Implementation of Jacobi Iterative Solver using Verilog HDL

1 Asmita Prabhakar Wakchaure, 2 Ms. Anitha Patibandla

1Student at MRCET College Affiliated from JNTU, Hyderabad, India.
2Assistant Professor at MRCET, Hyderabad, India.

Abstract
There are many engineering simulations and scientific computing applications where large linear system of equations is to be solved. With the generous development in scientific computing the use of large linear system of equations also increases. There are many solvers for solving these equations. Jacobi is one of the important solvers for solving linear system of equations. Jacobi method is more efficient when implemented in a pipelined manner in an FPGA and it also shows more parallelism in this paper hardware based implementation of Jacobi solver is discussed. The main idea of Jacobi to obtain preference is its computation that can be trivially done in parallel.

I. INTRODUCTION

There are two most common ways to solve large linear system of equations. These methods can be classified into two classes-direct method and iterative method. Generally the former one is more preferable [1]. In order to solve a system of linear equations there are two classes of iterative methods stationary iterative methods and non-stationary iterative Methods. In stationary iterative method the coefficients are iteration independent [2]. In case of Non stationary iterative methods coefficients are iteration dependent. It is represented by the computations that are dependent on the information of the current iteration step [2, 3]. Three iterative methods to solve sparse linear system are Jacobi method, Gauss Seidel method and Successive over relaxation (SOR) method

To discuss elaborately these methods let us consider a linear equation of the form Ax=b

Where A is the sparse co-efficient matrix, b and x be the known real n vector and unknown real n vector respectively. To solve this equation iterative method has the capability in terms of memory and time cost which are almost linear with size of [4].

These methods are used with the pre conditioner in order to improve their performance along with robustness [5]. A pre conditioner performs it modification of the original system, where M is a pre conditioner is as follows

\[ M^{-1}Ax = M^{-1}b \]  

II. METHOD ANALYSIS

Jacobi method

Jacobi is one of the important methods for solution of large linear system which is diagonally dominant and stable. It is a stationary iterative method. This method solves every variable locally with respect to the other variables that is one iteration corresponds to every variable once [3]. It is easy to implement as well as understand, but its convergence is quite slow [6]. It is also known as method of simultaneous iteration in which the value of \( X^{k+1} \) is only dependent on the value of \( X^k \)
where \( k \) is number of iterations. Let us consider (1) to understand Jacobi iteration and its iteration can be written as

\[
\sum_{i=1}^{n} q_i x_i = y_i
\]  

(3)

Consider other entries of \( x \) remain fixed when calculating \( x_i \) then it will give

\[
x_{i}^{k+1} = \frac{1}{a_{ii}} [b_i - \sum_{j \neq i} a_{ij} x_{j}^{k}]
\]  

(4)

\( A \) is split into \( A = D + A - D \) where \( D = \{a_{11}, \ldots, a_{nn}\} \), then (1) can be written as

\[
Dx + (A - D)x = b
\]  

(5)

\[
x = D^{-1}(D - A)x + D^{-1}b
\]  

(6)

and the matrix representation is

\[
x_{k+1}^{f} = D^{-1}(D - A)x_k + D^{-1}b
\]  

(7)

III. ARCHITECTURE ANALYSIS

This architecture consists of six multipliers, one divisor unit six delay unit and two subtractor units. The adders, subtractors and multiplier units are designed by using IEEE- 754 Floating Point. The outputs of multiplier1 (mult1) and multiplier2 (mult2) are applied to adder1. To make a delay the output of mult3 is apply to the delay unit which is used to make synchronization between the outputs of mult3 and the 2016 2nd International Conference on Control, Instrumentation, Energy & Communication 103 adder2. Adder2 adds the output of mult4 output from mult3 through a delay unit 1 adder3. Again the output from adder1 through then applied to adder4 which gives the output \( B \) values through a delay unit and the output applied to a subtract or unit (subtractor1) output \( b - \sum a \). The diagonal of \( A \) is applies gives the output of reciprocal of \( A \) through a applied to mult6 with the output from subtract or from mult6 gives the required result \( \frac{1}{A}(b - \sum a x) \). current values of \( x \) through a delay unit an mult5 is applied to subtract or 2 that gives the that calculates \( \|x_{k+1}^f - x_k^f\| \) and the control convergence condition. If the convergence d process is continued otherwise it is terminate.

Figure 1: Architecture of Jacobi S

The architecture can be divided into two mod

1. The calculation module.
2. The error calculation module.

The calculation module solves the equitation iteration. The nature of the Jacobi Method p to use a pipelined design without the data h row of the matrix has six values. This un account by providing the appropriate number units so that every cycle. Because of the pip result of the very first row of the matrix wise 94 cycles after the data enters the unit. Getting into and out of the calculation unit is probably part of this design. The most important issue unit is the use of delays. In a pipelined entered to the pipeline should stay together w time. There are a variety of delays units in which are simply a chain of flip flops mean passing through a pipeline. By using these be ensure that all paths of the design have eq.
In Error module two 64-bit floating point passes for every clock cycle. At first B is floating point unit that holds the track of passed through since the last reset? Same was and mult5. These then value is calculated by the Co through and the largest version module [11]. After that this module after that it will pass the information Divider to start the dividing of largest B value. After the calculation starts comparing the computer error value. If the computed error, the module asserts the there is no need to continue it leaves the stop flag low [12][13].

V. CONCLUSION

In this paper mathematical background and hardware based implementation of Jacobi method is discussed. For implementation in FPGA, Jacobi solver is preferred over the other basic iterative solvers as its show more parallelism. In high performance computing FPGA implementation of the solvers achieved more popularity than its GPU implementation. New approaches will be raised to accelerate its performance and more advances will be developed to enhance its performance. So in this era of high performance computing FPGA have become an emerging platforms for solvers like Jacobi which are extensively used for scientific computing application.

REFERENCES

7. Ignacio Bravo*, Pedro Jiménez, Manuel Mazo, José Luis Lázaro, Alfredo Gardel, “Implementation in FPGAs of Jacobi Method to Solve the Eigenvalue and Eigenvector Problem”, 1-4244-0 312-X/06/$20.00c 2006 IEEE

**Author Details:**

**Asmita Prabhakar Wakchaure** has completed her B.Engg. in Electronics Engineering from Amrutvahini College of Engineering, Sangamner, and Pune University affiliated college in 2002. She is pursuing her M.Tech in VLSI and Embedded Systems from MRCET College Affiliated from JNTU Hydrabad Maisammaguda, Secunderabad, Telangana.

**Ms. Anitha Patibandla** working as Assistance Professor, Department of E.C.E, MRCET, Hyderabad. She received B.Tech degree from Osmania university College Of Engineering, Hyderabad and M.Tech in V.L.S.I System design from J.N.T.U.H affiliated college. Her research interests include Analog design and Low power VLSI design.