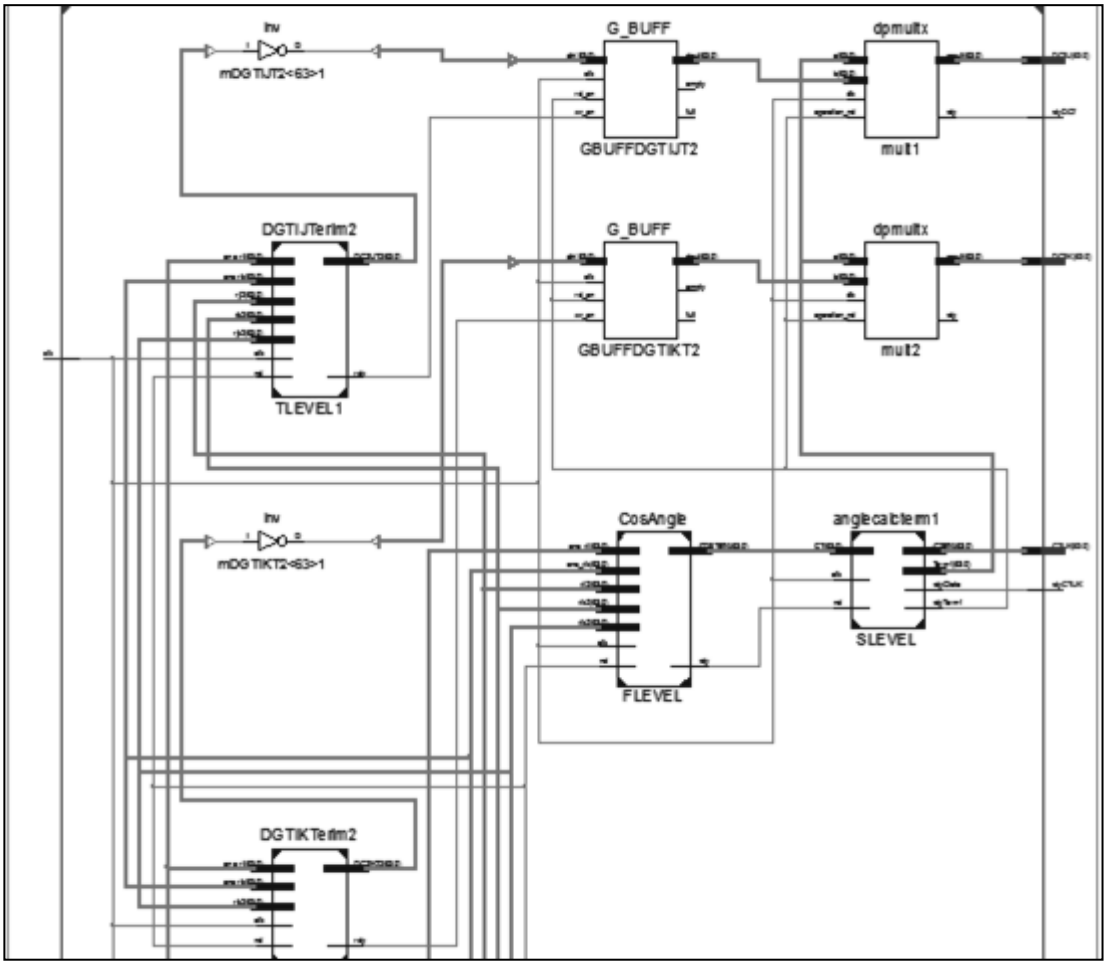


Figure 40: Schematic diagram of GTETA function.

4.5 Kernel.Projection:

This function is explained in the library.

4.6 Kernel.TwoPairsSummation:

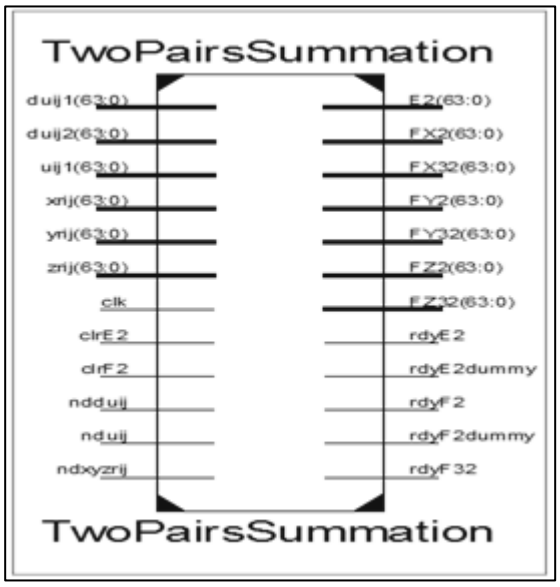


Figure 41: The back box representation of Twopairssummation module

This module is responsible for generating two pair force calculation and energy accumulation for repulsive part of Tersoff2 in the following equations. It takes uij1, duij1, duij2, xrij, yrij, zrij and generates E2, FX2, FY2, FZ2, FX3, FY3, FZ3.

It accumulates UIJ1 for ij pairs.

$$E2 = E2 + UIJ1 \qquad (4.12)$$

It accumulates force from duij1;

$$FX2 = FX2 + DUIJ1xXRIJ \qquad (4.13)$$

$$FY2 = FY2 + DUIJ1xYRIJ \qquad (4.14)$$

$$FZ2 = FZ2 + DUIJ1xZRIJ \qquad (4.15)$$

It generates force from duij2;

$$FX32 = DUIJ2xXRIJ \qquad (4.16)$$

$$FY32 = DUIJ2xYRIJ \qquad (4.17)$$

$$FZ32 = DUIJ2xZRIJ \qquad (4.18)$$

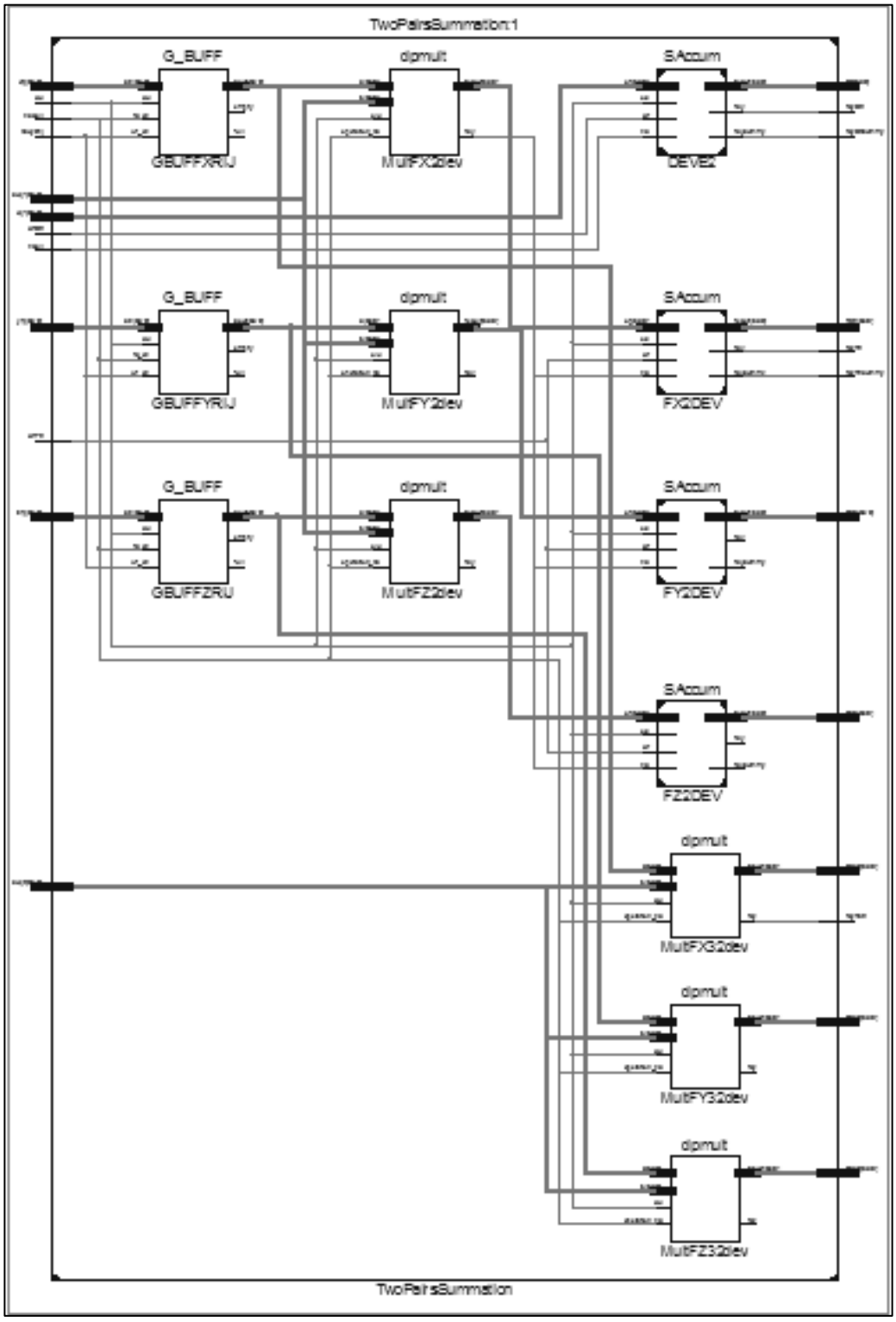


Figure 42: Schematic diagram of Twopairssummation module.

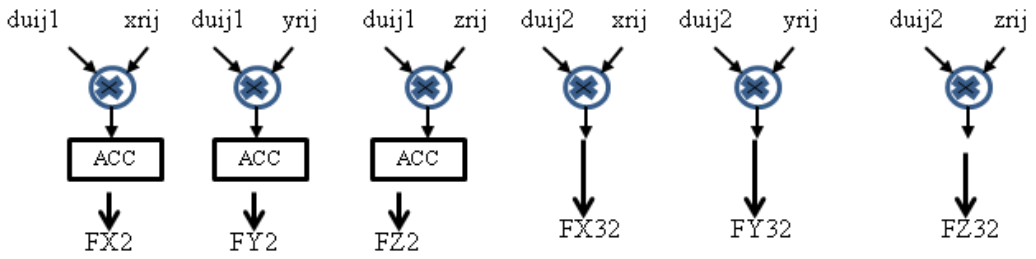


Figure 43: Schematic diagram of Force from two pair energy.

The dpmult instances are floating point multiplications. ACC function is simple accumulation device. G_BUFF devices for synchronization of the function process times.

4.7 Kernel.ThreePairsSummation:

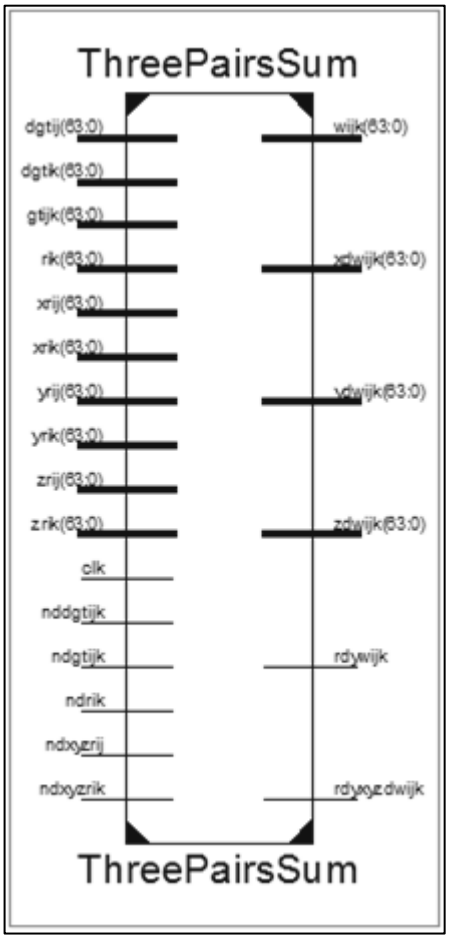


Figure 44: The black box diagram of ThreePairssummation Device.

This is responsible to generate three pair potential sub parameters WIJK, XDWIJK, YDWIJK, ZDWIJK from projections and dgtij, dgtik, gtijk. It consists of cut off function, log adders, multiplications and adders. It computes following equations;

$$WIJK = WIJK + FCIKxGTIJK \quad (4.19)$$

$$DWIJ = FCIKxDGTIJ \quad (4.20)$$

$$DWIK = DFCIKxGTIJK + FCIKxDGTIK \quad (4.21)$$

$$XDWIJK = XDWIJK + DWIJxXRIJ + DWIKxXRIK \quad (4.22)$$

$$YDWIJK = YDWIJK + DWIJxYRIJ + DWIKxYRIK \quad (4.23)$$

$$ZDWIJK = ZDWIJK + DWIJxZRIJ + DWIKxZRIK \quad (4.24)$$

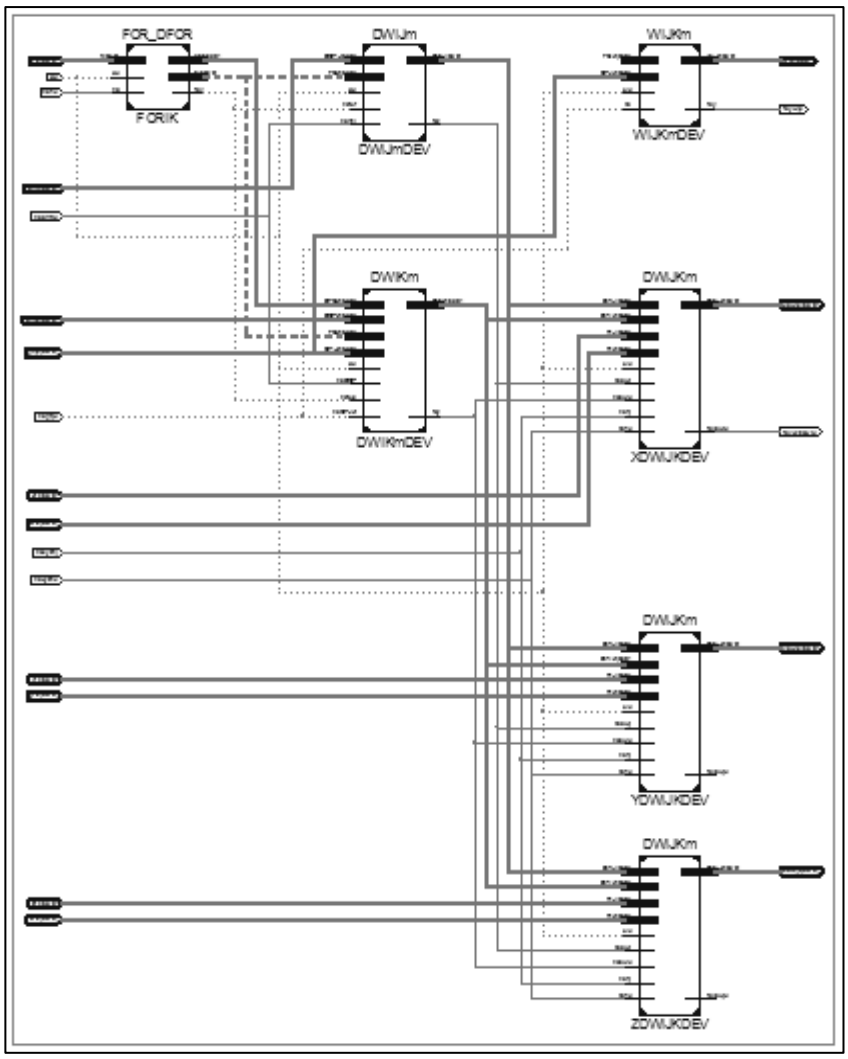


Figure 45: schematic diagram of ThreepairsSummation Device.

The device DWIJKm is responsible for solve;

$$XDWIJK = XDWIJK + DWIJxXRIJ + DWIKxXRIK \quad (4.25)$$

and replicated three times for computing YDWIJK, ZDWIJK. Log adder function is used for accumulation of this term.

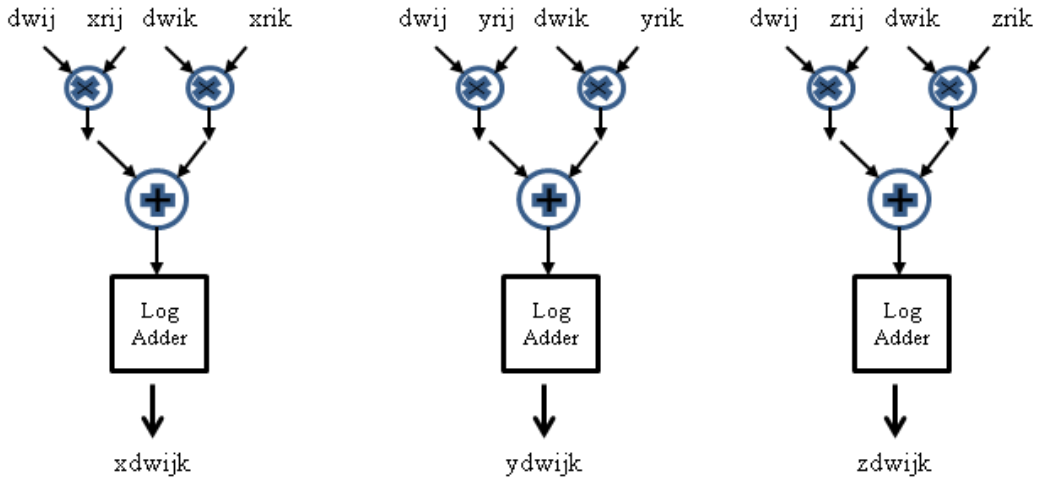


Figure 46: Schematic diagram of DWIJKm Device.

4.8 Kernel.PowerSystem:

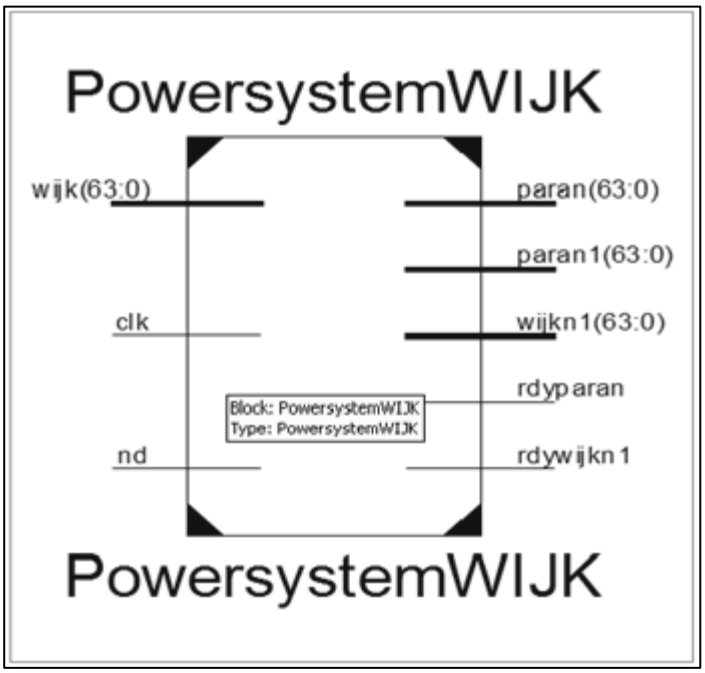


Figure 47: The black box diagram for powersystem module.

Powersystem module is responsible to generate following equations. In the design, we use logarithm function and exponential function to find the results.

$$WIJKN = WIJKN^{n} \quad (4.26)$$

$$WIJKN = WIJKN^{n-1} \quad (4.27)$$

$$PARAN = (1 + \beta^{n} xWIJKN)^{m} \quad (4.28)$$

$$PARAN1 = (1 + \beta^{n} xWIJKN)^{m-1} \quad (4.29)$$

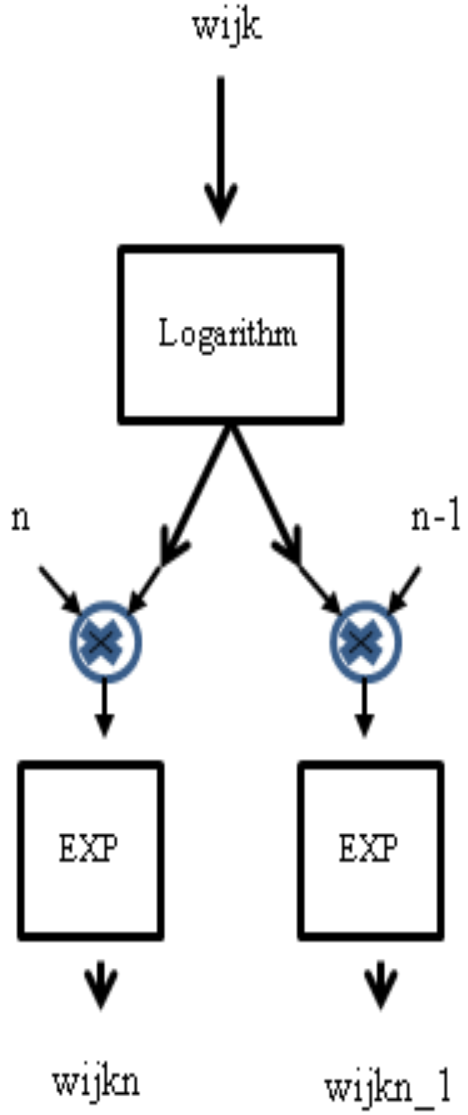


Figure 48: Schematic diagram for power calculation.

In the library, power function explained. We replicate this module to find PARAN, and PARAN1 terms. It consumes 290 DSP units and can operate up to 330 MHz.

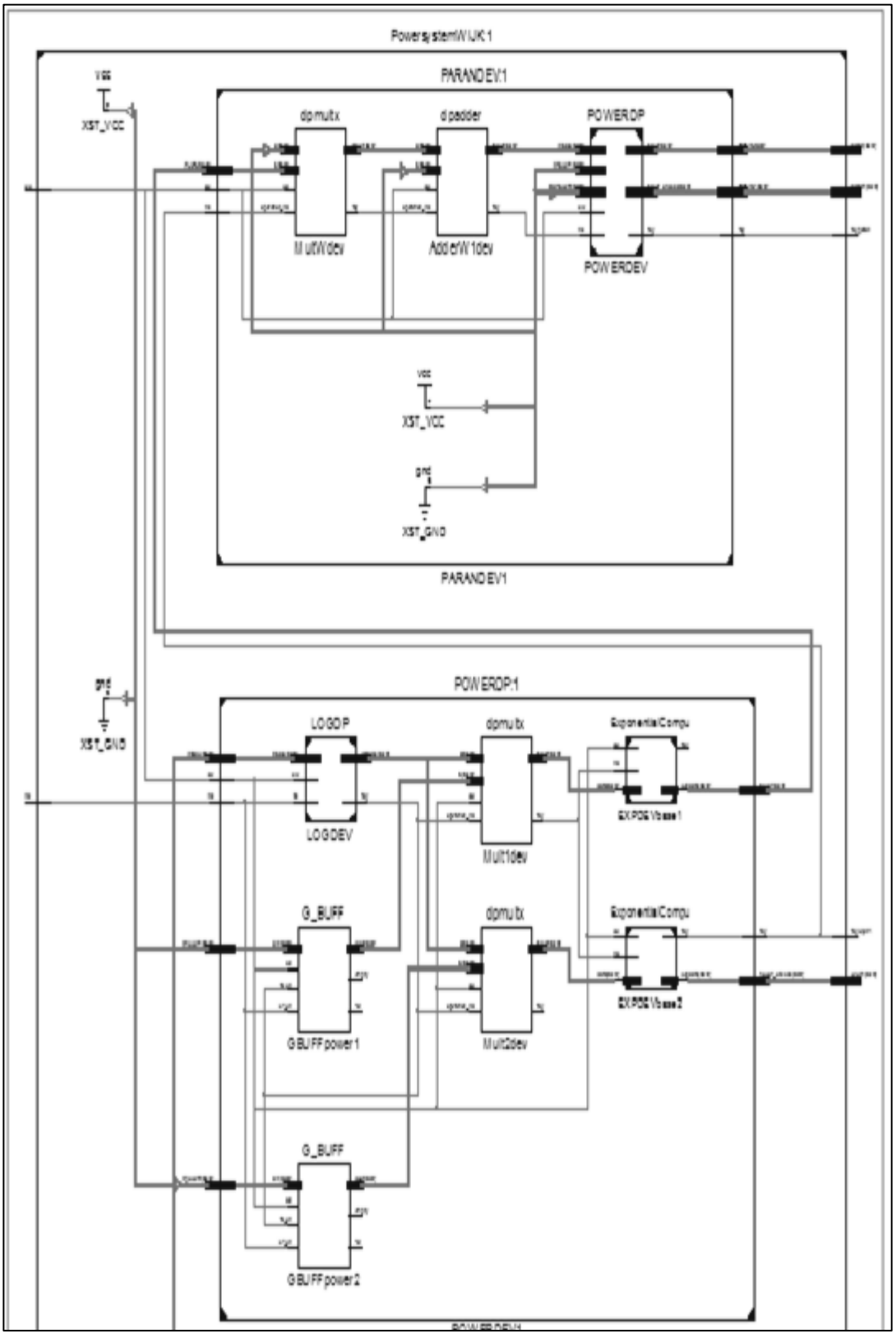


Figure 49: Detailed Schematic diagram for WIJKn, WIJKn1, PARAN and PARAN1 power calculation.

4.9 Kernel.ForceThreePairs:

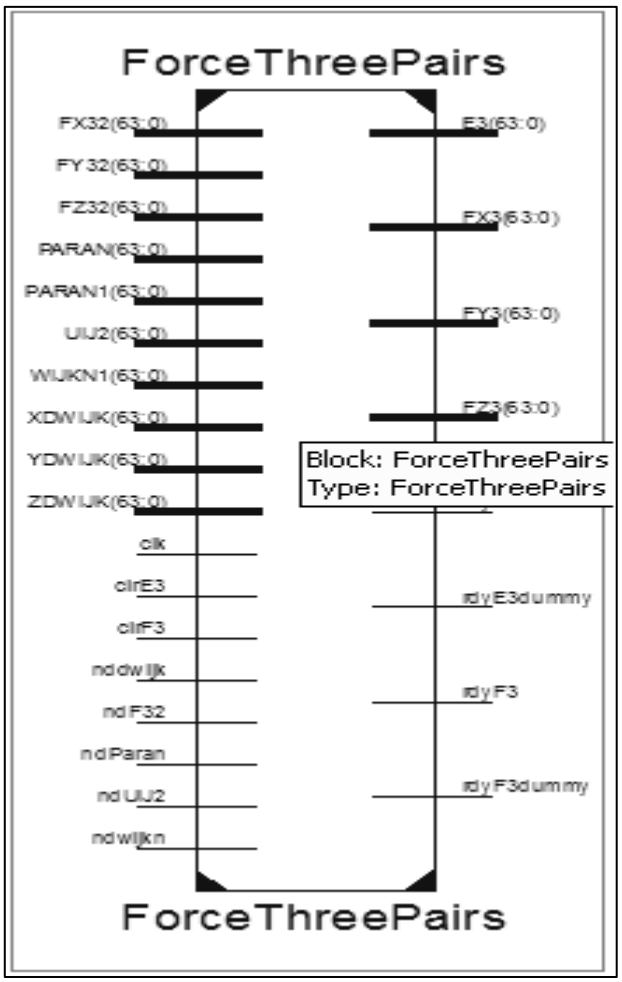


Figure 50: The black box diagram for Forcethreepairs device.

This module is responsible for calculating following equations;

$$F_{x3} = F_{x3} + F_{x32}xPARAN - 0.5x\beta^n xU_{ij\,2} * PARAN1xWIJKN1xXDWIJK \quad (4.30)$$

$$F_{y3} = F_{y3} + F_{y32}xPARAN - 0.5x\beta^n xU_{ij\,2} * PARAN1xWIJKN1xYDWIJK \quad (4.31)$$

$$F_{z3} = F_{z3} + F_{z32}xPARAN - 0.5x\beta^n xU_{ij\,2} * PARAN1xWIJKN1xZDWIJK \quad (4.32)$$

$$E_3 = E_3 + U_{ij\,2}xPARAN \quad (4.33)$$

At the first look each equation has an accumulation part and 6 terms of multiplication. There isn't any mathematical function that can calculate 6 multiplications. So, we decompose 6 of them and do the multiplications with two operands in parallel. This design consumes 111 DSP units and can operate up to 330 MHz.

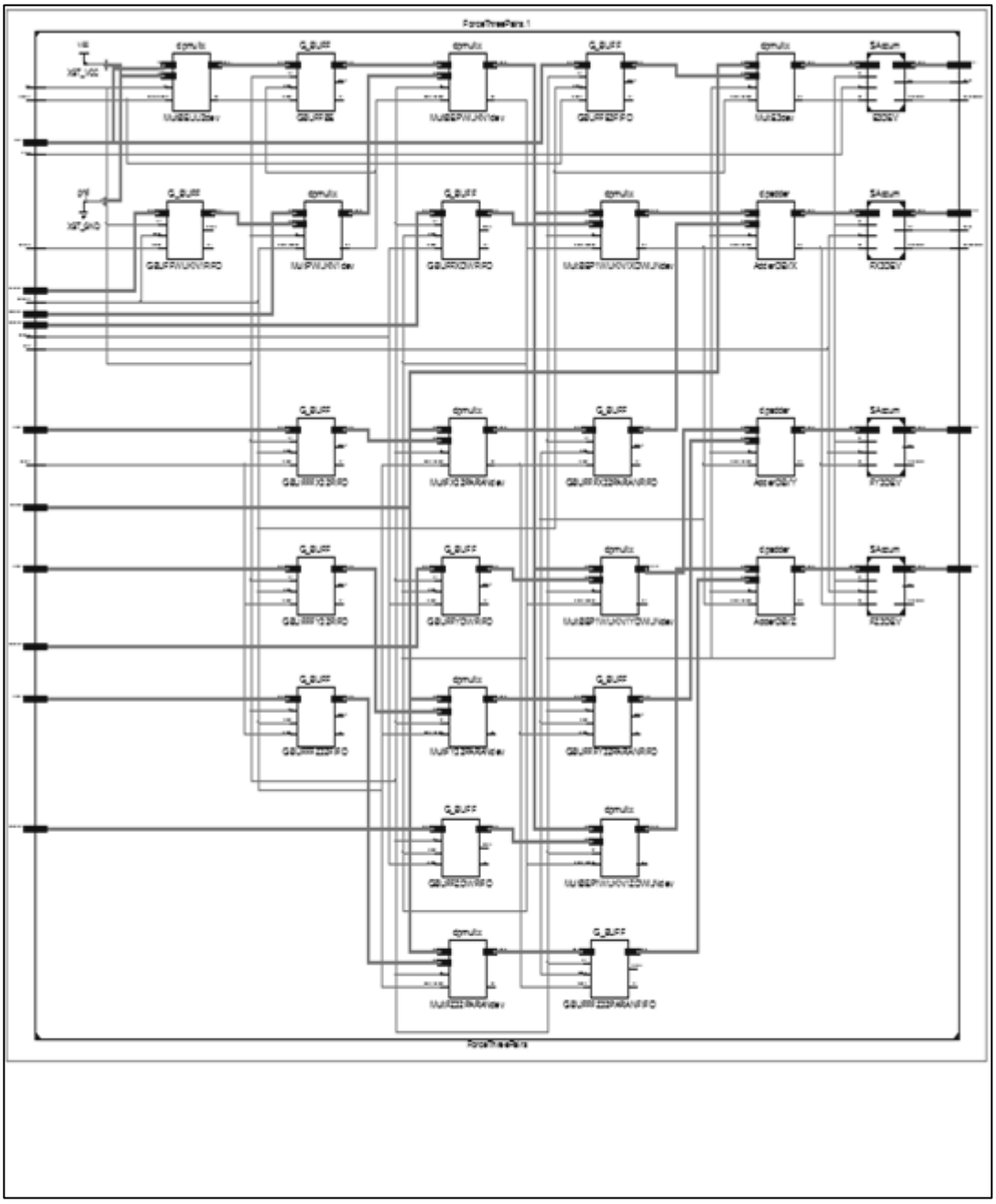


Figure 51: Schematic diagram for Forcethreepairs device.

Decomposition of such kind of equation;

$F_{x3} = F_{x3} + F_{x32} xPARAN - 0.5x\beta^n xU_{ij\,2} * PARAN1xWIJKN1xXDWIJK$ (4.34)

First, do the multiplications *0.5*BETAN*UIJ2* and *PARAN1*WIJKN1* in parallel then, distribute this term over XDWIJK, YDWIJK, ZDWIJK with three multiplication structure. And finally, accumulate them with simple accumulation device.

4.10 Kernel.ForceM:

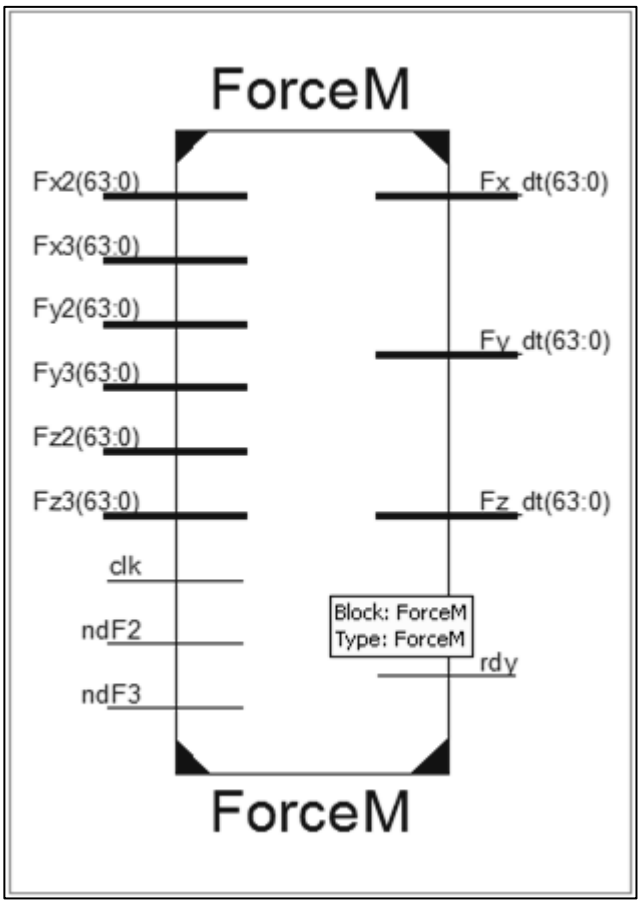


Figure 52: The Black box diagram ForcetM device.

This module is responsible for calculation of following equations;

$$F_x = BxF_{x3} - AxF_{x2} \quad (4.35)$$

$$F_y = BxF_{y3} - AxF_{y2} \quad (4.36)$$

$$F_z = BxF_{z3} - AxF_{z2} \quad (4.37)$$

In the design, we calculate three equations in parallel. All of the structure consumes 60 DSP units and can operate up to 330 MHz. First, we do the multiplications in parallel. Finally, we subtract the following equations to get results.

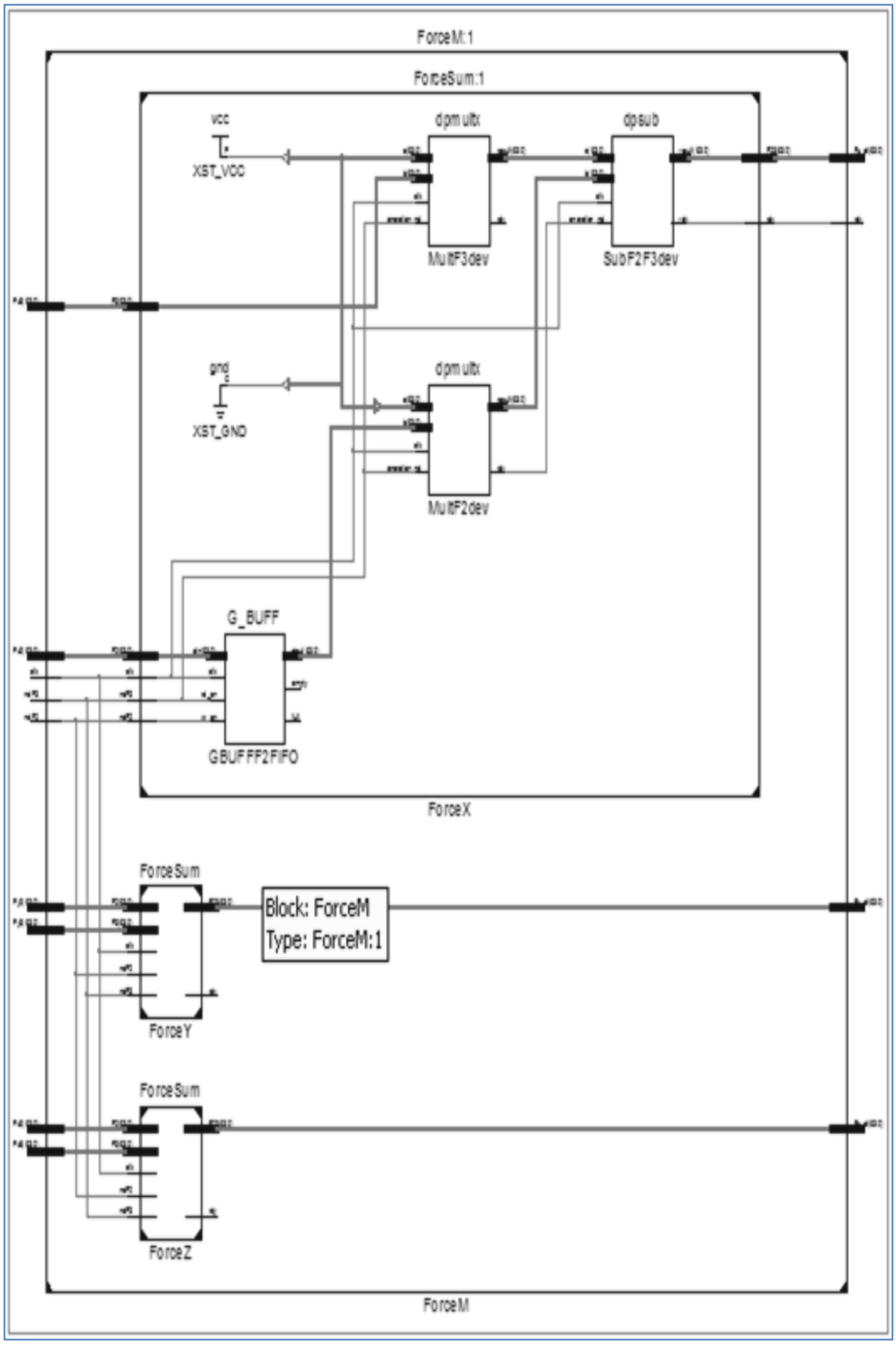


Figure 53: Schematic diagram ForcetM device.

4.11 Kernel.EnergyM:

This module is responsible for calculating following equation;

$$Energy_{Potential} = \frac{1}{2} x(AxE_2 - BxE_3) \qquad (4.38)$$

This equation can be computed with same strategy in *Kernel.ForceM* device. We don't need to calculate division by 2 with real resources. We can change A and B into A/2 and B/2 since they are constants.

CHAPTER 5

CONCLUSIONS

In this work, we reach almost X1000 acceleration with respect to an ordinary computer in terms of computation time. According to simulation of digital circuit, computation of a single MD step energy and force calculation is less than 40 us at 200 MHz clock frequency and the resulting abstract circuit diagram in Figure 54.

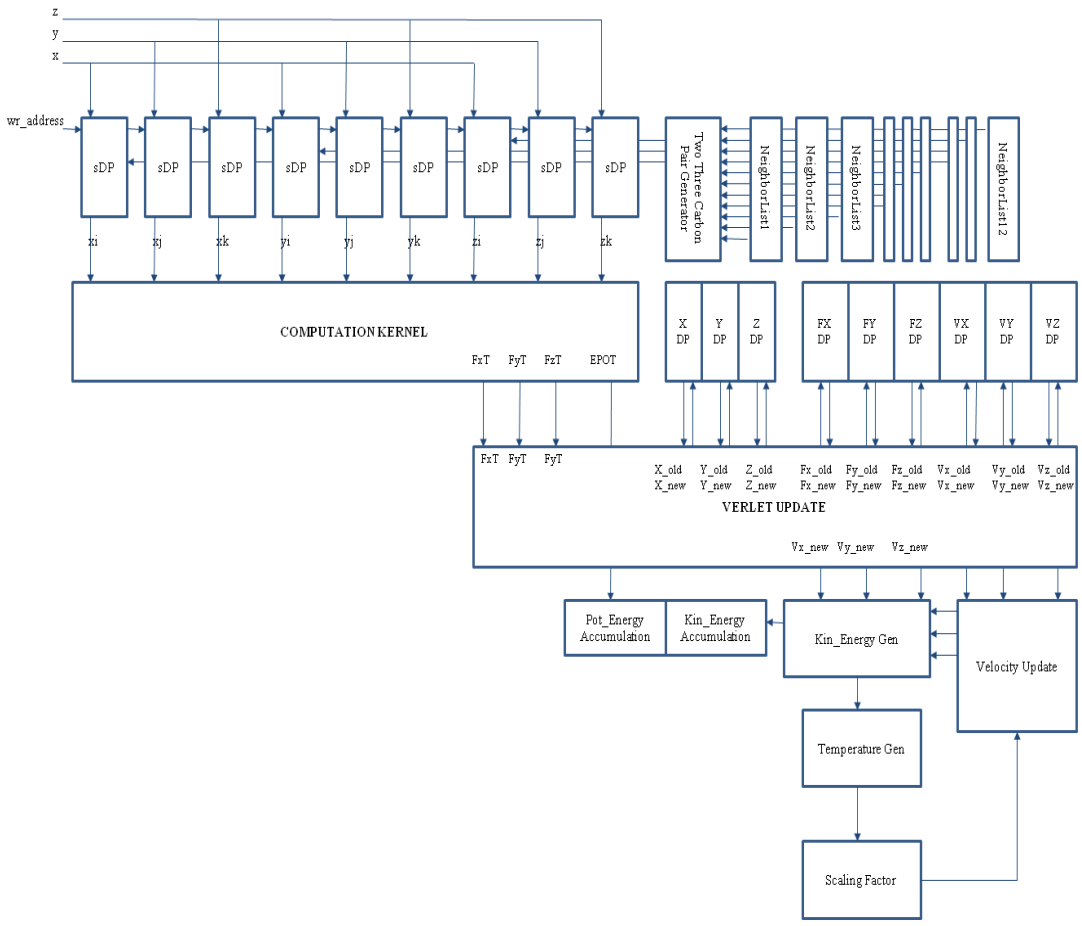


Figure 54: Tersoff2 Main Architecture.

In related chapters, we show that the entire sub modules of the Tersoff2 co-processor are pipelined. When we integrate the sub modules we can reach a fully pipelined architecture. This architecture has best time performance with special memory

structure. When we run the Tersoff on a PC, it will take lots of time cycles to get a single atom's force and energy. Moreover, we have to wait for next atom until previous atom is done. In the contrary, we are going to have the atom's force and energy in one cycle and we don't have to wait for next atom with our design in Figure 55.

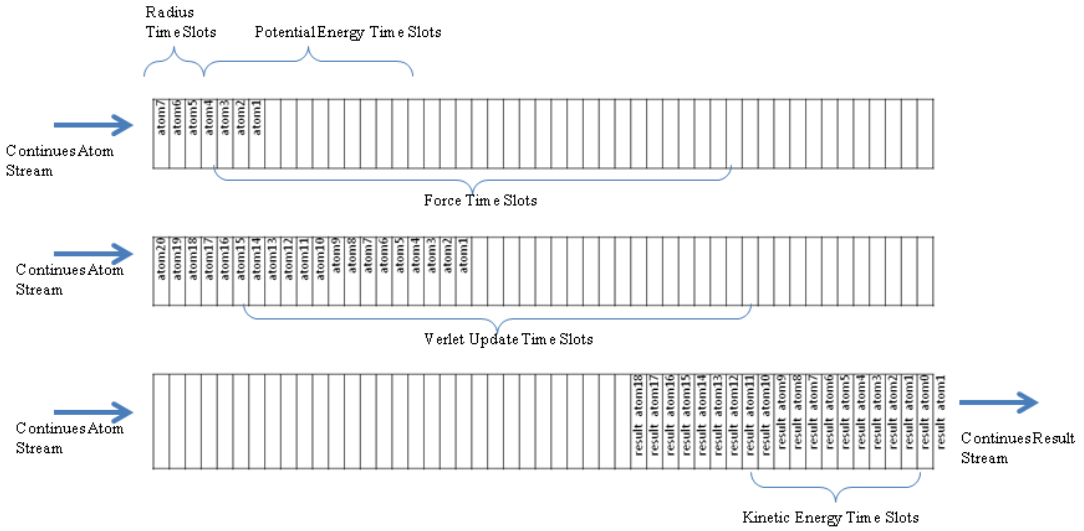


Figure 55: Speedup due to pipelined architecture of Tersoff2

This performance is achieved by fully pipelined architecture with special memory architecture. Detailed timing performance is in Figure 56.

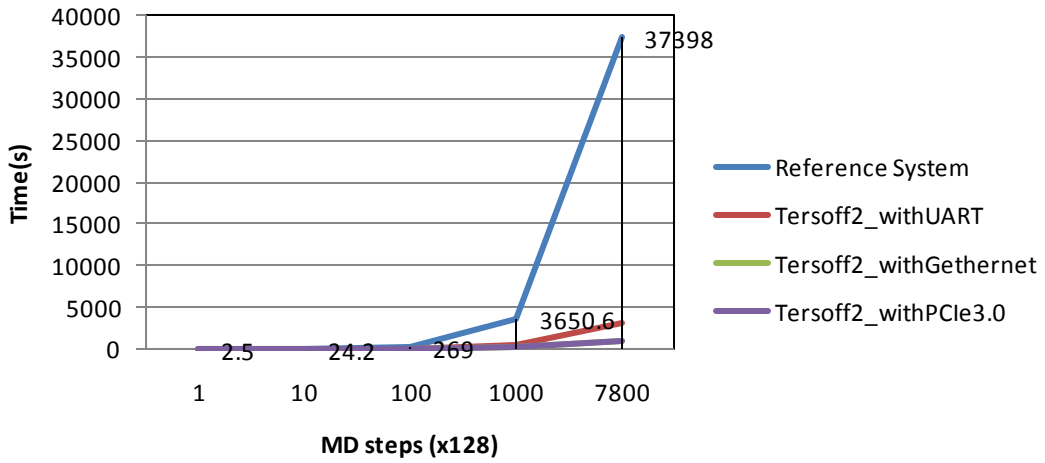


Figure 56: Computation time comparison between reference system and three Tersoff2 systems with three different communication channels.

According to profiler, CPU usage for printing data, takes 14-15 minutes and total Tersoff2 computation takes 35 s.

The total computation time is the sum of tersoff2 computation and the data i/o time in Figure 57.

For 1.000.000 MD step, Tersoff2 computation timing cost;

$$Time = 35\ us\ x\ 1000000$$

$$Time = 35\ s$$

For different communication channel we will have different communication time cost.

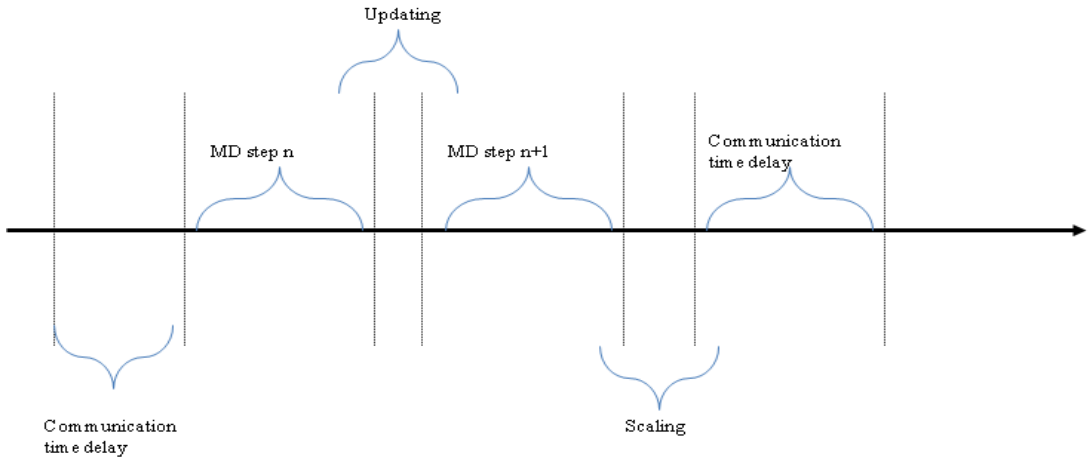


Figure 57: Tersoff time budget.

The resulting device power consumption is less than 50 W. The complete implement-tation is consumes %70 percentage of V7485T. The circuit can handle 4000 atomic system and it can be expanded to 10000 atomic systems. The number of atoms that can be simulated is limited due to limited memory resources of device which we use.

REFERENCES

[1] M.Allen,D, "Computer simulation of Liquids" New York: Oxford University Press,1987

[2] Rapaport, D. "The Art of Molecular Dynamics Simulation" Cambridge University Press, 2004.

[3].Erkoç, Şakir "Empirical many-body potential energy functions used in computer simulations of condensed matter properties" Physics Reports 278 (1997) 79-105

[4] Tersoff, J. "Empirical interatomic potential for silicon with improved elastic properties" Phys. Rev. Lett. 61 (1988) 2879.

[5] Wielgosz, Maciej. Jamro,Ernest "High Efficient Structure of 64 bit Exponential Function Implemented in FPGAs" FPL 2007 718-721

[6] Pineiro, Jose. Bruguera, Javier "High Speed Double Precision Computation of Reciprocal, Division, Square Root and Inverse Square Root" IEEE Transactions on computers Vol. 51 No.12 1377

[7] Ercegovac, M.D. Lang, T. "Reciprocation, Square Root, Inverse Square Root and some elementary functions Using Small Multipliers" Computers, IEEE Transactions on 49 , Issue: 7 Page(s): 628- 637 2000

[8] Ercegovac, M.D. Imbert, L. "Improving Goldschmidt Division, Square Root and Square Root Reciprocal," IEEE Trans. Computers, vol. 49, no. 7, pp. 759-763, July 2000.

[9] Ercolessi Furio, "A molecular dynamics primer" http://www.fisica.uniud.it/ ercolessi/md/md.pdf Jun, 1997

[10] Xiaidong, Y. Shengmei, M. "FPGA-Accelerated Molecular Dynamics Simulations: An Overview" ARC 2007, LNCS 4419 pp 293-301 Sprienger-Verlag 2007

[11] Scrofano, R. Prsanna, V. "Preliminary Investigation of Advanced Electrostatics in Molecular Dynamics on Reconfigurable Computers" ACM|IEEE SC06 2006

[12] Cho, E. Bourgeois, A. "Efficient and accurate FPGA-based simulator for Molecular Dynamics" IEEE International Symposium on Parallel and Distributed Processing, 2008. IPDPS 2008. 1-7

[13] Pottathuparambil, R Sass, R." Efficient and Accurate FPGA-based simulator for Molecular Dynamics" 2010 IEEE

[14] F.H.Stillinger and T.A.Weber "Computer simulation local order in condensed phases of silicon" Physical review B, vol.31, no.8, pp.5262 April 1085

[15] Guo, H. Su, L. Wang, Y. "FPGA-Accelerated Molecular Dynamics Simulations System" EmbeddedCom-ScalCom2009 IEEE

[16] Gu, Y. Vancourt, T. Herbordt, M.C. "Accelarating molecular dynamics simulations with configurable circuits" Vol. 153, No. 3 May 2006 IEEE Proc.Comput.Digit. Tech.

[17] Ercolessi, F."The Tersoff Potential" http://www.fisica.uniud.it/ ercolessi/ md/ md /node51.html

[18] Berber, S. Kwon, Y.K. "Unusually High Thermal Conductivity of Carbon Nanotubes" Phys. Review Letters Vol.54 No.20 4613-4616 2000

[19] Maruyama, S. "A Molecular Dynamics Simulation of Heat Conduction of Finite Length Single-Walled Carbon Nanotube" Microscale Thermophysical Engineering 7:1, 41-50 2003

[20] Liew, K.M. He, X. Q. "On the study of elastic and plastic properties of multi-walled carbon nanotubes under axial tension using molecular dynamics simulation" Acta Materialia 52(2004) 2521-2527

[21] Mylvaganam, K. Zhang, L.C. "Important issues in a molecular dynamics simulation for characterizing the mechanical properties of carbon nanotubes" Carbon 42 (2004) 2025-2032

[22] Huhtala, M. Kuronen, A. "Carbon Nanotube Structure: Molecular Dynamics Simulation at Realistic Limit" Computer Physics Communications 146 (2002) 30-37

[23] Kang, J.W. "Gigahertz actuator of multiwall carbon nanotube encapsulating metallic ions: Molecular Dynamics Simulations" Journal of Applied Physics Vol.96 Issue: 7 3900-3905 2004

[24] Charles, F. C. Charles, R.W. "Very-high-strength (60-GPa) carbon nanotube fiber design based on molecular dynamics simulations" J. Chem. Phys. 134, 204708 (2011)

[25] Davoodi, J. Alizade, H. Rafii-Tabar, H. "Molecular Dynamics Simulation of Carbon Nanotubes Melting Transitions" Journal of Computational and Theoretical Nanoscience, Volume 9, Number 4, April 2012 , pp. 505-509(5)

[26] Erik, C. N. Adri, C. T. Annemie, B. "Changing Chirality during Single-Walled Carbon Nanotube Growth: A Reactive Molecular Dynamics/Monte Carlo Study" J. Am. Chem. Soc., 2011, 133 (43), pp 17225–17231

[27] Gordana, D. Brian, E. White, Z. Z. "Reversible Surface Oxidation and Efficient Luminescence Quenching in Semiconductor Single-Wall Carbon Nanotubes" J. Am. Chem. Soc., 2004, 126 (46), pp 15269–15276

[28] Junfeng, G. Jijun, Z. Feng, D.“ Transition Metal Surface Passivation Induced Graphene Edge Reconstruction” J. Am. Chem. Soc., 2012, 134 (14), pp 6204–6209

[29] Cheng-Da W. Te-Hua F. Chi-Yu, C. “A molecular dynamics simulation of the mechanical characteristics of a C60-filled carbon nanotube under nanoindentation using various carbon nanotube tips” Carbon 49 (2011) 2053–2061

[30] Jeong, W. K., Oh, K. K. “A molecular dynamics simulation study on resonance frequencies comparison of tunable carbon-nanotube resonators” Applied Surface Science 258 (2012) 2014– 2016

[31] Jong, W. Y. Ho, J. H. “Molecular dynamics modeling and simulations of a single-walled carbon-nanotube-resonator encapsulating a finite nanoparticle” Computational Materials Science 50 (2011) 2741–2744

[32] Bruno, F. Nuno, S. José, N. C. L. “Strength and stiffness of carbon nanotubes under combined axial force and torsion via molecular dynamics simulations” ICCS 16 Porto 2011

[33] Ansari, R. Ajori, S. Arash, B. “Vibrations of single- and double-walled carbon nanotubes with layer wise boundary conditions: A molecular dynamics study” Current Applied Physics 12 (2012) 707-711

[34] Masaaki, Y. Yoshinori, C. “Performance evaluation of carbon nanotube-based oscillators and bearings under electron irradiation: Molecular dynamics study” Microelectronic Engineering 97 (2012) 241–246

[35] Jin-Wu, J. Wang, B.S. “Molecular dynamics simulation for heat transport in thin diamond nanowires” Physical Review B 83, 235432 (2011)

[36] Gu, Y. VanCourt, T. Herbordt, M.C. “Explicit design of FPGA-based co-processors for short-range force computations in molecular dynamics simulations” Parallel Computing, Volume 34, Issues 4–5, May 2008, Pages 261-277

[37] Kim, J.S. “TANOR: A Tool for Accelerating N-Body Simulations on Re-configurable Platform” Field Programmable Logic and Applications, FPL 2007. 68-73

[38] Gu, Y Herbordt, M.C. “FPGA-Based Multigrid Computation for Molecular Dynamics Simulations” FCCM '07 Proceedings of the 15th Annual IEEE Sym-posium on Field-Programmable Custom Computing Machines Pages 117-126

[40] Cho, E. Bourgeois, A. Tan, F. “An FPGA Design to Achieve Fast and Accurate Results for Molecular Dynamics Simulations” Parallel and Distributed Processing and Applications Lecture Notes in Computer Science Volume 4742, 2007, pp 256-267

[41] Xilinx Datasheet www.xilinx.com/support/documentation/ip_documentation/cordic_ds249.pdf

[42] Virtex 7 Userguide Xilinx Datasheet www.xilinx.com/ products/silicon-devices/fpga/virtex-7/index.html