データフローアーキテクチャ電磁界解析専用計算機の開発に関する研究

Doctoral Thesis

Design Study of FIT Dedicated Computer with Dataflow Architecture for Electromagnetic Field Simulations

Wang Chenxu

Muroran Institute of Technology

September 2023

Contents

Chapter 1

Introduction

The application and development of electromagnetic field technology is the key to promote the rapid development of electronic information industry. In the design and development of electronic and electrical equipment, electromagnetic field simulation is widely used in performance optimization, EMC analysis, etc., and in the design and development of industrial products, a shorter design cycle will greatly reduce production costs, thus accelerating the development of electromagnetic fields simulation computing has been a research hotspot in recent years [1]-[4]. Because ordinary computers face large and complex calculation problems, such as topology optimization based on GA for the optimal design of electrical equipment, the calculation takes 1 to 2 days [5]-[20]. In the face of large and complex problems, high performance computing is one of the effective ways to improve computing efficiency. As one of the development results of high-performance computing, supercomputers have the latest performance exceeding 1000PFLOP/S, but they consume millions of energy and are huge in size, high in price and shared by multiple people. Therefore, as another possibility of high-performance computing, our laboratory has been committed to developing low power consumption, low cost and green dedicated computers [41]-[51].

In previous work, we developed a FDTD-specific computer for the dataflow architecture of 3D microwave simulations [47]-[50]. In this study we discuss the conceptual design of a dedicated computer for FIT dataflow based on the BiCG-Stab

solver for 2-D magneto-static field simulation [51], 3-D electrostatic field simulation and 2-D eddy current field simulation.

This thesis mainly consists of five chapters. Chapter 2 presents the background of high performance computing, and the dedicated computer for FDTD that we have proposed. The third chapter briefly summarizes the basic knowledge related to electromagnetic field theory. The fourth chapter introduces the design and simulation results of the dedicated computer used for electromagnetic field simulation calculation. Chapter 5 summarizes the conclusions of this study and future developments.

Chapter 2

High-Performance Computing (HPC)

In this Chapter, we will briefly introduce the research background and the history of HPC technology. Then, we will give an overview of dedicated computer.

2.1 Research Background and Purpose

In recent years, with the rapid development of the electronic information industry, there is a huge demand for electromagnetic simulation calculations in the electromagnetic field. The application of electromagnetic field technology in production design is in various products ranging from electric power equipment to small electronic equipment, semiconductor integrated circuits and electric vehicles [1]-[4]. On the other hand, electromagnetic fields are also widely used in the field of scientific research. For example, research on plasma heating by millimeter-wave vortex field, etc. However, for example, when optimizing the design of electrical equipment, based on GA topology or model optimization, it takes 1 to 2 days to complete the calculation of the entire optimization process by repeatedly simulating millions of different numerical models [5]-[20]. When studying the heating of the plasma by the millimeter-wave vortex field, in order to grasp the propagation characteristics of the millimeter-wave vortex field in the magnetized cooling plasma, the researchers used the FDTD method to perform a 3-dimensional simulation of the millimeter-wave vortex field. It also takes 1 to 2 days to change the parameters and re-calculate the simulation until the results are obtained [58]. Traditional

numerical analysis of electromagnetic fields is usually carried out on ordinary computers. Even though the current computer technology has developed significantly in terms of performance and storage capacity, it still takes hours to days to process large-scale numerical calculations of electromagnetic fields. Especially with the development of semiconductor products and small electronic devices to higher frequencies, the behavior of electromagnetic fields has become more and more complex, and the scale of calculations has also become larger. In product design for industrial applications, a shorter design cycle is critical to lower overall cost. Therefore, in the past ten years, many studies have focused on the development of high-performance computing technology [21]-[57].

Supercomputer is one of the development results of high-performance computing technology. A supercomputer based on a CPU system has huge data storage capacity and extremely fast data processing speed. The latest calculation peak value exceeds 1000 PFlop/s. However, in the latest performance of global supercomputers, we can find from that the top five machines with the highest performance in the world all have a huge size, a computer system with more than millions of cores, and an energy consumption of more than one million. In addition, supercomputers are shared by many people at the same time, and usually the maximum performance of a supercomputer cannot be occupied by one user. Such expensive, bulky supercomputers are not suitable for product design in industry. Therefore, we consider another possibility to realize high-performance computing, that is dedicated computers.

Dedicated computers are not von Neumann computers with strong versatility. The limitations of von Neumann computer hardware configuration are avoided and also eliminate versatility. And the dedicated computers only perform high-speed calculations for certain problems. In recent years, with the advancement of LSI (Large Scale

Integrated) technology, configurable LSIs and FPGAs have become popular, and even now FPGAs have up to one million LEs, making low-cost LSI development possible. Therefore, for developments of portable, low power consumption, low cost and high performance computing technologies to be effectively used in industry applications, our laboratory has been investigating a method of dedicated computers for electromagnetic fields simulations.

2.2 The History and Summary of HPC Technology

High-performance computing plays a non-negligible role in solving major challenging problems and promoting the development of modern society and economy. Therefore, high-performance computing has been one of the research hotspots in the field of computer science and engineering applications in the past few decades. High-performance computing relies on efforts in multiple fields, including parallel algorithms, supercomputers, etc. The demand for supercomputers also promotes the advancement of computer architecture and core technology. The development of supercomputers can be roughly divided into four stages. The first stage is the birth of the first supercomputer, the second stage is the Cray era, the third stage is the cluster era, and the fourth stage is the GPU and hybrid era.

The first supercomputer in the world was the CDC 6600 developed by the Control Data Corporation's CDC in 1964, and the computations focused on the floating-point operations per second (FLPOS), then the first supercomputer could do calculation reach 500 KFLOP/s up to 1 MFLOP/s. In the 1980s, the supercomputer consisting of vector processors such as cray-1 was a pioneering computer system that opened the era of

vector machines. For example, Cray X-MP and Cray-YP, they both have two or more vector processors, and shared memory for all processors. During this period, the speed of supercomputers was initially reached through two mechanisms of vector processors and shared memory multiprocessing. Vector processors are designed using a pipelined architecture that can quickly perform a single floating-point operation on large amounts of data [32]. Achieving high performance depends on the data arriving in an uninterrupted stream to the processing unit. Shared memory multithreading means that a small number (up to 8) processors can access the same memory space, and inter-process communication is performed through shared memory. In the 1990s, the era of truly efficient parallel computers had begun, but were limited by shared memory access. Memory contention is a major hurdle to speed up, as vector processors require high-speed access to memory, but multiple processors working simultaneously can contend for memory, and then slowing down access to memory. An alternative to shared memory is distributed memory, where each processor has a dedicated memory space, such as a cluster using efficient PCs

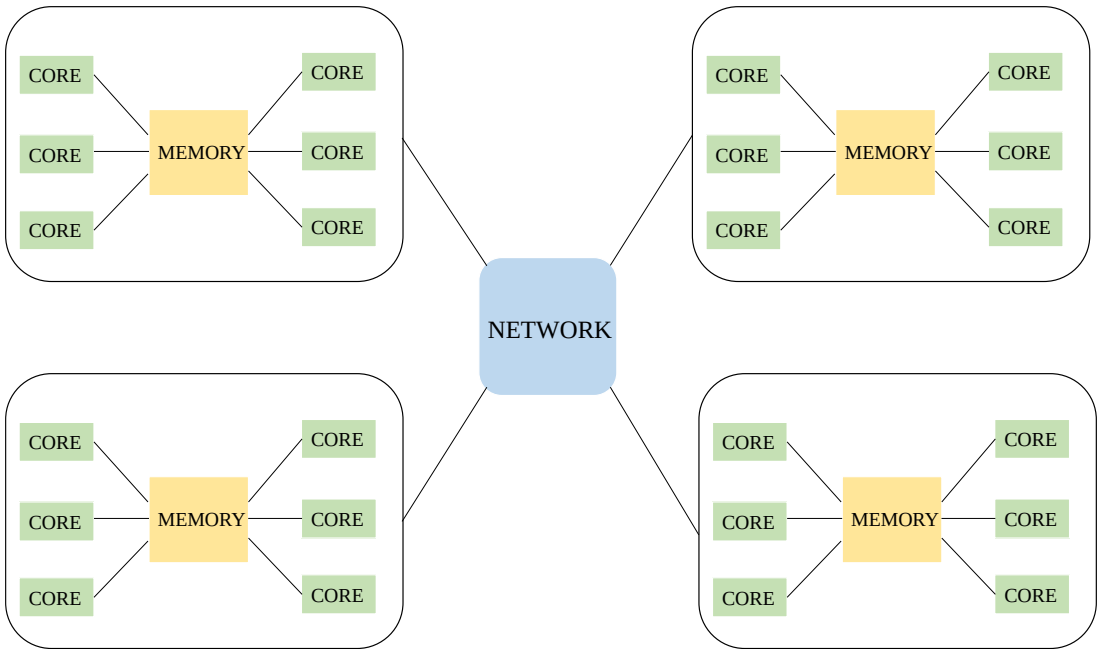


Fig. 2.2.1 Hybrid parallel computer

and network hardware, as shown in Fig.2.2.1. In the 2000s, the trend of increasing processor speed was replaced by increasing the number of processor cores. This results in a mixed cluster with a large number of processors, each with a small number of cores sharing random-access memory (RAM) and some cache space. With the development of general-purpose accelerator hardware such as GPUs, modern top supercomputers are hybrid clusters with a large number of standard processor nodes, each node has a multi-core processor with some individual caches, some shared caches and RAM shared by all cores, some GPU or other accelerators to offload specific types of computation from the CPU nodes, Computing speed is usually limited by the rate at which data is moved.

The development of high-performance computing to the present has experienced the development from parallel processors, clusters, and multi-core CPU/GPU [34]. As another possibility of HPC, the development of dedicated computers has also attracted much attention. Dedicated computers can achieve high-speed computing and large-scale computing by configuring the memory and processor to specialize in specific computing algorithms. In addition, dedicated computer is a portable HPC that anyone can use by connecting it directly to a general-purpose PC. A representative example of a dedicated computer is GRAPE, which is for simulating gravitational many-body problems [52]. In a system such as a galaxy where a large number of particles exert gravitational force on each other, the calculation of universal gravitation requires a huge amount of calculation. The computer project dedicated to gravitational calculations started in April 1989, and GRAPE-1 was completed in September of the same year. GRAPE-1 and GRAPE-2 completed in April 1990 are circuit board systems using about 100 commercial LSIs. For GRAPE-3, completed in 1991, researchers developed a custom LSI that integrated circuits equivalent to GRAPE-1 into a single chip, and 14.4 Giga FLOP/s was achieved

by running 48 units in parallel. With the development of LSI, the subsequent development of GRAPE has also realized Tera FLOP/s. In recent years, the trend of dedicated computers is not limited to gravitational many-body problems, but to develop dedicated computers employing various algorithms. For example, GRAPE-2A [53]-[54], a dedicated computer for molecular dynamics simulation of proteins with a similar architecture to GRAPE. And the ProGrape [55], a hardware platform that uses FPGA to automatically generate and implement arbitrary computing pipelines, the high-level language PGR [56] automatically generated by the pipeline. And GRAPE-DR [57], which specializes in SIMD operations. In addition, there are various fluid simulations using finite difference methods and dedicated computers dedicated to matrix calculations.

In the context of the strong demand for HPC techniques for microwave simulations, a lot of research has been done on dedicated computers for FDTD methods. The FDTD dedicated computer was first proposed by J. R. Marek in 1992 [21], the architecture proposed at that time was to add a dedicated LSI that only performs FDTD differential calculation on the existing PC architecture, and Mur's absorbing boundary condition is to use a general floating-point arithmetic LSI, known as a Floating-Point Application Specific Processor (FPASP), speeds up calculations by using dedicated hardware. The architecture was confirmed to work via VHDL simulations, which showed that it was overall 4.9 times faster than current workstations. However, due to the huge cost of creating LSI at that time, research on FDTD dedicated computers did not progress until 2002. In 2000, FPGA became popular, and it became possible for individuals to develop dedicated LSI, which led to the re-study of FDTD dedicated computers. The realization of the first FDTD dedicated computer on FPGA was completed by R. N Schneider in 2002 [22]-[23]. In this work, bit-serial pipelined arithmetic is introduced to reduce

arithmetic logic and increase parallelism. At the same time, Kawaguchi et al. proposed a hardware configuration and an overall control method for dealing with metal boundary conditions, and implemented a dedicated computer for FDTD by using TTL [26]. In the same year, L.Verducci developed a method using floating point arithmetic and VHDL simulation [24]-[25]. However, it is several times slower than a general PC due to lack of pipeline and cache memory. In 2003 Matsuoko realized a 2-D FDTD dataflow dedicated computer on FPGA, a fast architecture that can compute the entire field in parallel [28]. This architecture requires too large hardware scale and with current hardware technology, it is still difficult to realize the analysis domain required for practical computing. In 2004, Wang Chen implemented the two-dimensional FDTD method on FPGA, and equipped with a register suitable for the FDTD method [29]. In this approach, cache memory is used to speed up memory access and increase computation speed. And then, J. P. Durbano also introduced a similar caching method and implemented a three-dimensional FDTD method on FPGA [30]. In the same year, R. N. Schneider conducted research independently using software on a general-purpose PC together with a dedicated computer [31]. In this study, only simple FDTD differential calculations were performed on a dedicated computer, while complex processing such as PML absorbing boundary conditions and input wave sources were performed on a PC. In 2005, Suzuki from NTT introduced a method for partitioning the analysis domain into independent computations on a dedicated computer [32]. In 2007, J. P. Durbano carried out the FPGA development of a dedicated computer supporting various input wave sources [33]. In 2008, Kawaguchi et al. improved the dataflow architecture and developed an architecture with distributed memory to minimize bottlenecks due to memory access [34]-[35]. In 2009, Endo developed an architecture for parallel computing not only in space but also in time [39]

in 1-D FDTD methods. In addition, Sando developed a 2-D FDTD architecture [37] using multiple FPGAs. In the same year, Fujita realized a fully customized dedicated computer using FPGA and printed circuit board [36]. In 2010, Fujita used SDRAM exclusively for FDTD [42]-[44]. In 2011, Fujita developed a parallel computing system [45]-[46]. By using several dedicated computers with SDRAM, the communication bottleneck in the parallel computing of the FDTD method was hidden, and the introduction of Conformal Scheme was also studied to expand the scope of application. As mentioned above, previous research results showed that the calculation speed of the FDTD method can be faster than that of a general-purpose PC through the development of a dedicated computer, therefore in this study, we devote to the development of a dedicated computer for the FIT method based on dataflow architecture.

2.3 The Dedicated Computer

2.3.1 Dataflow Architecture

Computer architecture is the properties of the computer as seen by the programmer, i.e. the logical structure and functional characteristics of the computer. There are many types of computers in use today, but their architecture is almost the same, that is the von-Neumann architecture. The von Neumann architecture is the image of a computer conceived by von Neumann in the 1940s. In the von Neumann architecture, the program is treated as data, and the program is compiled into data, and then stored in the memory together with the data, so that the computer can call the program in the memory to process the data. This means that no matter what the program is, it will eventually be converted into data and stored in memory. To execute the corresponding program, it is only

necessary to sequentially fetch instructions from the memory and execute them. It should be known that in the early computer design, the data was stored in the memory, and the program was used as a part of the controller. Such a computer had low computational efficiency and poor flexibility. Then the von Neumann architecture eliminates the situation in the original computer system that can only rely on hardware to control the program, it stores the program code in the memory, realizes programmable computer functions, and realizes the separation of hardware design and program design. The flexibility to change a program for various purposes, and the operational clarity to fetch machine instructions one by one and perform simple processing. The core design ideas of the von Neumann architecture are as follows:

1. The final forms of programs and data are binary codes. Programs and data are stored in memory in binary form, and binary codes are also the codes that computers can recognize and execute.

2. Programs, data and instruction sequences are stored in the main memory in advance, so that the computer can accurately extract instructions from the memory and analyze and execute them when it is working.

3. Determine the five basic components of the computer, the arithmetic unit, the controller, the memory, the input device, and the output device, as shown in the Fig.2.3.1 below.

The main component of the arithmetic unit is the arithmetic logic unit (ALU), and the processing object is data. The main function is to complete arithmetic operations, logic operations and other operations under the action of control signals. The working process of the ALU is to read the data to be operated from the memory, and write the result processed by the arithmetic unit into the memory. The controller is the control unit, which is the command center of the computer. Under the control of the controller, the computer

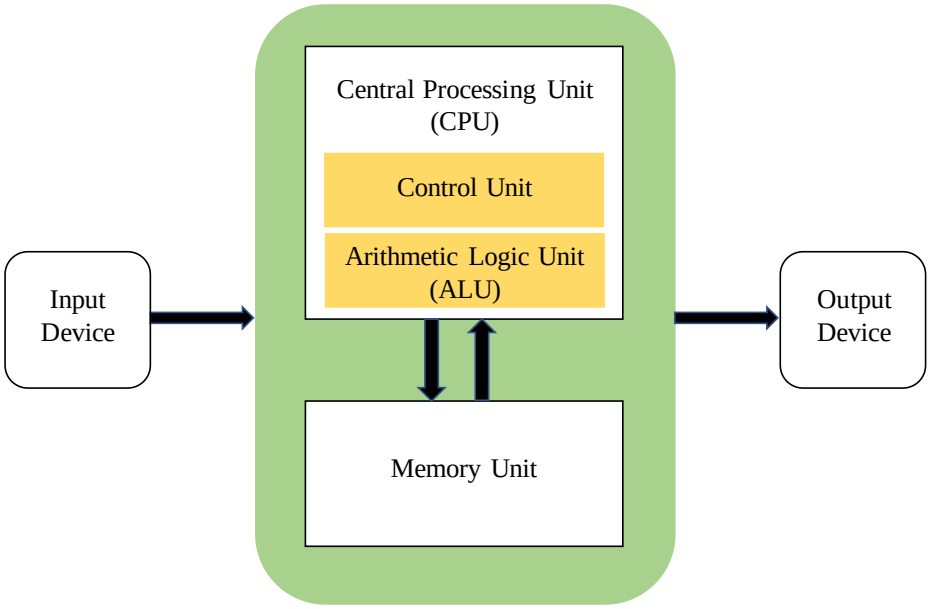


Fig.2.3.1 von Neumann architecture

automatically executes the program. The working process of the controller is to read instructions from the memory, perform translation and analysis, and then send control commands according to the instructions to control the relevant components to execute the operations contained in the instructions. The controller and the ALU together form the central processing unit (CPU).

The main function of the memory is to store programs and various data, and it can automatically complete the storage of programs or data at high speed during the operation of the computer. Memory is divided into internal memory and external memory. The internal memory is used to store the program and data that to be executed. The CPU can directly use instructions to read and write the internal memory according to the address. “Read” is to read the content of a storage unit in the memory and send it to a register of the CPU. “Write” means that under the control of the controller, the contents of a certain register in the CPU are transferred to a certain storage unit. The external memory mainly stores programs and data that are not needed "temporarily", and it can exchange data with

the memory, which is usually hard disk, disk, etc. The mouse and keyboard are the input devices, and the monitor is the output device.

Since it is based on reducing the cost of hardware, the solution to all the problems seems to be overly imposed on software. As the applications to deal with become more and more complex, the semantic gap between them and the nearly invariant machine instructions increases. Furthermore, there are no mechanisms incorporated into computer architectures to detect as many programming errors as possible, thus making it extremely difficult to develop large software. When we look at computers from this perspective, there are many problems with von Neumann architecture computers. Therefore, if we design a computer architecture that meets the software requirements and takes full advantage of the current advanced device technology, we should be able to achieve higher performance computers. As a new type of computer that breaks away from the traditional von Neumann computer framework, non-von Neumann computer is researched to meet this expectation. The immediate triggers for active research in this direction are the emergence of new machine models represented by dataflow machines [42], and the proposal of new programming styles represented by functional and logical types.

The research on dataflow methods began in the 1960s. At that time, the research on parallel computers was actively underway, and how to extract parallelism from programs became a problem. In 1974, Dennis proposed a data flow model as a computing model that can easily describe parallel processing, and then proposed a computer architecture to realize the model, and generalized it as a dataflow machine. Although dataflow machines are based on data-driven principles, they have the potential to maximize program parallelism and possess the advantage of being easy to program. One of the problems with parallel computer research conducted in the 1960s is the complexity of programming

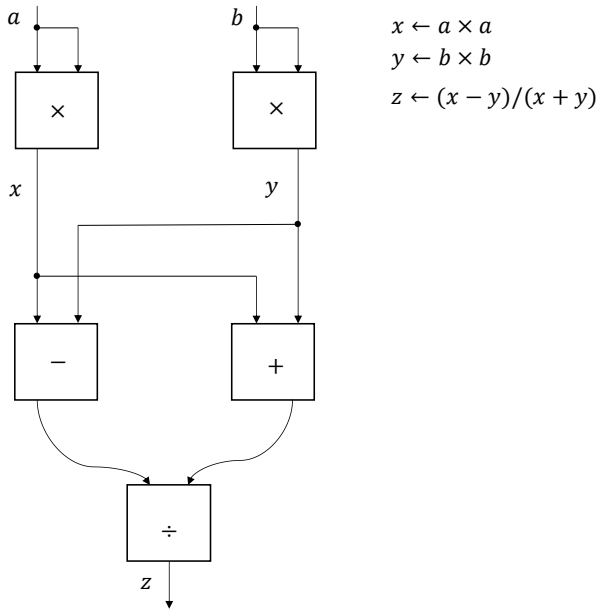


Fig.2.3.2 The dataflow graph

parallel computers. In contrast, a feature of dataflow machine research is that it arises from the research in parallel programming. Therefore, the programming and machines have been closely related since the beginning, which makes this research relevant. Several programming languages and some basic prototypes of dataflow machines were produced in the late 1970s. Then, as the research progressed, the problems with the initially simple data-driven approach became apparent. Therefore, further research refinements were carried out, such as incorporating request-driven approaches, introducing colored markers to deal with repetition and recursion, and providing storage for structured data. By the 1980s, the prototypes of comprehensive dataflow machines were developed, including not only hardware simulators, but also the LSIs that processed signals using the dataflow method. Currently, dataflow machines are used in various fields ranging from numerical computation, such as the numerical solution of partial differential equations, to non-numerical processing, such as list processing and inference processing, image processing, signal processing, communication exchange processing, etc.

The most common way to express the principle of data flow action is a directed graph, which is called a dataflow graph, as shown in Fig.2.3.2 where the process of calculating $(a^2 - b^2)/(a^2 + b^2)$ is illustrated. Firstly, the numbers a and b are entered from the above. Then, x and y are calculated to obtain z. However, subtraction can only be performed after both x and y are obtained. The final division can be performed after the results of addition and subtraction are obtained. Each operation can start when all its inputs are available, regardless of the order relationship with other operations, which allows many operations to be executed in parallel. In a data flow diagram, a, b, x, and y are all the names given to the lines in the diagram, rather than the names of the variables. The data flowing on each line is called a token. Thus, the operation $x \leftarrow a \times a$ begins when a token enters a and yields a token on x. The example presented in Fig.2.3.2 shows block structures in programming languages. As shown in Fig.2.3.3, this is represented by a switch node. The example used here is to set the value of x to $a + b$ if $a > b$, otherwise to $a \times a$. When the token reaches a and b, an operation is started to check

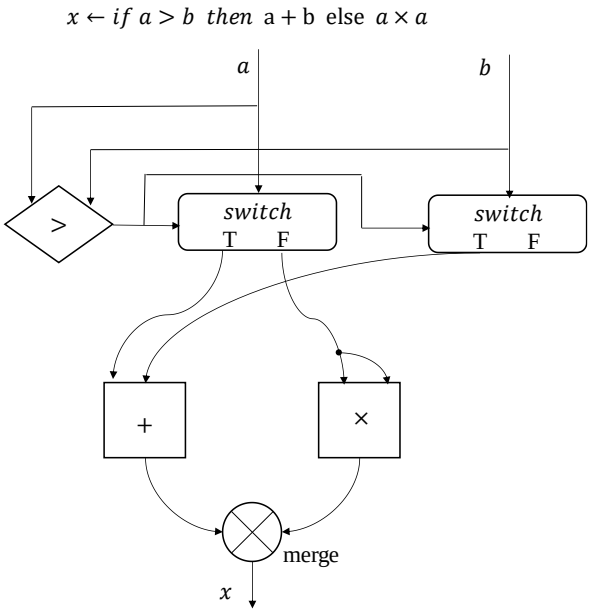


Fig.2.3.3 The example of condition judgment

whether the value on the left is large or small, and the result of this judgment is sent to the switch node on the right. For example, in case of $a > b$, a token with a value of True is sent, but the switch node passes the incoming token from the top to the bottom left or bottom right according to T, F from the token on the left. The bottom represents an operation that merges the tokens from both directions into one, and then it runs down a line. The above describes how the dataflow works.

2.3.2 3-D FDTD Dedicated Computer for Microwave Simulation

Our laboratory has been working on the development of dataflow machine, and we have developed a 3-D FDTD dataflow machine for microwave simulations [19]-[23]. In this section, the design of 3-D FDTD dataflow machine is explained in detail [20].

Fig.2.3.4 shows the circuit connection of 3-D FDTD dataflow machine. The digital circuit is configured to perform the calculation of the FDTD scheme. All the physical quantities of the electromagnetic fields are stored in registers, and these registers are directly connected through arithmetic circuits, so as to automatically execute the

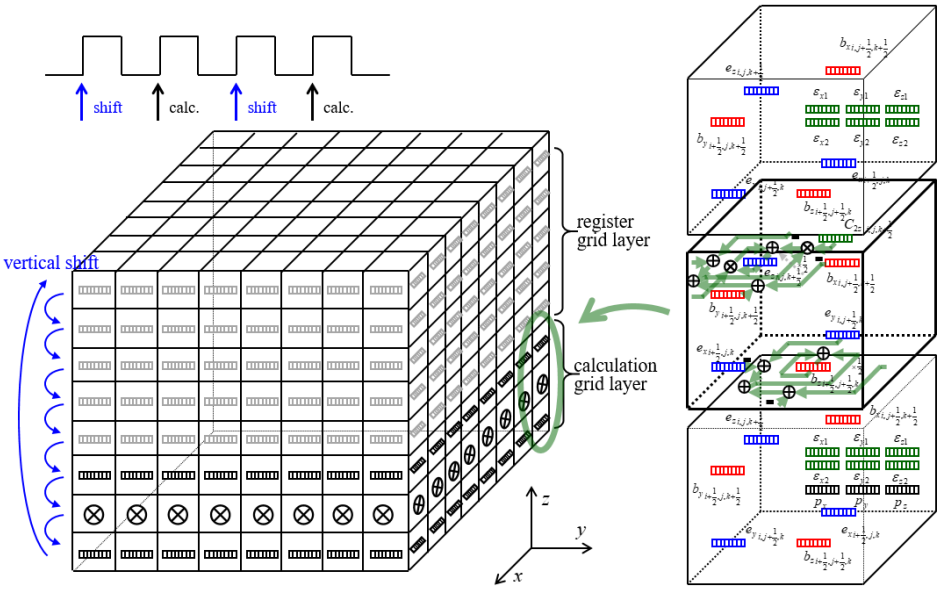


Fig.2.3.4 The circuit connection based on dataflow architecture

calculation of the FDTD scheme. In addition, a “Sliced 3-D grid circuit” is adopted to perform all 3-D grids circuit as shown in Fig.2.3.4. After the execution of circuit operation of the FDTD scheme in the middle layer of calculation grid layer located at the bottom of 3-D grids structure, the registered data in all region are shifted down by one layer. To repeat this process for all the vertical layers in both electric and magnetic fields, one time step FDTD calculation for the entire grid space is executed by $4\times z$ clock cycles. The proposed 3-D FDTD dataflow machine for microwave simulations is designed by the VHDL and it has been confirmed that the logic simulation can be carried out correctly.

Chapter 3

Fundamental Theories and Technologies

In Chapter 2, it has been described that the high-performance computing of the electromagnetic field is essential for the design of dataflow machine. In this Chapter, briefly introduces some fundamental theories required for numerical simulations, such as Poisson's equation, Finite Difference Methods (FDM), Finite Integration Technique (FIT), BiCG-Stab scheme and logic circuits.

3.1 Electromagnetic Field

Electromagnetic fields are the combination of invisible electric and magnetic forces. The electric field that changes over time generates a magnetic field, and the magnetic field that changes over time also generates an electric field. The two are interdependent of and transformed into each other to create an electromagnetic field. Caused by various natural phenomena such as the earth's magnetic field, they can also be artificially generated, such as mobile phones and computer screens which are the examples of devices that generate electromagnetic fields. Electromagnetic fields can be generated by the charged particles moving at varying speed, or by the currents of varying strength. No matter what the reason is, electromagnetic fields always propagate around at the speed of light to form electromagnetic waves. Electromagnetic waves are the electromagnetic fields that propagate in the form of waves, which allows them to penetrate substances and carry information. Therefore, they are widely used in various fields.

3.1.1 Electro-static Field

A special form of matter exists around an electric charge called an electric field. An electric field is an aspect of a unified electromagnetic field that behaves as a forceful action on a static charge introduced into the field. The electric field caused by a charge that is stationary relative to the observer and whose charge does not change with time is called an electrostatic field. Electrostatic fields widely exist in daily life, such as electrostatic adsorption, charge accumulation and other phenomena are related to electrostatic fields. Electrostatic fields are also widely used in industry and scientific research, such as electron beam processing, plasma physics and other fields.

The introduction of the electrostatic field is inseparable from the two most important field quantities, the electric field intensity **E** and the scalar electric potential $\emptyset$. Starting from Coulomb's law, on the basis of analyzing the electrostatic field in vacuum, the influence of conductors and dielectrics on the electric field is discussed separately, and a basic field quantity of the electrostatic field, electric field intensity **E** is introduced. Based on the application of vector analysis to clarify that the electrostatic field has irrotational properties, another important field quantity of the electrostatic field, the scalar electric potential $\emptyset$ is introduced. In addition, based on the study of the closed area integral of the electric field intensity vector, the electric flux density (also known as the electric displacement) **D** is introduced, and Gauss's law is derived. Together with the irrotational property of the electrostatic field, it constitutes the basic equation of the integral form of the electrostatic field. Finally, the basic equation of the differential form of the electrostatic field is applied to derive the Poisson’s equation that the scalar electric potential $\emptyset$ satisfies. In the case of electrostatic fields, Poisson's equation can be used to solve for the electric potential distribution and electric field intensity to better understand

and describe the nature and behavior of electrostatic fields.

3.1.1.1 Coulomb's Law

Coulomb's law is to describe the interaction between charges, and it is one of the basic laws describing the electrostatic field. A French scholar named Coulomb concluded after doing a series of delicate electrostatic experiments: In an infinite vacuum, when the distance between two static small charged bodies is much larger than their own geometric size. We can consider the simple case described by figure 3.1.1. Let F denote the force acting on an electrically charged particle with charge q_2 located at x, due to the presence of a charge q located at $\mathbf{x_1}$. According to Coulomb's law this force is, in vacuum, given by the expression:

$$\mathbf{F} = \frac{q_1 q_2}{4\pi\varepsilon_0} \cdot \frac{\mathbf{x} - \mathbf{x_1}}{|\mathbf{x} - \mathbf{x_1}|^3} \tag{3.1.1}$$

where, $\varepsilon_0 = 8.854 \times 10^{-12}$(F/m), is the dielectric constant in vacuum.

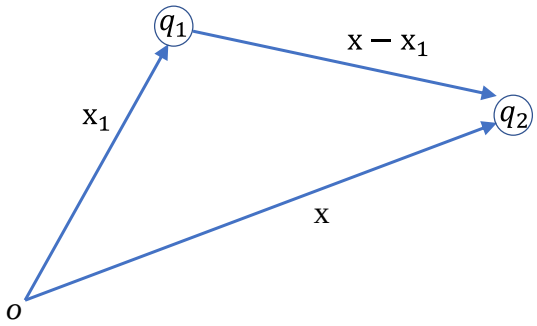


Fig. 3.1.1 Coulomb's law for a distribution of individual charges q_1.

Coulomb's law gives the magnitude and direction of the force between two charges, and the force between charges is transmitted at a finite speed through a special substance in the surrounding space, that is the electric field. Any charge creates an electric field in the space around it. An important characteristic of the electric field is that it exerts a force on any other charge in it, so the physical quantity, electric field intensity **E** is introduced

to describe this important characteristic of the electric field. The static vector electric field **E** can be defined by limiting process:

$$\mathbf{E} = \lim_{q_0 \to 0} \frac{\mathbf{F}}{q_0} \tag{3.1.2}$$

where, F is the electrostatic force, as defined in equation (3.1.1), q_0 is test particle with positive. The electric field **E** is a vector function that varies with the position of a point in space, and is only related to the electric field at the point, but has nothing to do with the charge amount of the test charge.

According to the definition of electric field intensity and Coulomb's law, if the coordinate of point charge q_1 is $\mathbf{x}_1$, the electric field intensity **E**(**x**) caused by it at **x** is:

$$\mathbf{E}(\mathbf{x}) = \frac{q_1}{4\pi\varepsilon_0} \cdot \frac{\mathbf{x} - \mathbf{x_1}}{|\mathbf{x} - \mathbf{x_1}|^3} \tag{3.1.3}$$

3.1.1.2 Conservative Field and Electro-static Potential

If there is a unit positive test charge q moves from point A to point B along a certain path ***l*** in the electrostatic field, as shown in the figure, the work done by the electric field force:

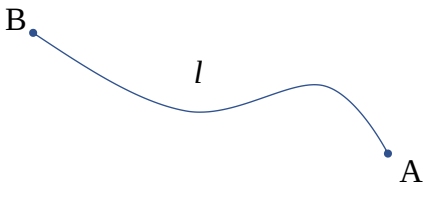


Fig. 3.1.2 Charge q moves from point A to point B along path l

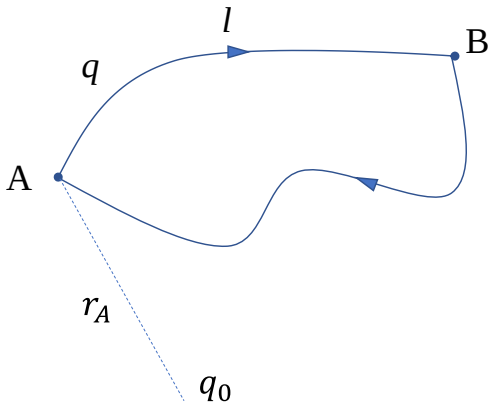


Fig. 3.1.3 Charge q moves along a closed path

$$W = \int_A^B \mathbf{E} \cdot d\boldsymbol{l} \tag{3.1.4}$$

If the test charge q moves from point A to point B and back to point A along the closed path $\boldsymbol{l}$ in the electrostatic field, the work done by the electric field force:

$$W = \int_A^B \mathbf{E} \cdot d\boldsymbol{l} = \frac{q_0}{4\pi\varepsilon_0}\int_{r_A}^{r_A} \frac{1}{r^2} dr = \frac{q_0}{4\pi\varepsilon_0}\left(\frac{1}{r^2} - \frac{1}{r^2}\right) = 0 \tag{3.1.5}$$

That is, in an electrostatic field, moving charges along a closed path, the work done by the electric field force is zero. usually written as follows:

$$\oint \mathbf{E} \cdot d\boldsymbol{l} = 0 \tag{3.1.6}$$

this is an important property of the electrostatic field, known as the loop law. Applying Stokes' theorem to Equation (3.1.5), then

$$\oint \mathbf{E} \cdot d\boldsymbol{l} = \oint \nabla \times \mathbf{E} \cdot d\mathbf{S} = 0 \tag{3.1.7}$$

and the differential form of the above equation can be written as:

$$\nabla \times \mathbf{E} = \mathbf{0} \tag{3.1.8}$$

The above formula shows that the curl of the electric field **E** of the electrostatic field is 0 everywhere. According to the vector analysis, the electric field **E** of the electrostatic field can be expressed by the gradient of a scalar function $\emptyset$ as follows:

$$\mathbf{E} = -\nabla\emptyset \tag{3.1.9}$$

this scalar function $\emptyset$ is called the scalar potential of the electrostatic field. It is another physical quantity that characterizes the characteristics of the electrostatic field.

3.1.1.3 Gauss's Law

Gauss's law is a fundamental law describing the electric field. On any closed surface S in an infinite vacuum electrostatic field, the area integral of the electric field intensity **E** is equal to $\frac{1}{\varepsilon_0}$ times the total charge $q = \int \rho dV$ inside the surface (V is the volume

limited by S), regardless of the charge outside the surface. It can be represented by the following formula:

$$\oint \mathbf{E} \cdot d\boldsymbol{S} = \frac{q}{\varepsilon_0} = \frac{1}{\varepsilon_0} \int \rho dV \tag{3.1.10}$$

the above formula is call Gauss's law for electrostatic field in vacuum.

When there is a dielectric, the electric field can be regarded as being caused by the free charge and the polarized charge in the vacuum, and the Gauss's law of the electrostatic field in the vacuum still available, but the total charge includes not only the free charge q, but also the polarized charge q_p,

$$\oint \mathbf{E} \cdot d\boldsymbol{S} = \frac{\int \rho dV + q_p}{\varepsilon_0} = \frac{q + q_p}{\varepsilon_0} \tag{3.1.11}$$

in the above formula, q and q_p are the total free charges and total polarized charges in the closed surface S, respectively. And the polarize the charge

$$q_p = \int \rho_p dV = \int -\nabla \cdot \boldsymbol{P} dV = \oint \boldsymbol{P} \cdot d\boldsymbol{S} \tag{3.1.12}$$

substituting into the Equation (3.1.11), we can obtain the follows:

$$\oint \mathbf{E} \cdot d\boldsymbol{S} = \frac{1}{\varepsilon_0} \int \rho dV - \frac{1}{\varepsilon_0} \oint \boldsymbol{P} \cdot d\boldsymbol{S} \tag{3.1.13}$$

and then,

$$\oint \mathbf{E} \cdot d\boldsymbol{S} = \frac{1}{\varepsilon_0} \int \rho dV - \frac{1}{\varepsilon_0} \oint \boldsymbol{P} \cdot d\boldsymbol{S} \tag{3.1.14}$$

To simplify the above formula, introduce a new physical quantity, electric flux density **D**, so as to obtain the general form of Gauss's law, as follows:

$$\oint \mathbf{D} \cdot d\boldsymbol{S} = \int \rho dV \tag{3.1.15}$$

it points out that whether in a vacuum or in a dielectric, the area integral of the electric flux density **D** on any closed surface S is equal to the total free charges in the surface, regardless of all polarized charges and free charges outside the surface.

Applying the Gaussian divergence theorem to the above formula:

$$\nabla \cdot \mathbf{D} = \rho \tag{3.1.16}$$

this is the differential form of Gauss' law. Gauss's law is very important in electrostatics and can be used to solve many electric field problems of charge distribution, such as spherical and planar symmetric charge distribution, etc.

3.1.1.4 Poisson's Equation

In the previous two sections, we have obtained the following two sets of basic equations:

$$\oint \mathbf{D} \cdot d\mathbf{S} = \int \rho dV \tag{3.1.17}$$

$$\oint \mathbf{E} \cdot d\boldsymbol{l} = 0 \tag{3.1.18}$$

and

$$\nabla \cdot \mathbf{D} = \rho \tag{3.1.19}$$

$$\nabla \times \mathbf{E} = \mathbf{0} \tag{3.1.20}$$

there is a constituent equation:

$$\mathbf{D} = \varepsilon_0 \mathbf{E} \tag{3.1.21}$$

(3.1.17) and (3.1.18) are expressed in integral form, which is called the fundamental equation of electrostatic field in integral form. (3.1.19) and (3.1.20) are called the fundamental equations of the electrostatic field in differential form.

The integral form of Gauss's law (3.1.17) shows that the closed area integral of the electric flux density **D** is equal to the total free charges enclosed in the surface, which characterizes a basic property of the electrostatic field. The loop characteristic of the electrostatic field (3.1.18) shows that the integral of the loop line of the electric field intensity **E** is always equal to zero, that is, the electrostatic field is a conserved field.

Although (3.1.18) is derived from the electric field in vacuum, it still available when there is a dielectric. This is because when there is a medium, the additional effect can be considered with the polarized charge. As far as the electric field is generated, the polarized charge, same as the free charge, obeys Coulomb's inverse square law, and the resulting electrostatic field belongs to the conservation field. The differential form of Gauss's law (3.1.19) indicates that the electrostatic field has a discrete field. Equation (3.1.20) is the differential form of the loop characteristic of the electrostatic field, which shows that the electrostatic field is an irrotational field. From a physical concept, the integral form describes the overall situation of each loop and each closed surface field quantity. The differential form describes the field quantity of each point and its neighborhood, and also reflects the change of the field quantity from one point to another, so that a further understanding of the field distribution can be obtained.

In the Gauss's law $\nabla \times \mathbf{D} = \rho$, substitute $\mathbf{D} = \varepsilon \mathbf{E}$ and $\mathbf{E} = -\nabla \emptyset$, then we can obtain:

$$\nabla \cdot \varepsilon(-\nabla \emptyset) = \rho \tag{3.1.22}$$

for a homogeneous dielectric, ε is a constant, then we can obtain:

$$\nabla^2 \emptyset = -\rho / \varepsilon_0 \tag{3.1.23}$$

this is Poisson's equation for electric potential $\emptyset$. In the region of free charge body density $\rho = 0$, the above formula can be written as:

$$\nabla^2 \emptyset = 0 \tag{3.1.24}$$

this is Laplace's equation for electric potential $\emptyset$.

Poisson's equation and Laplace's equation express the general relationship between the spatial variation of potential at each point in the field and the free charge density at that point, which is the differential equation that the potential function should satisfy. The solution of all electrostatic field problems can be attributed to the process of seeking the

solution of Poisson's equation or Laplace's equation under certain conditions. And then, the conversion relationship between the main theorems of electrostatic field is shown in the Fig.3.1.4.

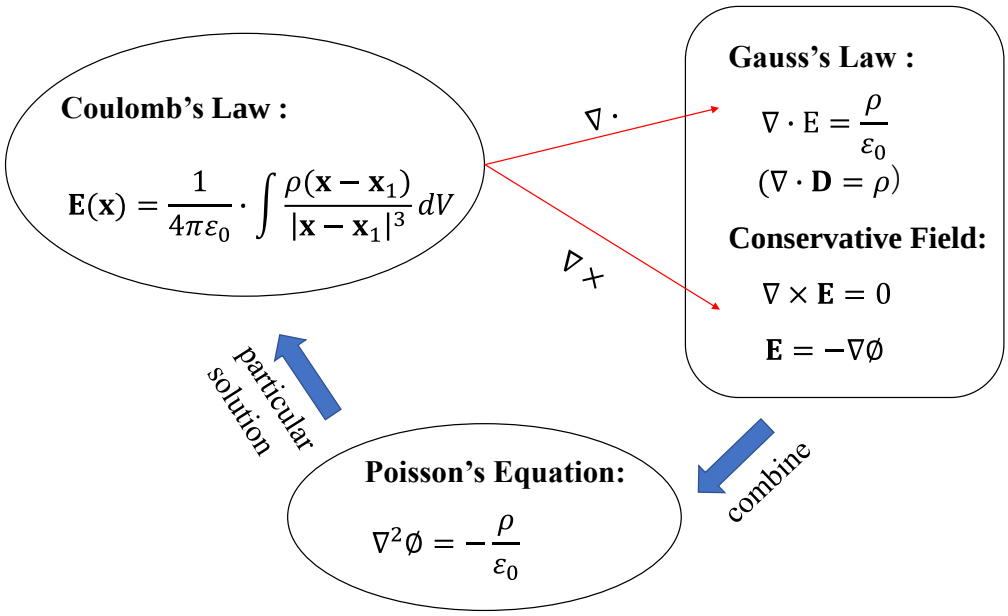


Fig. 3.1.4 The conversion of main theorems of electrostatic field

3.1.2 Magneto-static Field

In the previous section we explained the electrostatic field, and in this section we explain the magneto-static field. Around the current, there is not only an electric field but also a magnetic field. The magnetic field is another aspect of the unified electromagnetic field, which manifests itself as a forceful action on the moving charges, which is introduced into the field, that is, the charged conductor is subjected to the force of the magnetic field in the field. Relative to the observer, the current generated by the electric charge that moves in a straight line or has a constant speed is called a constant current, and the magnetic field caused by the constant current is a magneto-static field. That is, a stationary charge causes an electrostatic field, and a constant current causes a magneto-

static field. The magneto-static field is a fundamental concept in many physics and engineering disciplines, including electrodynamics, magnetic resonance imaging, electromagnetic induction, etc.

First, the most important field vector in the magneto-static field, the magnetic induction B, is introduced. On the basis of analyzing the magnetic field in vacuum, the performance of the magnetically permeable medium in the magneto-static field is discussed, and the additional effect of the magnetizing current after magnetization is considered, and then the magnetization vector M is introduced. On the basis of studying the loop integral of magnetic induction in vacuum and magnetically permeable medium, the magnetic field intensity vector **H** is introduced, and Ampere's loop law ($\oint \mathbf{H} \cdot d\boldsymbol{l} = I$) is derived, together with the principle of continuity of magnetic flux ($\oint \mathbf{B} \cdot d\mathbf{S} = 0$), it constitutes the basic equation (integral form) of the magneto-static field. According to the basic equation in integral form, we can obtain the basic equation in differential form ($\nabla \cdot \mathbf{B} = 0$, and $\nabla \times \mathbf{H} = \boldsymbol{J}$), and introduce the magnetic vector potential **A** according to the basic equation in differential form, then derive Poisson's equation ($\nabla^2 \mathbf{A} = -\mu \boldsymbol{J}$).

3.1.2.1 Biot-Savart Law

There are two loops composed of thin wires in vacuum, l and l'. And then the constant currents *I* and $I'$are passed through these two thin wires respectively. In the two circuits, select the element current $I'd\boldsymbol{l}'$ and $Id\boldsymbol{l}$, the direction of current $d\boldsymbol{l}'$and $d\boldsymbol{l}$, corresponds to the direction of flow of I' and I respectively, as shown in the Fig.3.1.5. $\boldsymbol{r}'$ and r are the position vectors of the element current, and R= r-$\boldsymbol{r}'$ is their relative position vector. The force of the current loop l' on the current loop l is measured by experiments.

$$\mathbf{F} = \frac{\mu_0}{4\pi}\oint_l \oint_{l'} \frac{Id\boldsymbol{l} \times (I'd\boldsymbol{l}' \times \boldsymbol{e}_R)}{R^2} \tag{3.1.25}$$

the above formula is Ampere's law in vacuum, which gives the force between two current loops. where μ_0 is the magnetic permeability in vacuum, and $\mu_0 = 4\pi \times 10^{-7}\mathrm{H}/m$.

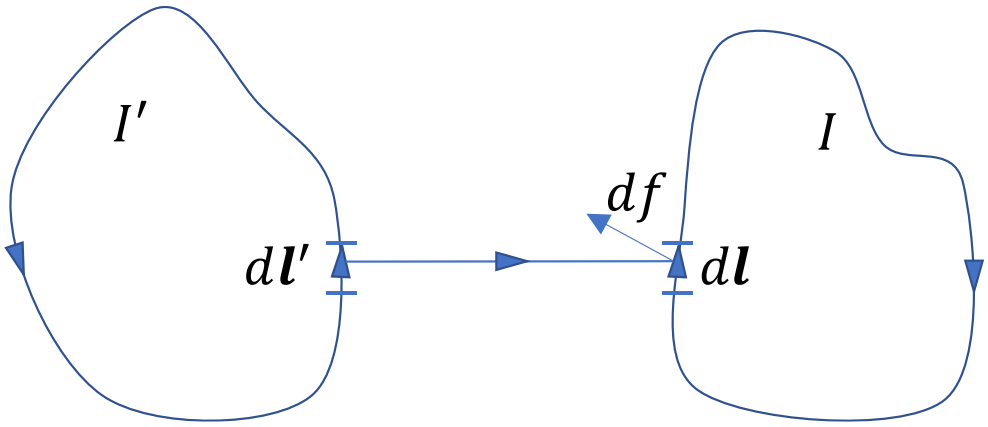


Fig. 3.1.5 The two currents loop

The equation (3.1.25) can be rewritten as:

$$\mathbf{F} = \oint_l Id\boldsymbol{l} \times \left(\frac{\mu_0}{4\pi}\oint_{l'} \frac{I'd\boldsymbol{l}' \times \boldsymbol{e}_R}{R^2}\right) \tag{3.1.26}$$

from the point of view of the field, the amount in the parentheses of the above equation (3.1.26) represents the effect of the current l'at $Id\boldsymbol{l}$, expressed by **B**:

$$\mathbf{B} = \frac{\mu_0}{4\pi}\oint_{l'} \frac{I'd\boldsymbol{l}' \times \boldsymbol{e}_R}{R^2} \tag{3.1.27}$$

the above formula is called the Biot-Savart Law. **B** is called the magnetic induction intensity, which is the basic field quantity that characterizes the characteristics of the magnetic field, and its unit is T (Tesla). In addition, the direction of the magnetic field is always perpendicular to the line of element and position vector in a plane. It is given by the right-hand thumb rule where the thumb points is to the direction of conventional current and the other fingers show the direction of magnetic field, as shown in Fig.3.1.6.

In addition to $Id\boldsymbol{l}$, there are also $\boldsymbol{J}$dV and $\boldsymbol{K}$dS in the element current section.

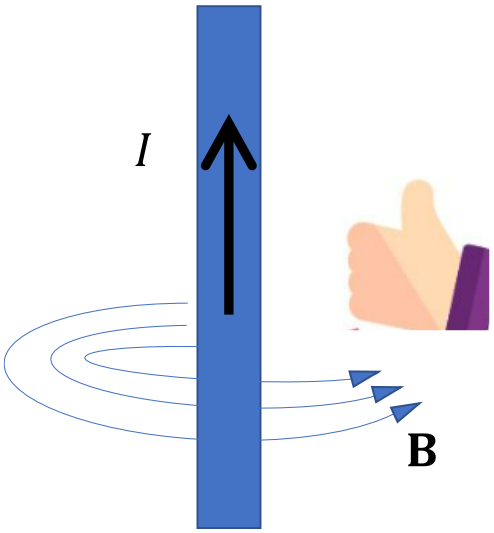


Fig. 3.1.6 The direction of the magnetic filed

Correspondingly, the Biot-Savart Law can also be written as:

$$\mathbf{B}(x,y,z)=\frac{\mu_0}{4\pi}\oint_{V'}\frac{\boldsymbol{J}(x',y',z')\times\boldsymbol{e}_R}{R^2}dV' \tag{3.1.28}$$

and

$$\mathbf{B}(x,y,z)=\frac{\mu_0}{4\pi}\oint_{S'}\frac{\boldsymbol{K}(x',y',z')\times\boldsymbol{e}_R}{R^2}dS' \tag{3.1.29}$$

If there is a line current loop with current intensity I in the magnetic field, then the force of the magnetic field on the current loop can be written as:

$$\mathbf{F}=\oint_l Id\boldsymbol{l}\times\mathbf{B} \tag{3.1.30}$$

this is the general form of Ampere's law. If there is a charge q moving at a speed $\boldsymbol{v}$ in a magnetic field, then the force of the magnetic field on it is the force of the magnetic field acting on the moving charge, also known as the Lorentz force:

$$\mathbf{F}=q\boldsymbol{v}\times\mathbf{B} \tag{3.1.31}$$

It can be seen from the above formula that the static charge will not be subjected to the force of the magnetic field in the magnetic field, and the force on the moving charge is

always perpendicular to the speed of motion. It can only change the direction of the speed, but cannot change the magnitude of the speed. Therefore, there is a different with the Coulomb force, the Lorentz force does not work.

Same as the E line in the electrostatic field, the B line can also be used in the magneto-static field. The differential Equation for B:

$$\mathbf{B} \times d\boldsymbol{l} = \mathbf{0} \tag{3.1.32}$$

3.1.2.2 Ampere's Law

If the magnetic field is caused by an infinitely long straight wire carrying current I in the vacuum, and according to the previous section we explained, the magnetic induction intensity $B = \mu_0 I/2\pi\rho$ at a distance from the wire ρ is known. Take a closed loop l as the integration path in any plane perpendicular to the wire, as shown in Fig.3.1.7. The element length dl on the integral path, the distance to the wire is ρ, the angle to the axis is d∅, and the angle with B is α, then $\rho \mathrm{d}\emptyset = \mathrm{d}\boldsymbol{l}\cos\alpha$. And then we can obtain:

$$\oint_l \mathbf{B} \cdot d\boldsymbol{l} = \oint_l \frac{\mu_0 I}{2\pi\rho} e_\emptyset \cdot d\boldsymbol{l} = \oint_l \frac{\mu_0 I}{2\pi\rho} \rho \mathrm{d}\emptyset = \frac{\mu_0 I}{2\pi} \int_0^{2\pi} \mathrm{d}\emptyset = \mu_0 I \tag{3.1.33}$$

If the integral circuit is not interlinked with the current, as shown in Fig.3.1.8, then because of $\int_0^0 \mathrm{d}\emptyset = 0$, thus $\oint_l \mathbf{B} \cdot d\boldsymbol{l} = 0$.

If there is more than one current interlinked by the integration path, as shown in Fig.3.1.9, obviously there should be

$$\oint_l \mathbf{B} \cdot d\boldsymbol{l} = \mu_0 (I_1 + I_2 - I_3) \tag{3.1.34}$$

To sum up, in the magnetic field of vacuum, any loop takes the line integral of B, and its value is equal to the algebraic sum of the magnetic permeability of the vacuum multiplied by the current passing through the surface limited by the loop. That is,

$$\oint_l \mathbf{B} \cdot d\boldsymbol{l} = \mu_0 \sum_{k=1}^{n} I_k \qquad (3.1.35)$$

The above formula is Ampere's loop law in vacuum. The positive or negative of the current I_k in the formula depends on whether the direction of the current and the winding direction of the integral circuit conform to the right-handed spiral relationship, and it is positive when they match, otherwise it is negative

For a symmetrical magnetic field distribution, the calculation of B can be made very simple by applying Ampere's loop law. At this time, the integration path should be properly selected so that the B and dl directions of each point on the integration path have the same angle, and the value of B is equal.

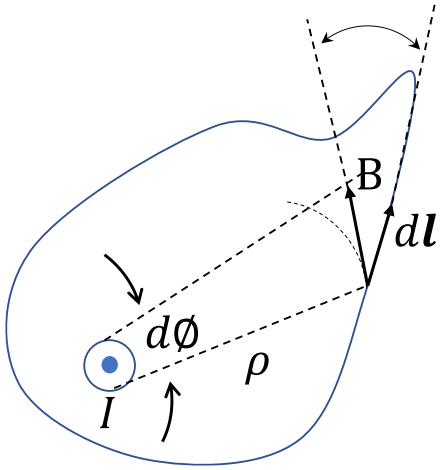


Fig. 3.1.7 Integration path

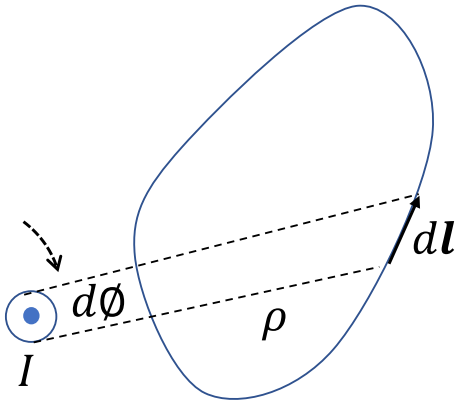


Fig. 3.1.8 Integration path without current interlinked

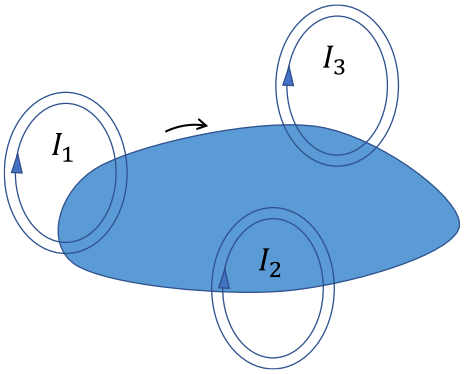


Fig. 3.1.9 Integration path with current interlinked

If in a magnetic field with a magnetically permeable medium, and Take a closed path l arbitrarily, then the line integral of the magnetic induction along this loop should be:

$$\oint_l \mathbf{B} \cdot d\boldsymbol{l} = \mu_0(I + I_m) \tag{3.1.36}$$

I in the above formula represents the free current, and I_m is the magnetizing current. The above formula can be rewritten as:

$$\oint_l \mathbf{B} \cdot d\boldsymbol{l} = \mu_0(I + \oint_l \mathbf{M} \cdot d\boldsymbol{l}) \tag{3.1.37}$$

and then,

$$\oint_l (\frac{\mathbf{B}}{\mu_0} - \mathbf{M}) \cdot d\boldsymbol{l} = I \tag{3.1.38}$$

and set,

$$\frac{\mathbf{B}}{\mu_0} - \mathbf{M} = \mathbf{H} \tag{3.1.39}$$

where, H is the magnetic field intensity, and the Equation (3.1.38) can be rewritten as:

$$\oint_l \mathbf{H} \cdot d\boldsymbol{l} = I \tag{3.1.40}$$

It should be noted that the I on the right side of the equal sign in the above formula is the free current passing through the area enclosed by the loop l, not including the magnetizing current.

If more than one free current passes through the area defined by the loop l, then

$$\oint_l \mathbf{H} \cdot d\boldsymbol{l} = \sum I_k \tag{3.1.41}$$

The Equation (3.1.38) and (3.1.41) is the general form of Ampere's Loop Law。

It shows that in the magnetic field, the line integral of the magnetic field intensity **H** along any closed path is equal to the algebraic sum of the free currents (excluding the magnetizing current) passing through the area enclosed by the loop. If the direction of the current and the winding direction of the integral circuit conform to the right-hand spiral

relationship, the current in the Equation (3.1.38) and (3.1.41) is positive. The Equation (3.1.41) shows that the loop line integral of **H** is only related to the free current, and has nothing to do with the magnetizing current, that is, it has nothing to do with the distribution of the magnetic medium. However, it cannot be understood that the distribution of **H** has nothing to do with the distribution of the magnetic medium.

For an isotropic linear medium, there is a proportional relationship between the magnetization and the magnetic field strength, that is

$$\mathbf{M}=x_m\mathbf{H} \tag{3.1.42}$$

In the above formula, x_m is called the magnetic susceptibility of the medium, which is a dimensionless pure number

According to the Equation (3.1.39) and (3.1.42), we can obtain

$$B=\mu_0(\mathbf{H}+\mathbf{M})=\mu_0(1+x_m)\mathbf{H}=\mu_0\mu_r\mathbf{H} \tag{3.1.43}$$

or

$$B=\mu\mathbf{H} \tag{3.1.44}$$

μ in the Equation (3.1.44) is the medium permeability, and $\mu_r=\frac{\mu}{\mu_0}$ is called the relative permeability, which is a pure number.

It is worth noting that the relationship shown in Equation (3.1.44) is only applicable to isotropic linear magnetically permeable media, while Equation (3.1.39) has no such limitation.

If the current surrounding the magnetic field is infinitely filled with uniform and isotropic magnetically permeable medium, the direction of the magnetic induction intensity **B** at each point in the magnetic field will be consistent with that produced when the same current is placed at the same position in an infinite vacuum, and the value of **B** at each point is increased by the same multiple, that is, increased by μ_r times. Therefore,

for the calculation of the magnetic induction intensity in this special case, it is sufficient to replace μ_0 with the permeability μ of the magnetically permeable medium.

3.1.2.3 Gauss's Law and Magnetic Vector Potential

A vector field is usually studied in two ways, namely by computing its flux through any closed surface and its line integral along any closed curve. In this section, we will obtain its basic equations by analyzing the flux and loop line integrals of a magneto-static field.

In the magnetic field, the flux of **B** passing through any surface *S* is called magnetic flux $\emptyset_m$, as shown Fig3.1.10, therefore,

$$\emptyset_m = \oint_S \mathbf{B} \cdot d\mathbf{S} \tag{3.1.45}$$

where the unit of magnetic flux is Wb.

Experiments show that the line of magnetic induction is closed, with neither beginning nor end. This shows that there is no magnetic charge in nature for the **E**-line to emit or terminate like electric charges, so there is no source for the **B**-line to emit or terminate.

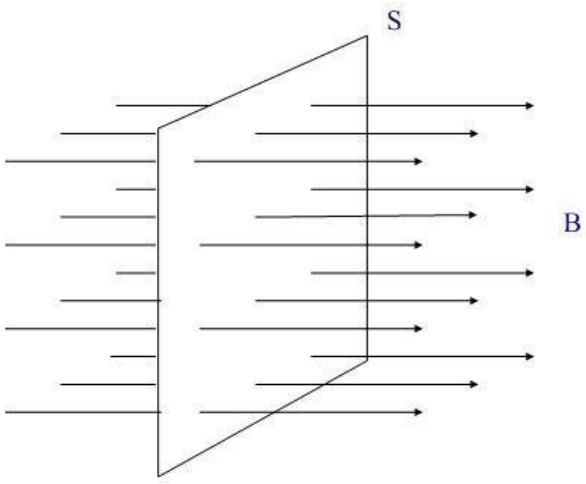


Fig. 3.1.10 The magnetic flux

Thus, for any closed surface, there is,

$$\oint_S \mathbf{B} \cdot d\mathbf{S} = 0 \tag{3.1.46}$$

The properties of the magnetic field represented by the above Equation (3.1.46) is also called the principle of magnetic flux continuity (integral form).

By using the Gauss's divergence theorem, we can obtain as follows:

$$\oint_S \mathbf{B} \cdot d\mathbf{S} = \oint_V \nabla \cdot \mathbf{B} dV = 0 \tag{3.1.47}$$

and then,

$$\nabla \cdot \mathbf{B} = \mathbf{0} \tag{3.1.48}$$

This is the differential form of the flux continuity principle, which states that a magneto-static field is a divergence-free field. If the divergence of this field is equal to zero, it is probably a constant magnetic field.

The principle of magnetic flux continuity and Ampere's Loop Law characterize the basic properties of the magneto-satic field. Regardless of the distribution of the magnetic medium, all magneto-static fields have these two characteristics. Relist their expressions here as follows:

$$\oint_S \mathbf{B} \cdot d\mathbf{S} = 0 \tag{3.1.49}$$

$$\oint \mathbf{H} \cdot d\boldsymbol{l} = I \tag{3.1.50}$$

and call them as the fundamental equations of the magneto-static field (in integral form).

Applying Stokes' theorem to the above Equation (3.1.50), and using the area integral of $\boldsymbol{J}$ to express the free current, we can obtain:

$$\oint \mathbf{H} \cdot d\boldsymbol{l} = \oint_S (\nabla \times \mathbf{H}) \cdot d\mathbf{S} = \oint_S \boldsymbol{J} \cdot d\mathbf{S} \tag{3.1.51}$$

the above Equation (3.1.51) is available for any area with l as the peripheral boundary,

and then,

$$\nabla \times \boldsymbol{H} = \boldsymbol{J} \tag{3.1.52}$$

this is the differential form of Ampere's Loop Law, and it can be seen that the magnetic field has a curl field.

Equations (3.1.48) and (3.1.52) together are called the differential form of the fundamental equation of the magneto-static field. It can be seen that the magneto-static field is a passive field with curl.

For the two field quantities **B** and **H**, we can obtain:

$$\mathrm{B} = \mu_0 \mathbf{H} + \mu_0 \mathbf{M} \tag{3.1.53}$$

For an isotropic linear medium, then

$$\mathrm{B} = \mu \mathbf{H} \tag{3.1.54}$$

In the electrostatic field, due to $\nabla \times \mathbf{E} = 0$, the potential function was introduced to characterize the characteristics of the electrostatic field, thus simplifying the analysis and calculation of the electric field. Is it possible to obtain a potential function for a magneto-static field.

Due to non-curl, the electrostatic field can be described by introducing a potential function. However, due to the non-dispersive nature of the magnetic field, a vector function **A** can be introduced to make as follows:

$$\mathrm{B} = \nabla \times \mathbf{A} \tag{3.1.55}$$

Obviously, the above Equation (3.1.55) always satisfies $\nabla \cdot \mathbf{B} = \nabla \cdot (\nabla \times \mathbf{A}) \equiv 0$. This vector function **A** is called the magnetic vector potential of the magneto-static field, also known as the vector magnetic potential. Its unit is Wb.

3.1.2.4 Vector Poisson's Equation

By the differential form of Ampere's Loop Law as follows:

$$\nabla \times \mathbf{H} = \boldsymbol{J} \tag{3.1.56}$$

at the same time, considering the **B** in the isotropic linear magnetic medium, so there is,

$$\nabla \times \mathbf{B} = \mu \boldsymbol{J} \tag{3.1.57}$$

substituting Equation (3.1.55) into the above Equation (3.1.57), we can obtain,

$$\nabla \times \nabla \times \mathbf{A} = \mu \boldsymbol{J} \tag{3.1.58}$$

applying vector identities

$$\nabla \times \nabla \times \mathbf{A} = \nabla(\nabla \cdot \mathbf{A}) - \nabla^2 \mathbf{A} \tag{3.1.59}$$

and then,

$$\nabla(\nabla \cdot \mathbf{A}) - \nabla^2 \mathbf{A} = \mu \boldsymbol{J} \tag{3.1.60}$$

In a vector field, to determine a vector, its divergence and curl must be known at the same time. The divergence of **A** must therefore now be specified. For simplicity, we make as follows:

$$\nabla \cdot \mathbf{A} = 0 \tag{3.1.61}$$

The above Equation (3.1.61) is called the Coulomb gauge condition. In this way, Equation (3.1.60) can be written as:

$$\nabla^2 \mathbf{A} = -\mu \boldsymbol{J} \tag{3.1.62}$$

It shows that the magnetic vector potential **A** satisfies Poisson's equation in vector form.

The Equation (3.1.62) is equivalent to the Poisson's Equation in three scalar forms as follows,

$$\nabla^2 A_x = -\mu J_x \tag{3.1.63}$$

$$\nabla^2 A_y = -\mu J_y \tag{3.1.64}$$

$$\nabla^2 A_z = -\mu J_z \tag{3.1.65}$$

The form of these three equations is exactly the same as Poisson's equation of

electrostatic field potential. Referring to the solution form of Poisson's equation in the electrostatic field, when the current is distributed in a limited space and the value of the magnetic vector potential at infinity is specified to be zero, the solutions of each of the equations in above 3 equations are

$$A_x = \frac{\mu}{4\pi}\oint_{V'}\frac{J_x dV'}{R} \tag{3.1.66}$$

$$A_y = \frac{\mu}{4\pi}\oint_{V'}\frac{J_y dV'}{R} \tag{3.1.67}$$

$$A_z = \frac{\mu}{4\pi}\oint_{V'}\frac{J_z dV'}{R} \tag{3.1.68}$$

Combining the above three equations, we can obtain

$$\mathbf{A} = \frac{\mu}{4\pi}\oint_{V'}\frac{\boldsymbol{J}dV'}{R} \tag{3.1.69}$$

It was pointed out earlier that the elementary current section also has the form of $I\mathrm{d}\boldsymbol{l}$ and $\boldsymbol{K}\mathrm{dS}$, so the magnetic vector potential caused by the entire current of these two current distributions should be

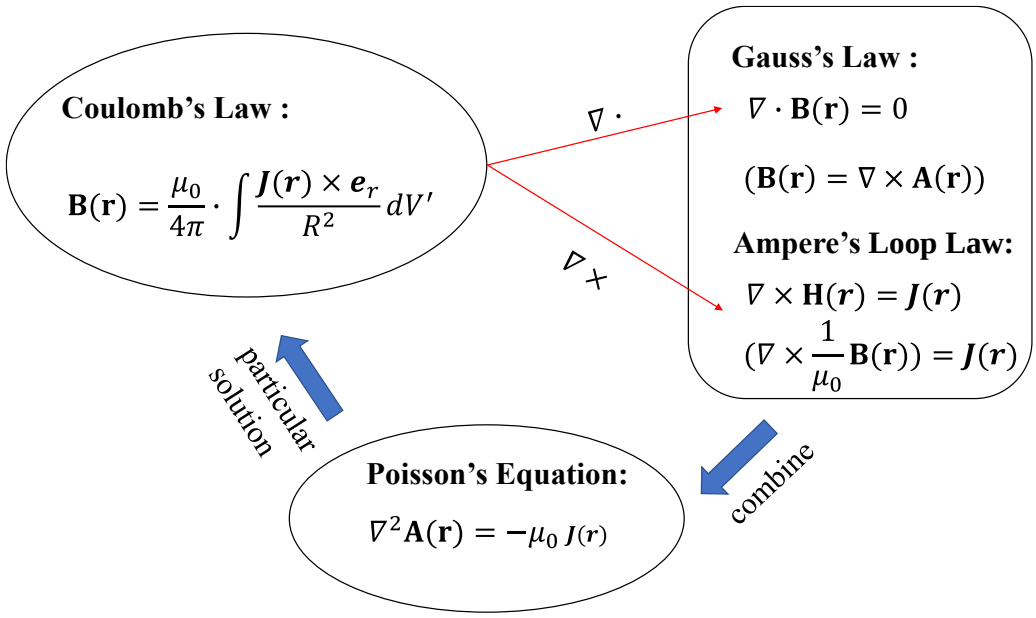


Fig. 3.1.11 The conversion of main theorems of magneto-static field

$$\mathbf{A} = \frac{\mu}{4\pi}\oint_{l'}\frac{Idl'}{R} \tag{3.1.70}$$

$$\mathbf{A} = \frac{\mu}{4\pi}\oint_{S'}\frac{\boldsymbol{K}dS'}{R} \tag{3.1.71}$$

From Equation (3.1.66), (3.1.67) and (3.1.68), it can be seen that the magnetic vector potential generated by each element current has the same direction as the element current.

At the end of section, the conversion relationship between the main theorems of magneto-static field is shown in the Fig.3.1.11.

3.1.3 Eddy Current Field

There are bulky conductors in many electrical equipment (such as the core of generators and transformers, etc.). But when these bulk conductors are in a changing magnetic field, currents are induced inside them. The characteristic of these currents is that they form a closed loop inside the bulk conductor and flow in a vortex shape, so they are called eddy currents. For example, when the solenoid coil in the cylindrical conductor

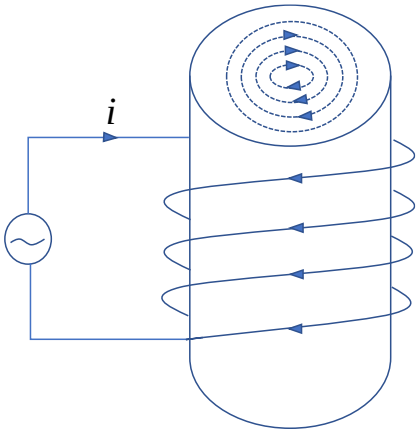


Fig. 3.1.12 The eddy current

core has an alternating current, the induced current or eddy current appears in the cylindrical conductor core, as shown in the Fig.3.1.12.

Eddy current is the current circulating in the conductor, just like the vortex flow in the

water flow. They are caused by magnetic field changes and flows in closed loops perpendicular to the plane of the magnetic field. They can be created when a conductor passes through a magnetic field, or when the magnetic field around a stationary conductor changes, i.e. anything that causes a conductor to change in the intensity or direction of the magnetic field can create eddy currents. The size of the eddy current is directly proportional to the size of the magnetic field, the area of the coil and the rate of change of the magnetic flux, and inversely proportional to the resistivity of the conductor.

Like any current flowing through a conductor, an eddy current generates its own magnetic field. Lenz's law states that the direction of a magnetically induced current, like an eddy current, causes the resulting magnetic field to change in opposition to the field that generated it. This resistance created by opposing magnetic fields is exploited in eddy current braking, which is commonly used as a method of stopping spinning power tools and roller coasters.

When the eddy current flows in the conductor, it will generate loss and cause the conductor to heat up, so it has a thermal effect. Likewise, eddy currents, like other electric currents, generate magnetic fields. This magnetic field is a change that weakens the external magnetic field, that is, the eddy current has a demagnetization effect. These two effects of eddy currents are both beneficial and detrimental. In industry, the thermal effect of eddy current is used to heat and smelt metals, and the demagnetization effect of eddy current is used to make electromagnetic brakes. However, in some cases, it is necessary to try to reduce the eddy current, so the study of the eddy current problem has practical significance.

Various electromagnetic phenomena can be described by Maxwell's equations under certain conditions,

$$\nabla \times \mathbf{H} = \boldsymbol{J} \tag{3.1.72}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{3.1.73}$$

$$\nabla \cdot \mathbf{D} = \rho \tag{3.1.74}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{3.1.75}$$

If the curl is taken on both sides of Equation (3.1.72), and the left side is expanded using Equation $\nabla \times \nabla \times \mathbf{F} = \nabla(\nabla \cdot \mathbf{F}) - \nabla^2 \mathbf{F}$, we can obtain,

$$\nabla \times \nabla \times \mathbf{H} = \nabla(\nabla \cdot \mathbf{H}) - \nabla^2 \mathbf{H} = \nabla \times \boldsymbol{J} \tag{3.1.76}$$

then use $\boldsymbol{J} = \sigma \mathbf{E}$, $\mathbf{B} = \mu \mathbf{H}$ and Equation (3.1.73) to eliminate $\boldsymbol{J}$, we can obtain,

$$\nabla^2 \mathbf{H} = \mu\sigma \frac{\partial \mathbf{H}}{\partial t} \tag{3.1.77}$$

and in the same way,

$$\nabla^2 \mathbf{E} = \mu\sigma \frac{\partial \mathbf{E}}{\partial t} \tag{3.1.78}$$

This is the differential equation satisfied by the electric field **E** and the magnetic field **H** at any point in the conductor.

Multiplying both sides of Equation (3.1.78) by the conductivity σ, and taking $\boldsymbol{J} = \sigma \mathbf{E}$ into account, gives

$$\nabla^2 \boldsymbol{J} = \mu\sigma \frac{\partial \boldsymbol{J}}{\partial t} \tag{3.1.79}$$

this is the differential equation that the current density $\boldsymbol{J}$ satisfies.

The electric field intensity **E**, magnetic field intensity **H** and current density $\boldsymbol{J}$ in the eddy current problem will obey the Equation (3.1.77), (3.1.78) and (3.1.79). These equations are usually called eddy current equations, and they are the basis for studying eddy current problems

3.1.4 Electromagnetic Wave

Starting from the basic equations of the electromagnetic field, this section derives the wave equation satisfied by the electric field **E** and magnetic field **H** of electromagnetic waves, and then discusses the solution of the wave equation under the condition of unbounded homogeneous media, that is uniform plane electromagnetic waves.

The basic equations of the electromagnetic field show that there is a coupling between the changing electric field and the changing magnetic field, and this coupling exists in space in the form of wave, that is, there is electromagnetic wave propagation in space. The propagation of changing electromagnetic field in space is called electromagnetic wave, which is radiated by the field source. A charged particle will form an electric field, which covers the entire universe. Once the position of this particle (that is, the field) changes, the electric field intensity at any point in space will change, and the information of this change will be transmitted at the speed of light. There are so many particles in the space, what they produce will superposition each other, and the field change information they produce will interfere with each other, some will strengthen and some will cancel, then a path with the smallest phase difference will be left in the end, this path has the characteristics of a wave, we call this wave an electromagnetic wave. Electromagnetic

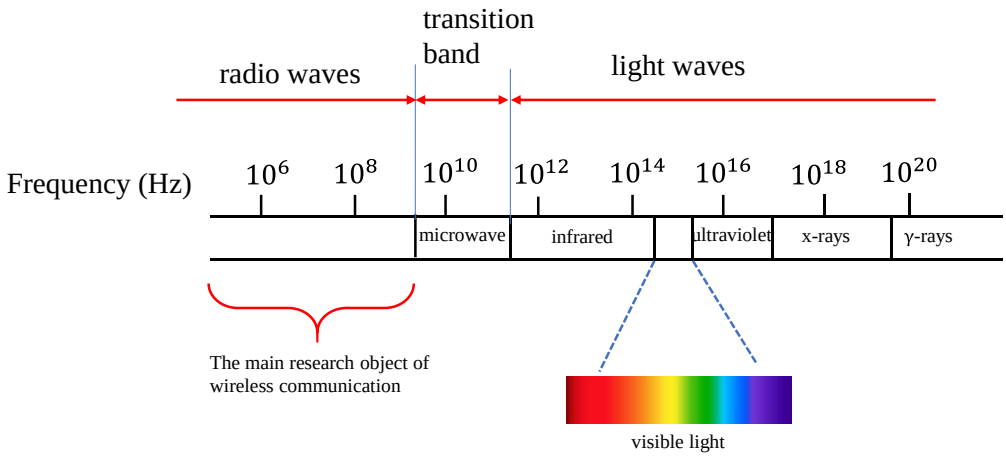


Fig. 3.1.13 Electromagnetic wave classification

waves do not need to rely on media to propagate, and their propagation speed in vacuum is the speed of light. Electromagnetic waves can be classified by frequency, from low frequency to high frequency, mainly including radio waves, megahertz radiation, microwaves, infrared rays, visible light, ultraviolet rays, x-rays and γ-rays as shown in the Fig.3.1.13.

In the passive space, both conduction current and free charge are zero, that is, $\mathrm{J} = 0$, $\rho = 0$. And then if the medium in the passive space is isotropic, linear and uniform, $\mathbf{D} = \varepsilon\mathbf{E}$, $\mathbf{B} = \mu\mathbf{H}$, $\boldsymbol{J} = \sigma\mathbf{E}$, according to the basic equations of the electromagnetic field, we can obtain,

$$\nabla \times \mathbf{H} = \gamma\mathbf{E} + \varepsilon\frac{\partial \mathbf{E}}{\partial t} \tag{3.1.80}$$

$$\nabla \times \mathbf{E} = -\mu\frac{\partial \mathbf{H}}{\partial t} \tag{3.1.81}$$

$$\nabla \cdot \mathbf{H} = 0 \tag{3.1.82}$$

$$\nabla \cdot \mathbf{E} = 0 \tag{3.1.83}$$

taking the curl of Equation (3.1.80) and using Equation (3.1.81), we can obtain,

$$\nabla \times \nabla \times \mathbf{H} = -\sigma\mu\frac{\partial \mathbf{H}}{\partial t} - \mu\varepsilon\frac{\partial^2 \mathbf{H}}{\partial t^2} \tag{3.1.84}$$

and then using the vector equation $\nabla \times \nabla \times \mathbf{H} = \nabla(\nabla \cdot \mathbf{H}) - \nabla^2\mathbf{H}$ and considering Equation (3.1.82), the above equation becomes,

$$\nabla^2\mathbf{H} - \sigma\mu\frac{\partial \mathbf{H}}{\partial t} - \mu\varepsilon\frac{\partial^2 \mathbf{H}}{\partial t^2} = 0 \tag{3.1.85}$$

and then,

$$\nabla^2\mathbf{E} - \sigma\mu\frac{\partial \mathbf{E}}{\partial t} - \mu\varepsilon\frac{\partial^2 \mathbf{E}}{\partial t^2} = 0 \tag{3.1.86}$$

Equation (3.1.85) and (3.1.86) are the equations satisfied by **E** and **H** in the passive space, called the electromagnetic wave equation. They are the basis for the study of

electromagnetic wave problems.

3.1.5 Maxwell's Equations

Maxwell's equations are a set of 4 partial differential equations that describe the world of electromagnetics. These equations describe how electric fields and magnetic fields are generated by charges and electric currents, and how they will propagate, interact with each other, and be influenced by other objects. The following 4 equations are the classical forms of Maxwell's Equations.

Gauss's law of electric fields can be expressed as:

$$\nabla \cdot \mathbf{D} = \rho \tag{3.1.87}$$

Gauss's law of magnetism can be expressed as:

$$\nabla \cdot \mathbf{B} = 0 \tag{3.1.88}$$

Ampère's circuital law can be expressed as:

$$\nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t} \tag{3.1.89}$$

Faraday's law of induction can be expressed as:

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{3.1.90}$$

Two supplementary expressions of Maxwell's equation, the electric flux and the magnetic flux, which are concerned with the areal density of the dielectric constant or and permeability, can be expressed as the following:

$$\mathbf{D} = \varepsilon \mathbf{E} \tag{3.1.91}$$

$$\mathbf{B} = \mu \mathbf{H} \tag{3.1.92}$$

and, the current density and the electric field are connected to the conductivity σ of the conductor by Ohm's law, which is expressed as the following equation:

$$\mathbf{J} = \sigma \mathbf{E} \tag{3.1.93}$$

As the foundation of classical electromagnetism, classical optics, and electric circuits, most of the electromagnetic properties that are required for microwave engineering can be deduced from Maxwell's equations.

3.2 Numerical Simulation Scheme for EM Field

3.2.1 Overview of Numerical Scheme for Partial Differential Equations

The application of electromagnetic field theory has spread to almost all fields of technical science, such as, life science and medicine, material science, information science and earth science. The research content of computational electromagnetism involves a wide range of fields. It penetrates into various fields of electromagnetism, and is interrelated and interdependent with electromagnetic field engineering and electromagnetic field theory. For electromagnetic field engineering, computational electromagnetics is to solve the problems of modeling, simulation and optimal design of more and more complex electromagnetic field problems in actual electromagnetic field engineering. And electromagnetic field engineering also provides experimental results for it to verify the correctness of its calculation results. For electromagnetic field theory, computational electromagnetics research can provide methods and calculation results for complex numerical and analytical operations. The research of electromagnetic field theory also provides electromagnetic laws and mathematical equations for computational electromagnetics research, and then verifies its calculation results.

The methods for solving electromagnetic field problems can be summarized into two categories, each of which contains several methods, the first category is analytical method,

the second category is numerical method.

Analytical methods include setting up and solving partial differential equations (PDE) or integrals equation. The classic method for solving partial differential equations is the method of separation of variables. The method of solving integral equations is mainly transformation mathematics. The advantage of the analytical method is as follows,

(1) The solution can be expressed as the explicit of a known function, so that precise numerical results can be calculated.

(2) It can be used as a test standard for approximate solutions and numerical solutions.

(3) In the analysis process and in the explicit solution, the internal connection of the problem and the effect of each parameter on the numerical result can be observed.

Traditionally, most electromagnetic field problems are solved based on analytical models. Starting from Maxwell's equations and adding specific boundary conditions, a system of differential or integral equations can be obtained, as shown in Fig.3.2.1.

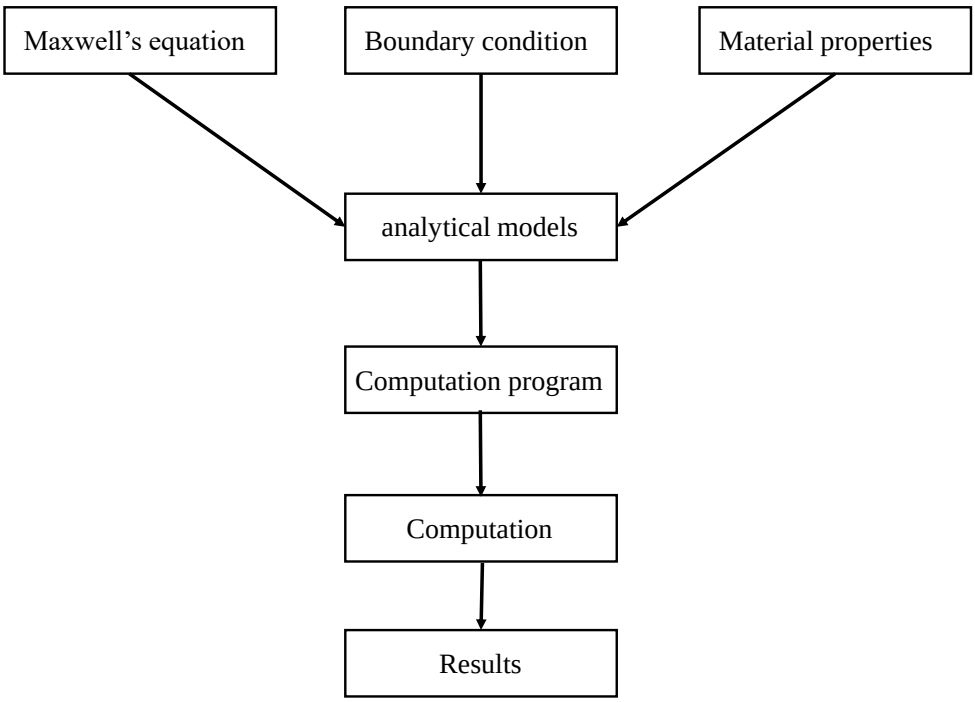


Fig. 3.2.1 Traditional solution process of Electromagnetic field problems

Compared with the analytical method, the numerical method has stronger applicability. The emergence of numerical methods makes the analysis and research of electromagnetic field problems enter from analytical classical methods to numerical analysis methods of discrete systems, so that many complex electromagnetic field problems that are difficult to solve by analytical methods may be obtained through computer-aided analysis of electromagnetic fields. High-precision discrete solutions (numerical solutions) can greatly promote the development of various numerical calculation methods for electromagnetic fields. However, the disadvantage of the numerical method is that the amount of data input is large, the amount of calculation is large, and it is greatly limited by hardware conditions. In principle, numerical methods can solve geomagnetic field engineering problems with any complex geometry and complex materials. However, in engineering applications, due to the limitations of computer storage capacity, execution time and numerical error of the solution, numerical methods are difficult to complete the task.

There are two main types of methods currently used in computational electromagnetics, one is numerical methods based on integral equations of electromagnetic field problems, such as the method of moments series. The other is based on differential equations of electromagnetic field problems Numerical methods, such as the family of finite difference methods. The finite element method based on the variational principle can be classified as a differential equation method, and can also be described in the language of the method of moments. It should be pointed out that because the integral equation description and the differential equation description of the electromagnetic field problem can be converted to each other, for the same electromagnetic problem, the above two types of methods are equivalent to each other. Moreover, combining the integral equation method

and the differential equation method to solve the same electromagnetic problem can play to their respective advantages.

3.2.2 Finite Difference Methods (FDM)

In the calculation method of numerical analysis of electromagnetic field, the finite difference method (FDM) is the earliest application method. The finite difference method (FDM) (referred to as the difference method) is a numerical method based on the difference principle. It essentially transforms the problem of the continuous domain of the electromagnetic field into a discrete system problem, that is, approaches the real solution of the continuous field through the numerical solution of each discrete point on the grid discretization model. The finite difference method has a wide range of applications. It can not only solve the potential field in the uniform or inhomogeneous linear medium, but also solve the field in the nonlinear medium. It can not only solve the constant field or quasi-steady field, but also solve the time-varying field. Among the numerical methods of boundary value problems, this method is quite simple. If the computer storage capacity allows, it is possible to use a finer grid, so that the discretization model can approach the real problem more accurately, and obtain sufficient numerical solution to precision.

In this section, we take the solution of the electrostatic field boundary value problem as an example to introduce the finite difference method (FDM). The basic idea of finite difference method is as follows,

(1) Divide the field with a grid

(2) Replace the Laplace equation with the potential at each grid node as the difference equation of the unknown

(3) Change the problem of finding the solution of the Laplace equation into the problem of finding the solution of the system of simultaneous difference equations.

According to the previous section, the basic equation of the electrostatic field is as follows,

$$\mathbf{E} = -\nabla\emptyset \tag{3.2.1}$$

$$\nabla \cdot \mathbf{D} = \rho \tag{3.2.2}$$

and considering $\mathbf{D} = \varepsilon\mathbf{E}$, the above Equation (3.2.2) can be rewritten as,

$$\nabla^2\emptyset = -\frac{\rho}{\varepsilon} \tag{3.2.3}$$

that is Poisson's Equation.

As shown in the Fig.3.2.2, in the two-dimensional region D, the electric potential function $\emptyset$ satisfies the Laplace equation ($\rho = 0$),

$$\frac{\partial}{\partial z} = 0 \tag{3.2.4}$$

and then,

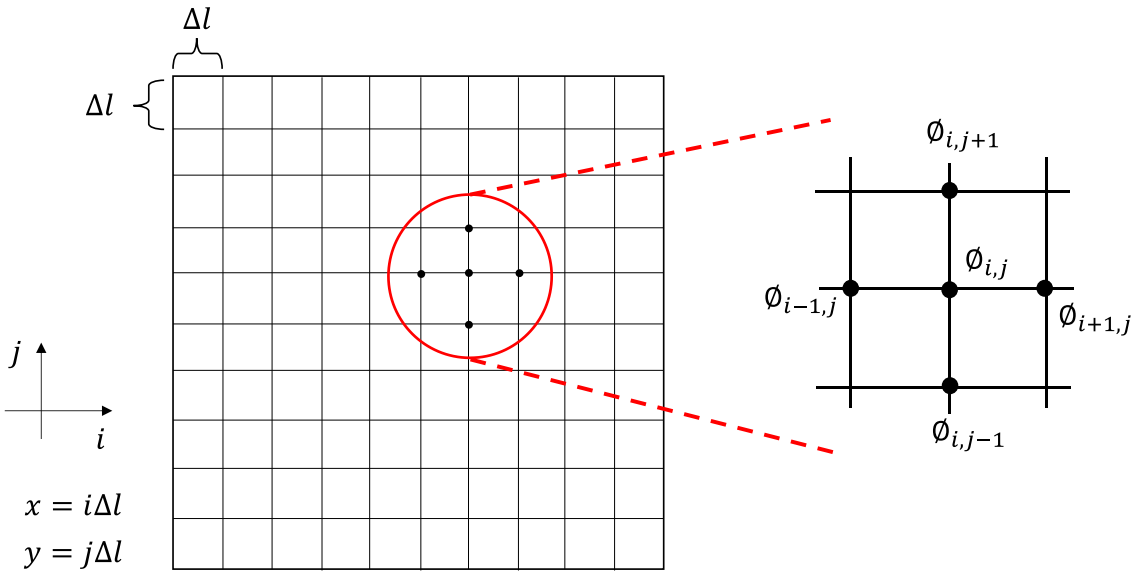


Fig. 3.2.2 Finite difference meshing

$$\frac{\partial^2 \emptyset(x,y)}{\partial x^2} + \frac{\partial^2 \emptyset(x,y)}{\partial y^2} = 0 \tag{3.2.5}$$

To apply the finite difference method, the distribution mode of the grid nodes must be determined first. As shown in the Fig.3.2.2, use two sets of straight lines (grid lines) parallel to the *x* and *y* axes to divide the field D into enough square grids. The intersection points of the grid lines are called nodes, and the distance between two adjacent parallel grids lines is Δl.

After dividing the grid, it is necessary to discretize the Laplace equation,

$$\frac{\partial \emptyset}{\partial x} = \frac{\emptyset_{i+1,j} - \emptyset_{i,j}}{\Delta l} \tag{3.2.6}$$

this is forward difference, and the backward difference as follows,

$$\frac{\partial \emptyset}{\partial x} = \frac{\emptyset_{i,j} - \emptyset_{i-1,j}}{\Delta l} \tag{3.2.7}$$

Here Δl is small enough. So for the second order partial differential equation, we have the follows,

$$\frac{\partial^2 \emptyset}{\partial x^2} = \frac{\partial}{\partial x}\frac{\partial \emptyset}{\partial x} = \frac{\frac{\emptyset_{i+1,j} - \emptyset_{i,j}}{\Delta l} - \frac{\emptyset_{i,j} - \emptyset_{i-1,j}}{\Delta l}}{\Delta l} \tag{3.2.8}$$

We substitute the above Equation (3.2.8) into the Laplace's equation (3.2.5), then the discretization of the Laplace's equation is completed,

$$\frac{\frac{\emptyset_{i+1,j} - \emptyset_{i,j}}{\Delta l} - \frac{\emptyset_{i,j} - \emptyset_{i-1,j}}{\Delta l}}{\Delta l} + \frac{\frac{\emptyset_{i,j+1} - \emptyset_{i,j}}{\Delta l} - \frac{\emptyset_{i,j} - \emptyset_{i,j-1}}{\Delta l}}{\Delta l} = 0 \tag{3.2.9}$$

after rearranging the above equation, we can obtain,

$$\emptyset_{i+1,j} + \emptyset_{i-1,j} + \emptyset_{i,j+1} + \emptyset_{i,j-1} - 4\emptyset_{i,j} = 0 \tag{3.2.10}$$

the above equation is called the difference form of the Laplace's equation, or the difference equation.

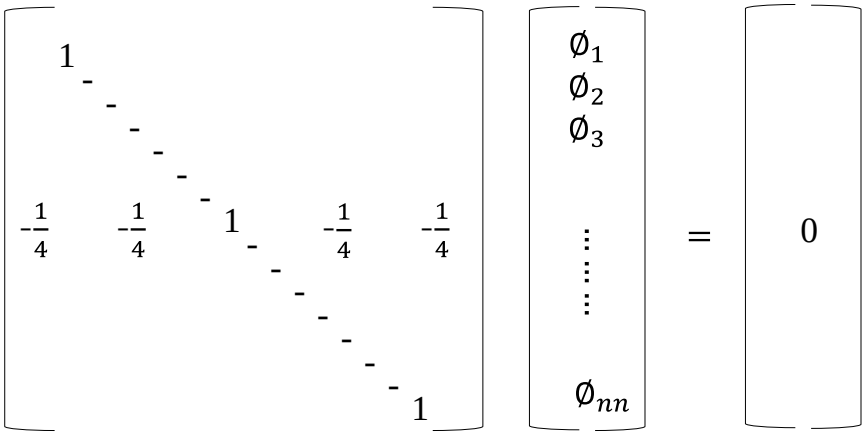


Fig. 3.2.3 The equations for FDM

It can be seen from the above that each node in the field D has a differential equation, through which the electric potentials of each internal node and the node electric potential on the boundary are connected. As long as the solution of the simultaneous equations is

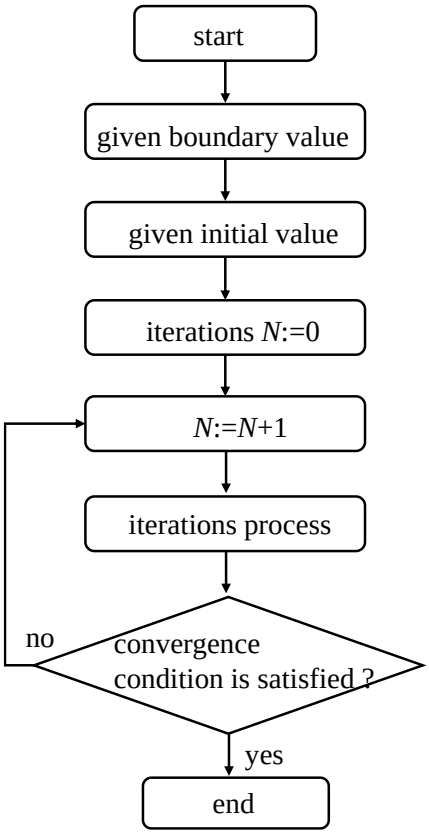


Fig. 3.2.4 Iterative program

shown in the Fig.3.2.3, the electric potential value of each node can be obtained. When solving practical problems, due to the large number of nodes, the number of simultaneous difference equations can often reach hundreds or even thousands. The usual direct methods for solving simultaneous equations (such as determinant method, elimination method, etc.) are no longer applicable. Therefore, for the solution of large sparse matrices, we usually consider iterative methods, which will be explained in detail in later chapters. When calculating with the aid of a computer, its program block diagram is as shown Fig.3.2.4.

3.2.3 Finite-Difference Time-Domain Method (FDTD)

The finite-difference time-domain (FDTD) method is a numerical method for the analysis of electromagnetic field, and its basic algorithm was proposed by K. S. Yee in 1966. In the time domain, Maxwell's curl equation is changed to a discrete difference form, and the continuous space is divided into finite grids for calculation. The larger the number of grids, the more accurate the calculation result, and the corresponding large increase in the calculation amount. In this section, we take 3D microwave simulation as an example to introduce the FDTD method in detail.

The Maxwell's equation is as follows,

$$\nabla \times \mathbf{E} = -\mu \frac{\partial \mathbf{H}}{\partial t} \tag{3.2.11}$$

$$\nabla \times \mathbf{H} = \varepsilon \frac{\partial \mathbf{E}}{\partial t} \tag{3.2.12}$$

$$\nabla \cdot \mathbf{E} = 0 \tag{3.2.13}$$

$$\nabla \cdot \mathbf{H} = 0 \tag{3.2.14}$$

and the differential equation of Equation (3.2.11) is as follows,

$$\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = -\mu \frac{\partial H_x}{\partial t} \tag{3.2.15}$$

$$\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x} = -\mu \frac{\partial H_y}{\partial t} \tag{3.2.16}$$

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = -\mu \frac{\partial H_z}{\partial t} \tag{3.2.17}$$

and the differential equation of Equation (3.2.12) is as follows,

$$\frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z} = \varepsilon \frac{\partial E_x}{\partial t} \tag{3.2.18}$$

$$\frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x} = \varepsilon \frac{\partial E_y}{\partial t} \tag{3.2.19}$$

$$\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} = \varepsilon \frac{\partial E_z}{\partial t} \tag{3.2.20}$$

Then considering the Yee algorithm of 3D Maxwell's equation. The Yee algorithm is used to assign each component of the electromagnetic field to each cell. First, the space is divided into cubes as shown in Fig.3.2.5. The six components of the electromagnetic field are placed on the edge of the cube and the center point of the surface at the sampling points of the space. The electric field and magnetic field components are in the Always differ by half a grid step in any direction.

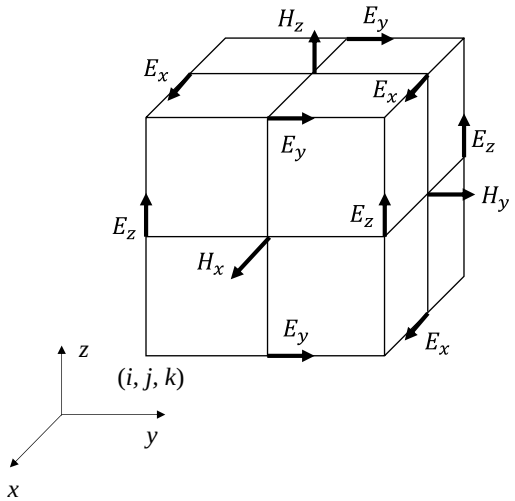


Fig. 3.2.5 The Yee cell

Each component of the electromagnetic field is staggered in space. Arranged as follows,

$$E_x(x,y,z,t) = E_x\left(\left\{i+\frac{1}{2}\right\}\Delta x, j\Delta y, k\Delta z, n\Delta t\right) \equiv E_x^n(i+\frac{1}{2},j,k) \tag{3.2.21}$$

$$E_y(x,y,z,t) = E_y\left(i\Delta x, \left\{j+\frac{1}{2}\right\}\Delta y, k\Delta z, n\Delta t\right) \equiv E_y^n(i,j+\frac{1}{2},k) \tag{3.2.22}$$

$$E_z(x,y,z,t) = E_z\left(i\Delta x, j\Delta y, \left\{k+\frac{1}{2}\right\}\Delta z, n\Delta t\right) \equiv E_z^n(i,j,k+\frac{1}{2}) \tag{3.2.23}$$

$$H_x(x,y,z,t) = H_x\left(\left\{i+\frac{1}{2}\right\}\Delta x, j\Delta y, k\Delta z, n\Delta t\right) \equiv H_x^{n+\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2}) \tag{3.2.24}$$

$$H_y(x,y,z,t) = H_y\left(i\Delta x, \left\{j+\frac{1}{2}\right\}\Delta y, k\Delta z, n\Delta t\right) \equiv H_y^{n+\frac{1}{2}}(i+\frac{1}{2},j,k+\frac{1}{2}) \tag{3.2.25}$$

$$H_z(x,y,z,t) = H_z\left(i\Delta x, j\Delta y, \left\{k+\frac{1}{2}\right\}\Delta z, n\Delta t\right) \equiv H_z^{n+\frac{1}{2}}(i+\frac{1}{2},j+\frac{1}{2},k) \tag{3.2.26}$$

In terms of time, Yee samples the electric field component and the magnetic field component by half a step, as shown in Fig.3.2.6.

Differentialize Maxwell's equations so that each component of the electromagnetic field can be placed in the space-time of the Yee algorithm. We here take $\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z} = -\mu\frac{\partial H_x}{\partial t}$ as an example to explain,

$$\begin{aligned}&\frac{1}{\Delta y}\left\{E_z^n\left(i,j+1,k+\frac{1}{2}\right) - E_z^n(i,j,k+\frac{1}{2})\right\}\\ &\qquad -\frac{1}{\Delta z}\left\{E_y^n\left(i,j+\frac{1}{2},k+\frac{1}{2}\right) - E_y^n\left(i,j+\frac{1}{2},k\right)\right\}\\ &\qquad = -\frac{\mu}{\Delta t}\left\{H_x^{n+\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right) - H_x^{n-\frac{1}{2}}(i,j+\frac{1}{2},k+\frac{1}{2})\right\}\end{aligned} \tag{3.2.27}$$

and then the above equation can be rewritten as,

$$\begin{aligned}&H_x^{n+\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right) =\\ &H_x^{n-\frac{1}{2}}\left(i,j+\frac{1}{2},k+\frac{1}{2}\right) - \frac{\Delta t}{\mu\Delta y}\left\{E_z^n\left(i,j+1,k+\frac{1}{2}\right) - E_z^n\left(i,j,k+\frac{1}{2}\right)\right\}\\ &\qquad +\frac{\Delta t}{\mu\Delta z}\left\{E_y^n\left(i,j+\frac{1}{2},k+1\right) - E_y^n\left(i,j+\frac{1}{2},k\right)\right\}\end{aligned} \tag{3.2.28}$$

Equation (3.2.15) ~ (3.2.20) can be differentiated by the Yee algorithm, which is similar to the differential process of above equation.

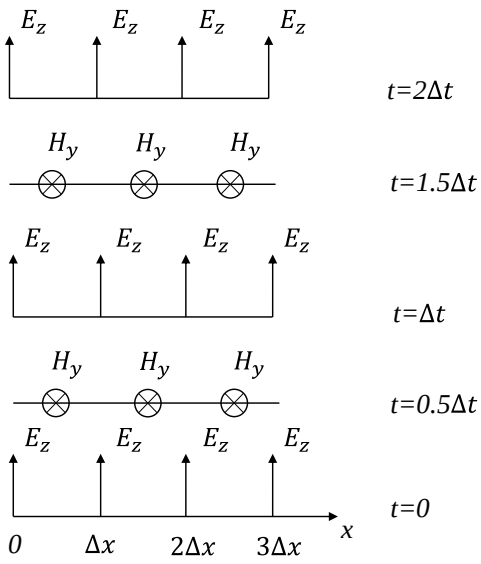


Fig. 3.2.6 The Yee cell in the terms of time

3.3 Numerical Scheme for Matrix Solver

In the previous section, we introduced the numerical analysis method of electromagnetic field simulation, and then introduced the method of solving simultaneous equations, such as LU decomposition, simple iterative method, and iterative method for solving large sparse matrices.

3.3.1 LU Decomposition

LU decomposition is a type of matrix factorization, which aims to represent a certain matrix A as a product of two or more matrices. As the name shows, LU decomposition is to represent the matrix A as A=LU, where L matrix represents Lower Triangular matrix, and the U matrix represents Upper Triangular matrix.

First, we consider a system of equations in three variables, and it can be written in the form of Ax=b as follows,

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{3.3.1}$$

here,

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{3.3.2}$$

And then we consider to generate matrix as A=LU,

$$L = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \tag{3.3.3}$$

and,

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \tag{3.3.4}$$

that means,

$$\begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{3.3.5}$$

by expanding the left side matrices of the above equation, we can obtain,

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ l_{21}u_{11} & l_{21}u_{12} + u_{22} & l_{21}u_{13} + u_{23} \\ l_{31}u_{11} & l_{31}u_{12} + l_{32}u_{22} & l_{31}u_{13} + l_{32}u_{23} + u_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{3.3.6}$$

therefore, by equating the corresponding elements on both sides of above equation, we can obtain as follows,

$$u_{11} = a_{11}, u_{12} = a_{12}, u_{13} = a_{13},$$

$$l_{21}u_{11} = a_{21}, l_{21}u_{12} + u_{22} = a_{22}, l_{21}u_{13} + u_{23} = a_{23} \tag{3.3.7}$$

$$l_{31}u_{11} = a_{31}, l_{31}u_{12} + l_{32}u_{22} = a_{32}, l_{31}u_{13} + l_{32}u_{23} + u_{33} = a_{33}$$

by solving the above equation, we can obtain,

$$u_{11} = a_{11}, u_{12} = a_{12}, u_{13} = a_{13},$$

$$l_{21} = \frac{a_{21}}{a_{11}}, u_{22} = a_{22} - \frac{a_{21}a_{12}}{a_{11}}, u_{23} = a_{23} - \frac{a_{21}a_{13}}{a_{11}}$$

$$l_{31} = \frac{a_{31}}{a_{11}}, l_{32} = \frac{a_{32} - \frac{a_{31}a_{12}}{a_{11}}}{a_{22} - \frac{a_{21}a_{12}}{a_{11}}}, \tag{3.3.8}$$

$$u_{33} = a_{33} - \frac{a_{31}a_{13}}{a_{11}} - (\frac{a_{32} - \frac{a_{31}a_{12}}{a_{11}}}{a_{22} - \frac{a_{21}a_{12}}{a_{11}}})a_{23} - \frac{a_{21}a_{13}}{a_{11}}$$

Now, we can write LUX=b, and then we assume UX=Y, where,

$$\mathrm{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} \tag{3.3.9}$$

so we can rewrite LY=b, and next we consider to solve this equation,

$$\begin{bmatrix} 1 & 0 & 0 \\ \frac{a_{21}}{a_{11}} & 1 & 0 \\ \frac{a_{31}}{a_{11}} & \frac{a_{32} - \frac{a_{31}a_{12}}{a_{11}}}{a_{22} - \frac{a_{21}a_{12}}{a_{11}}} & 1 \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{3.3.10}$$

thus, we can get,

$$Y_1 = b_1,$$

$$\frac{a_{21}}{a_{11}} Y_1 + Y_2 = b_2$$

$$\frac{a_{31}}{a_{11}} Y_1 + \frac{a_{32} - \frac{a_{31}a_{12}}{a_{11}}}{a_{22} - \frac{a_{21}a_{12}}{a_{11}}} Y_2 + Y_3 = b_3 \tag{3.3.11}$$

solving the above equation, we can get,

$$Y_1 = b_1,$$

$$Y_2 = b_2 - \frac{a_{21}}{a_{11}} b_1 \tag{3.3.12}$$

$$Y_3 = b_3 - \frac{a_{31}}{a_{11}}\ b_1 - \frac{a_{32} - \frac{a_{31}a_{12}}{a_{11}}}{a_{22} - \frac{a_{21}a_{12}}{a_{11}}}(b_2 - \frac{a_{21}}{a_{11}}b_1)$$

At the final step of LU decomposition, we consider UX=Y now,

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} \tag{3.3.13}$$

because the U and Y are known matrices, so we can get the final result as follows,

$$x_1 = \frac{Y_1 - (\frac{Y_2 - \frac{Y_3 u_{23}}{u_{33}}}{u_{22}})u_{12}}{u_{11}}$$

$$x_2 = \frac{Y_2 - \frac{Y_3 u_{23}}{u_{33}}}{u_{22}} \tag{3.3.14}$$

$$x_3 = \frac{Y_3}{u_{33}}$$

3.3.2 Simple Iterative Method

The simple iterative method is a numerical calculation method for solving problems such as roots of functions and solutions of equations. Based on the fixed-point theorem, the method continuously approaches the solution of the function by iterating an initial approximation until the accuracy requirement is satisfied.

Specifically, for a linear system of equation, A*x*=b, the simple iterative method converts the equation system into the form of *x*=B*x*+c, where *x*=g(x)is a new function, usually taken as the deformation of A*x*=b. Then, starting from an initial value x_0, the iterative formula is used repeatedly until the precision requirement satisfied, that is, $|x_{n+1} - x_n| < \varepsilon$, where ε is the preset precision value.

In practical applications, simple iterative method has the advantages of simple

implementation, easy understanding and debugging, but the convergence speed of simple iterative method may be slower than other numerical calculation method, and the accuracy of the solution can not be guaranteed.

3.3.3 Conjugate Gradient Method (CG)

Many scientific calculations ultimately boil down to solving a large-scale linear equation system A**x**=**b**, and various methods to get the solution of system of linear equations have been proposed depending on the application. Sparse matrix, dense matrix, direct method and iterative method. Sparse matrices appear in numerical analysis such as FEM and FDM, and dense matrices usually appear in numerical analysis such as BEM. Direct methods such as LU decomposition have the advantages of stability and wide application range, but at the same time, compared with iterative methods, they require more computing time and memory when calculating dense matrices, and are not suitable for large-scale calculations. The iterative method is divided into stationary iterative and nonstationary iterative. stationary iterative methods such as Gauss's elimination, Jacobi, etc. usually converge slowly. In the previous section, we explained the simple iterative method, one of the stationary iterative methods. The nonstationary iterative method adds constraints and optimization conditions, also known as the Krylov subspace method, usually including CG iterative method, BiCG-Stab iterative method and GMRES iterative method, etc. Compared with the direct methods, the iterative methods have the advantages of less calculation and memory, and is suitable for parallel computing, but because the convergence is greatly affected by the boundary conditions, there is a possibility of non-convergence. In this subsection, we explain the concept of Krylov subspace method to illustrate the CG iterative algorithm.

Given the following system of linear equations,

$$\mathrm{A}\mathbf{x}=\mathbf{b} \tag{3.3.15}$$

The basic idea of krylov subspace method is to find the approximate solution of the above equation in a subspace with a smaller dimension. This method is also regarded as a projection method, which is to find the projection of the true solution in a certain subspace (it can be an orthogonal projection or an oblique projection.)

The first step of Krylov subspace method is to find the suitable subspaces *K*,

$$K_n = K_n(A, \boldsymbol{v}) = span(\boldsymbol{v}, A\boldsymbol{v}, A^2\boldsymbol{v}, \dots, A^{n-1}\boldsymbol{v}) \tag{3.3.16}$$

where $\boldsymbol{v}$ is a vector, $\boldsymbol{v} \in R^N$, and $n \leq N$, and K_n is the subspace of all the vector *m* of R^N, thus we can write the following form,

$$m = \pi(A)\boldsymbol{v}, \pi \subseteq p_{n-1} \tag{3.3.17}$$

where p_j is the set of all the polynomials.

Since we are looking for an approximate solution in a lower-dimensional subspace, so there is,

$$K_1 \subseteq K_2 \subseteq K_3 \subseteq K_4 \dots \tag{3.3.18}$$

and the dimensional cannot exceed the *N*. Then considering the minimal degree *v*,

$$\dim K_n(A, \boldsymbol{v}) = \min(\mathrm{n}, v) \tag{3.3.19}$$

So far, we have determined the subspace, and the second step of krylov subspace method is to give a initial value $\boldsymbol{x}_0$ and the corresponding residual is as follows,

$$\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0 \tag{3.3.20}$$

and then generates iterates $\boldsymbol{x}_n$,

$$\boldsymbol{x}_n - \boldsymbol{x}_0 = \pi_{n-1}(A)\boldsymbol{r}_0,\ \boldsymbol{x}_n \in \boldsymbol{x}_0 + K_n(A, \boldsymbol{r}_0) \tag{3.3.21}$$

and the residual $\boldsymbol{r}_n = \boldsymbol{b} - A\boldsymbol{x}_n$ will satisfy,

$$\boldsymbol{x}_n - \boldsymbol{x}_0 = \xi_n(A)\boldsymbol{r}_0 \in AK_n(A, \boldsymbol{r}_0) \in AK_{n+1}(A, \boldsymbol{r}_0) \tag{3.3.22}$$

where $\xi_n \in p_n$,

$$\xi_n(z) = 1 - z\pi_{n-1}(z),\ \xi_n(0) = 1 \tag{3.3.23}$$

Conjugate Gradient method as the one of representative nonstationary iterative method, mainly for symmetric positive definite matrices to solve. Then we will list the execution process of CG method.

compute $\boldsymbol{r}^{(0)} = \boldsymbol{b} - A\boldsymbol{x}^{(0)}$

for i=1,2,3,…

$p_{i-1} = \boldsymbol{r}^{(i-1)}\boldsymbol{r}^{(i-1)}$

If i=1

$\boldsymbol{p}^{(1)} = \boldsymbol{r}^{(0)}$

else

$\beta_{i-1} = p_{i-1}/p_{i-2}$

$\boldsymbol{p}^{(i)} = \boldsymbol{r}^{(i-1)} + \beta_{i-1}\boldsymbol{p}^{(i-1)}$ (3.3.24)

end if

$\boldsymbol{q}^{(i)} = A\boldsymbol{p}^{(i)}$

$\alpha_i = \boldsymbol{p}_{i-1}/\boldsymbol{p}^{(i)}\boldsymbol{q}^{(i)}$

$\boldsymbol{x}^{(i)} = \boldsymbol{x}^{(i-1)} + \alpha_i\boldsymbol{p}^{(i)}$

$\boldsymbol{r}^{(i)} = \boldsymbol{r}^{(i-1)} - \alpha_i\boldsymbol{q}^{(i)}$

check convergence $|r|$

end

3.3.4 Bi-Conjugate Gradient Stabilized (BiCG-Stab)

In this section, we will explain another representative nonstationary iterative method, that is Bi-Conjugate Gradient Stabilized (BiCG-Stab). Compared with the CG method,

BiCG-Stab method is more stable, and BiCG-Stab method can handle not only symmetric matrices, but also asymmetric matrices. The main idea of the BiCG-Stab algorithm is based on the bilateral Lanczos algorithm, which is an iterative method based on the residual orthogonal subspace. The unilateral Lanczos algorithm iterates through specific rules in the residual subspace, so that the residual tends to 0, and then to get the equation solution, this iteration is finite. However, BiCG-Stab method, an iterative method based on krylov residual subspace, has fast convergence speed, high precision and good stability. The iteration process of BiCG-Stab method is as follows,

$\boldsymbol{x}_0$ initial value guess

$\boldsymbol{r}_0 = \mathbf{b} - A\boldsymbol{x}_0$

$r_0^*, (r_0^*, \boldsymbol{r}_0) \neq 0$, e.g., $r_0^* = \boldsymbol{r}_0$

$\beta_{-1} = 0$

for n=0, 1, 2, ……until $\| \boldsymbol{r}_n \| \leq \varepsilon \|\mathbf{b}\|$

begin

$$p_n = \boldsymbol{r}_n + \beta_n (p_n - \xi_n A p_{n-1})$$

$$\alpha_n = \frac{(\boldsymbol{r}^*, \ \boldsymbol{r}_n)}{(\boldsymbol{r}^*, \ A\boldsymbol{p}_n)} \tag{3.2.25}$$

$$\boldsymbol{t}_n = \boldsymbol{r}_n - \alpha_n A p_n$$

$$\xi_n = \frac{(A\boldsymbol{t}_n, \ \boldsymbol{t}_n)}{(A\boldsymbol{t}_n, \ A\boldsymbol{t}_n)}$$

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_{n+1} + \alpha_n \boldsymbol{p}_n + \xi_n \boldsymbol{t}_n$$

$$\boldsymbol{r}_{n+1} = \boldsymbol{t}_n - \xi_n A \boldsymbol{t}_n$$

$$\beta_n = \frac{\alpha_n (\boldsymbol{r}^*, \ \boldsymbol{r}_{n+1})}{\xi_n (\boldsymbol{r}^*, \ \boldsymbol{r}_n)}$$

end

3.3.5 GMRES

GMRES is an iterative algorithm, which is often used to solve the solution of large sparse non-symmetric equations. It approaches the solution of the equation through the vector that minimizes the residual obtained in the Krylov subspace. If there is a system of equations in the following form,

$$\mathrm{A}\mathbf{x}=\mathbf{b} \tag{3.3.26}$$

The characteristic polynomial of A is a nullifying polynomial of A, that is, there is a set of coefficients $a_1 \sim a_2$, and then we can get the following equation,

$$f(\mathrm{A}) = A^n + a_{n-1}A^{n-1} + \cdots + a_0 I \tag{3.3.27}$$

and the above equation can be rewritten as the following form,

$$A^{-1} = \frac{1}{a_0}A^{n-1} - \frac{a_{n-1}}{a_0}A^{n-2} - \cdots - \frac{a_1}{a_0}\ I \tag{3.3.28}$$

When solving a system of linear equations, any initial value x_0, corresponding to an initial residual $r_0 = b - Ax_0$, and then the exact solution of this system of equations can be expressed as,

$$x = x_0 + A^{-1}r_0 \tag{3.3.29}$$

by substituting A^{-1}, we can draw the conclusion: the exact solution of the system of equations can be found in the space theoretically. However, it is difficult to achieve in the reality, and then we will search for approximate solutions in a lower-dimensional subspace as we explained in the previous section about Krylov subspace method.

Define the m-dimensional Krylov subspace,

$$K_m = span(\boldsymbol{r}, A\boldsymbol{r}, A^2\boldsymbol{r}, \dots, A^{m-1}\boldsymbol{r}) \tag{3.3.30}$$

where, $\dim K_m = m$.

Assume that the best approximate solution of the analytical solution is x^m in the affine space $x_0 + K_m$. Then the solution for x^m can be transformed the two parts, one is to search for a set of suitable basis, and the other one is to solve the linear expression

coefficient y of x^m under this set of basis.

The next step is to consider the selection of the basis, because $\boldsymbol{r}, A\boldsymbol{r}, A^2\boldsymbol{r}, \ldots, A^{m-1}\boldsymbol{r}$ are linearly independent, so they are the basis of K_m. However, we can obtain some good properties and simplify subsequent calculations by selecting a set of orthonormal bases and using the orthonormal bases. A set of orthonormal basis is constructed by the Arnoldi process based on modified Gram-Schmidt orthogonalization.

After obtaining the orthonormal basis, we generally use the method of minimizing residuals: $\|\boldsymbol{r}_m\|_2 = \|\boldsymbol{b} - A\boldsymbol{x}^m\|_2$, the smaller it is, the closer it is to the exact solution.

With the previous foreshadowing, now we can start the process of introducing the GMRES algorithm:

(1) Let m=1.

(2) Define the Krylov subspace.

(3) Search for the best approximate solution in $x_0 + K_m$.

(4) Judging whether the approximate solution meets the accuracy requirements, if so, return the result, otherwise, increment m and return to step (2).

The iterative process of GMRES algorithm is as follows,

Set an initial guess $\boldsymbol{x}_0$

Compute $\boldsymbol{r}_0 = \mathbf{b} - A\boldsymbol{x}_0$

Set $\beta = \|r_0\|_2,\ \boldsymbol{v}_1 = \boldsymbol{r}_0/\beta,\ \boldsymbol{e}_1 = [\beta, 0, \ldots, 0]^T$

For *k*=1, 2, …

$\boldsymbol{w}_{k+1} = A\boldsymbol{v}_k$ (3.3.31)

For *i*=1, 2, … *k*

$h_{i,k}$=$(\boldsymbol{w}_{k+1}, \boldsymbol{v}_i\ )$

$\boldsymbol{w}_{i,k} = \boldsymbol{w}_{i,k} - h_{i,k}\boldsymbol{v}_i$

End For

$h_{k+1,k}=\|\boldsymbol{w}_{k+1}\|_2$

$$\boldsymbol{w}_{k+1}=\frac{\boldsymbol{w}_{k+1}}{h_{k+1,k}}$$

For i=1, 2, … k-1

$$\begin{pmatrix} h_{i,k} \\ h_{i+1,k} \end{pmatrix}=\begin{pmatrix} \bar{c}_i & \bar{s}_i \\ -s_i & c_i \end{pmatrix}\begin{pmatrix} h_{i,k} \\ h_{i+1,k} \end{pmatrix}$$

End For

$$c_k=\frac{h_{k,k}}{\sqrt{h_{k,k}^2+\left|h_{k+1,k}\right|^2}}$$

$$s_k=\frac{h_{k,k}}{\sqrt{h_{k,k}^2+\left|h_{k+1,k}\right|^2}}$$

$$e_{k+1}=-s_k e_k$$

$$e_k=\bar{c}_k e_k$$

$$h_{k,k}=\sqrt{h_{k,k}^2+\left|h_{k+1,k}\right|^2}$$

$$h_{k+1,k}=0$$

If $e_{k+1}\leq e\|\boldsymbol{b}\|_2$ then

$$\boldsymbol{y}_k=H_k^{-1}\boldsymbol{e}_1$$

$$\boldsymbol{x}_k=\boldsymbol{x}_0+\sum_{i=1}^{k}\boldsymbol{v}_i\,\boldsymbol{y}_i$$

Stop

End If

End For

3.4 Logic Circuits

3.4.1 Basis of logic Circuits

A logic circuit is an electronic circuit that is used to implement logical operations and logical functions in Boolean algebra. It is a part of a digital circuit, usually consisting of logic gates (such as AND, OR, NOT, etc.), which can accept one or more inputs and generate one or more outputs. Logic circuits are the basis of modern computers and electronic devices, which are used to process digital signals and data. Logic circuits can generally be divided into two types: combinational circuits and sequential logic circuits. The output of a combinational logic circuit depends only on the current input, not on past inputs or outputs. The output of a sequential logic circuit depends on current inputs and past inputs and, or outputs. Logic circuits are commonly used in a computer's CPU and memory, but also in controllers and other digital circuits. As with any digital circuit design, the design and implementation of logic circuits is critical.

As an integral part of ASIC design, logic circuit contains a large number of logic circuits in ASIC. ASIC stands for Application-Specific Integrated Circuit, which is different from general-purpose IC (Integrated Circuit), and this IC are manufactured to realize specific functions. Most ASIC technologies use standard cells, which are predesigned logic blocks consisting of one to a few logic gates. ASIC cell library may have hundreds of standard cells, such as AND, NAND, OR, NOR, Exclusive-OR, Exclusive-NOR, D flip-flop, latch, etc. In the ASIC design process, the design of the logic circuit is a very important step. Logic designers need to ensure the correctness and performance of the design through logic simulation and verification, because the optimization of logic circuits can lead to better performance and smaller area, thereby

reducing cost and power consumption.

Since we will involve a large number of operations when designing the solver, and these operations are basically completed by the full adder and the shift register, we will explain the full adder and the shift register here. In digital circuits, the operation of temporarily storing the binary data or code being processed is called registering. In any modern digital circuit system, especially some large-scale digital processing systems, it is often impossible to process all the data at one time, so some data codes that need to be processed must be registered in the process of processing. So that it can be used anytime when needed. Registers are also divided into different types according to their functions, for example, general-purpose registers for arithmetic units, address memory for specifying memory addresses, and memory registers for temporary storage when inputting and outputting data from memory.

We will describe the shift register in detail here. A shift register is a register with a shift function. Shifting one binary digit to the left results in doubling, while shifting one binary digit to the right halves it. And all bits are shifted left or right synchronously with the system clock. Since this function can be applied to multiplication and division, it can be used by adding it to a general-purpose register or the accumulator of the arithmetic unit. Since the shift register shifts in synchronization with the system clock, using D-FF or JK-FF, data input to the shift register can be input serially starting from the least significant bit, or can be input in parallel with each FF. When outputting data, there is a serial output for the most significant bit and a parallel output for each bit. By combining these functions, it can be used as a serial/parallel converter or a parallel/serial converter for data input/output.

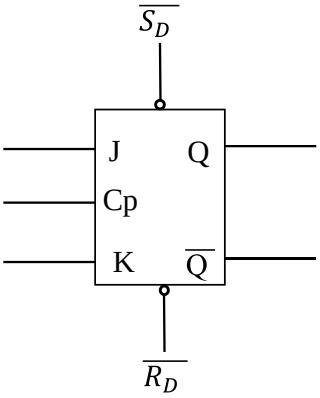


Fig. 3.4.1 The JK-FF

There is a JK-FF as shown in Fig.3.4.1, in the JK-FF, J and K can be input at the same time, we use JK-FF to design a 4-bit right shift register as an example to illustrate the shift register. And this designed shift-register as shown in Fig.3.4.2.

In electronics, an adder is a digital circuit component used to perform add operations, and is the basis of the arithmetic logic unit in the core microprocessor of an electronic computer. The half adder is an adder that does not consider the carry of the previous stage, but obtains the sum of a binary number and the carry of the next stage, which will be used

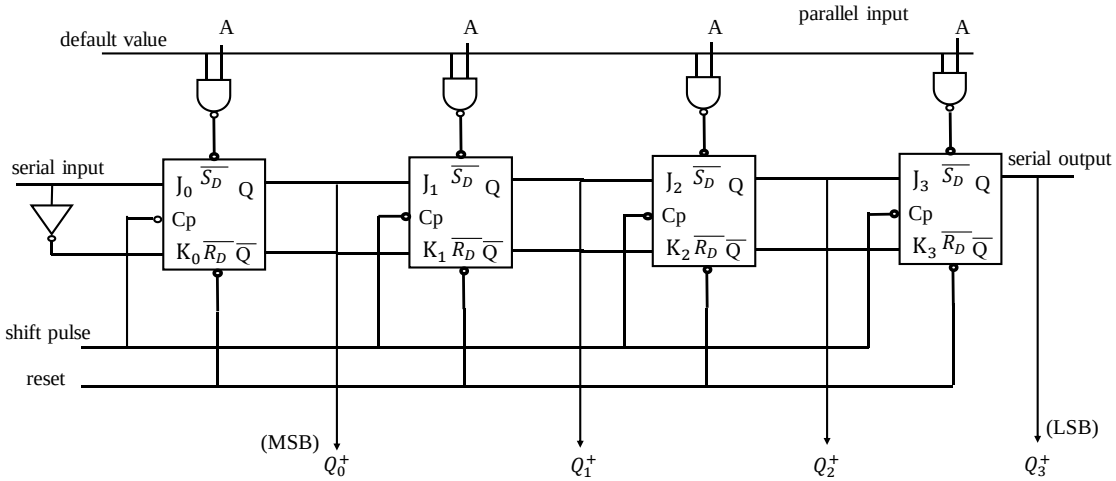


Fig. 3.4.2 The shift-register

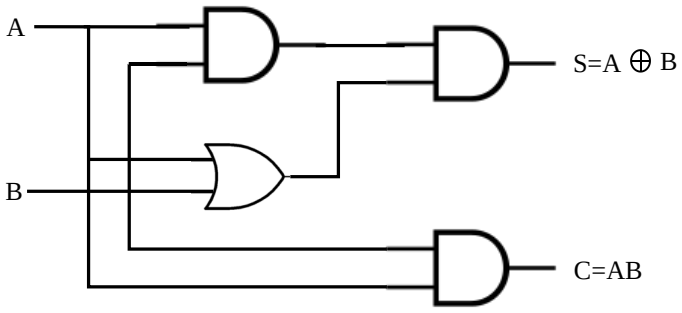


Fig. 3.4.3 The circuit of half-adder

Fig. 3.4.4 The symbol of half-adder

in the design of the full adder. The circuit and symbol of the half adder is shown in the Fig.3.4.3 and Fig.3.4.4 below. And then, we consider to design a full adder based on a one bit half-adder, the full adder is an adder that takes into account the carry of the previous stage. Fig.3.4.5 shows the circuit of the full adder. C is the carry from the previous stage, and S is the sum.

Add two one-bit binary numbers, and output the sum and carry out according to the received low-order carry signal. The three inputs of the full adder are two addends A and

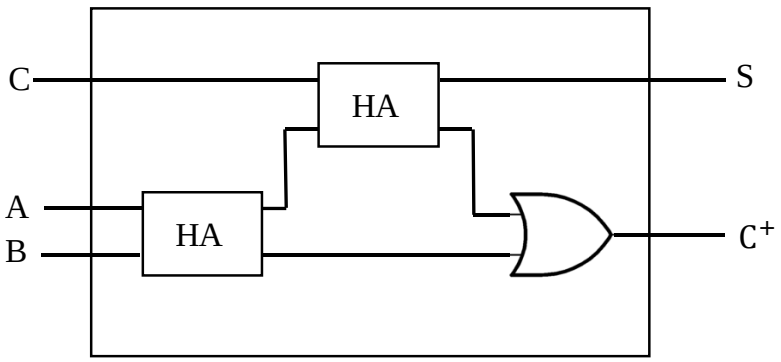


Fig. 3.4.5 The circuit of full-adder

B and the low-order carry C. The full adder can usually be cascaded to form the basic of a multi-bit (8-bit, 16-bit, 32-bit) binary number adder part. The difference between the full adder and the half adder is that the full adder can receive a low-order carry input signal C.

3.4.2 Integer and floating format

In this section, we explain how numbers are represented in computers. There are two types of methods for processing numerical values inside a computer, floating-point and fixed-point. The difference between the two is that the decimal point of floating-point numbers is floating, and the decimal point of fixed-point numbers is fixed.

Floating-point values are approximate representations of any real number in a computer. Specifically, this real number is obtained by multiplying an integer or fixed-point number (that is, the mantissa) by the integer power of a certain base (usually use binary in computers). This representation method is similar to scientific notation with a base of decimal. Floating point numbers are represented as follows,

$$N=M\times R^{E} \tag{3.4.1}$$

In any such system, we choose a base R and a precision P. M (ie, the mantissa) is P digits of the form $\pm \mathrm{d.ddd}\ldots\mathrm{ddd}$. If the first bit of M is a non-zero integer, M is said to be normalized. There are some descriptions that use a single sign bit (S stands for + or -) to signify, such that M must be positive, E is the exponent. As shown in the following

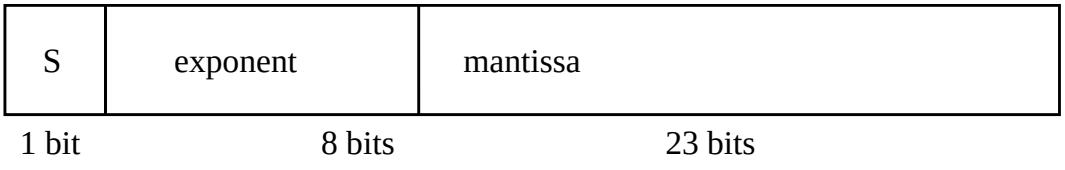


Fig. 3.4.6 IEEE Floating point representation

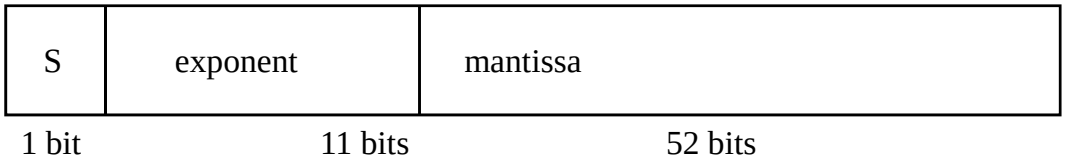


Fig. 3.4.7 IEEE Double precision floating point representation

formula, 31245.0 is the floating-point number *N*, 3.1245 is the mantissa *M*, 10 is the base, and 3 is the exponent.

$$31245.0=3.1245\times 10^{3} \tag{3.4.2}$$

The floating-point number format defined by the IEEE 754 standard is as shown in Fig.3.4.6 and Fig.3.4.7. This is a Float-type in the Fig.3.4.6. For Float-type floating point, the MSB is the sign bit, the exponent bit is 8bits, and the mantissa is 23 bits, 32 bits in total. Since the mantissa is normalized, the MSB of mantissa must be non-zero, and the MSB is hidden, therefore the mantissa is actually 24bits. And then this is a Double type in the Fig.3.4.7. For Double-type floating point, the MSB is sign bit, the exponent is 11bits, and the mantissa is 52 bits, 64 bits in total.

Next, we will explain fixed-point values. The decimal point of floating-point values changes with the exponent, so the range of decimals that can be expressed by floating-point values is very wide, but the amount of calculation of floating-point values is very

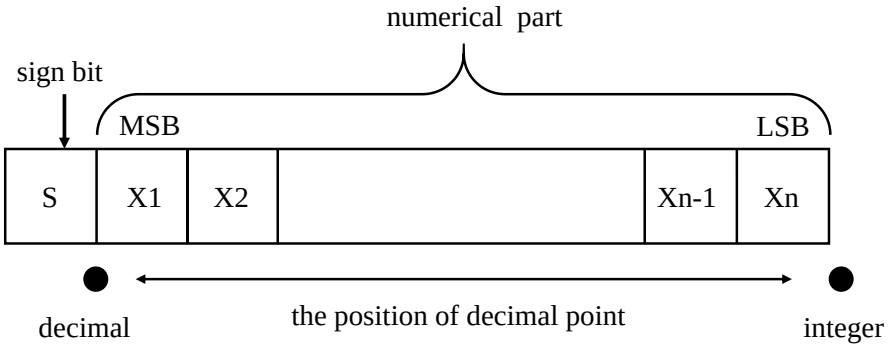


Fig. 3.4.8 Signed representation of fixed-point values

large (we can get this according to its definition). The decimal point of a fixed-point number is fixed at one position and does not occupy a single bit. As shown in the figure below, the decimal point of a fixed-point number is fixed between the sign bit and the most significant bit (MSB), and the decimal point of a pure integer is fixed after the lowest significant bit (LSB).

Methods of handling positive and negative numbers in the fixed-point representation system include absolute value representation, bias representation, and complement representation. The absolute value representation is a method in which the MSB is a sign bit, and when the value of the MSB is 0, it represents a positive number, and when it is 1, it represents a negative number. When performing subtraction in terms of absolute values, a comparison circuit, a multiplexer circuit for data exchange, a sign determination circuit, will be required to realize the subtraction operations, so the circuit scale of the arithmetic unit is increased accordingly.

Bias expression is the conversion of a negative number into a positive number by adding a bias value. The bias value is chosen to be the smallest negative absolute value. However, it is difficult to read the numerical value intuitively, and similar to the absolute value expression, additional circuits are required to implement addition and subtraction

positive value		negative value	
decimal	2’s complement	decimal	2’s complement
1	0001	-1	1111
2	0010	-2	1110
3	0011	-3	1101
4	0100	-4	1100
5	0101	-5	1011

Fig. 3.4.9 The two’s complement

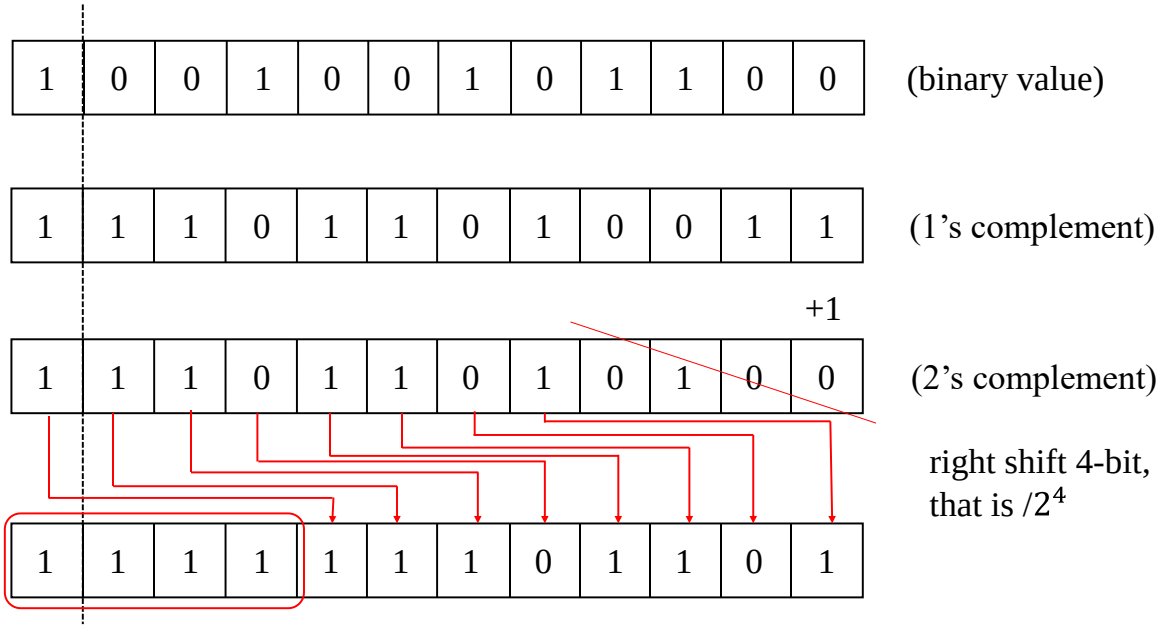


Fig. 3.4.10 The right shift process of negative value based on 2's complement

operations.

There are two types of complement of binary number, one' s complement and two's complement. We can get the one's complement of a binary number by inverting the given number simply. To get the two's complement of binary number is that one's complement number of given number plus one to the LSB. Find two's complement of each decimal numbers as shown in Fig.3.4.9.

In computers, data is stored in complement code. Next, we perform a shift on the basis of the complement to realize the multiplication and division operation.

Regarding shifting, we need to pay special attention to positive numbers. No matter left shifting or right shifting, they are complemented with 0, and the complement of negative numbers needs attention. Left shifting needs to complement 0 on the right, and right shifting needs to complement 1 on the left. The Fig.3.4.10 below is an example of division by shifting a negative number to the left.

3.4.3 Hardware Description Languages (HDL)

Design methods using HDL (Hardware Description Languages) have been widely used in the design of large-scale integrated circuits such as ASIC (Application Specific Integrated Circuit), and the design is not limited to ASIC, but also has many advantages for relatively small-scale designs using FPGA, PLD, etc. The Fig.3.4.11 below shows the comparison between logic circuit input design and HDL design.

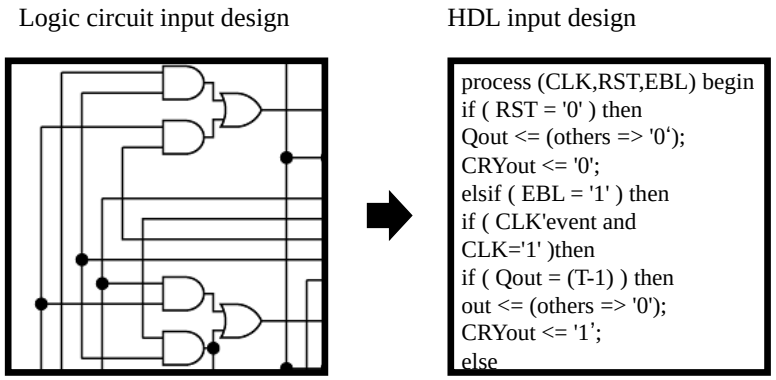


Fig. 3.4.11 Logic circuit input design and HDL input design

HDL design does not need to consider complex formulas by designing at a higher level of abstraction. This frees designers from the burden of design and shortens the design cycle. In addition, a high level of abstraction makes it easier to change the design, allowing designers to build more complete systems.

description language name	vendor	features
VHDL	U.S. Department of Defense	A wide range of fields can be described
Verilog-HDL	As a language for simulator	A wide range of fields can be described, but the description ability is not as high as VHDL
SFL	As a language for PARTHENON system	Can only be written in RTL
UDL/I	Japan Electronic Industry Development Association	Can only be written in RTL

Fig. 3.4.12 Comparison of various HDL

Hardware description languages include VHDL (VHSIC HDL), Verilog-HDL, UDL/I(Unified Design Language for Integrated Circuit), and SFL (Structured Function Description language). Each function is shown in the Fig.3.4.12, and each description language has its advantages and disadvantages, but among them, VHDL is the most widely used as an industry standard.

VHDL is a highly descriptive language that can be described at different levels. Describe the algorithm of the entire system (architecture-level description), by modeling hard disk data exchange and motor control, etc., describe the entire system at a higher abstraction level, or at a level (RTL: Register Transfer Level) that allows logic circuit generation. Of course, logical gates can also be described. In the actual large-scale design, CPU, communication, image processing design, etc., first we describe the architecture-level and behavior-level, and verify the entire system. Design efficiently by catching system errors earlier. Afterwards, the parts of the verification system to be converted to ASICs are rewritten at the logic generation level (RTL), and logic circuits are generated.

When designing with VHDL, there are mainly four units that will be used as the following:

(1) Entity

(2) Architecture

(3) Configuration

(4) Package

Entity as mentioned above is that to define the interface (inputs and outputs) as shown in the Fig. 3.4.13. Entity can be viewed as a black box, before we define what happens inside the black box, we know what are my inputs and outputs, so that we can write entity because entity tells us who are the members in the “house”. That is, entity helps us define

input and output ports and also specify what type of ports they will be.

Architecture describes the innards of circuit. What actually is happening from those

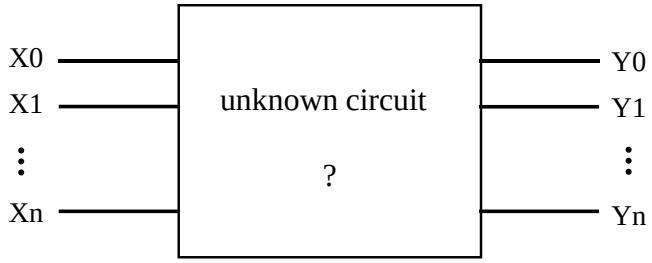


Fig. 3.4.13 The Entity of VHDL

inputs and how it generates outputs. That is, Architecture let us get the information of unknown circuit in Fig.3.4.12. To model the architecture of an Entity, we have several modeling methods:

(1) Dataflow

(2) Behavioral

(3) Structural

(4) Mixed

If we have an entity, we can map multiple architectures to same one Entity. So this one Entity how does know which architecture to map. And then we can introduce the Configuration. We specify that in Configuration file in Configuration part of the VHDL design we specify which Entity maps to which Architecture.

The Package as mentioned above, it reads everything defined in the std_logic_1164 and std_logic_unsigned packages in the VHDL standard library IEEE. Required for type declarations and for using operations.

Chapter 4

Dataflow Architecture Dedicated Computer Based on FIT Scheme for Electromagnetic Field Simulations

In this Chapter, we introduce the simulation of FIT dataflow machine based on fundamental theories and formulations described in Chapter3.

4.1 Dedicated Computer for 2-D Magneto-static Field Simulation

In this section, the design of an overview architecture of dataflow machine for 2-D magneto-static field and its specific circuit design including matrix solver will be explained. As mentioned in Chapter 3.1.2, the vector Poisson's equation is the basis of Magneto-static field. Then, the calculation of 2-D magneto-static field is simulated on the basis of FIT (Chapter 3.2.2). After the whole grid space of the whole grids space of 2-D magneto-static field is discretized, the distribution of magnetic potential in the form of matrix equation is determined. To solve this matrix equation, a stable and quick convergence iterative method is considered, that is BiCG-Stab (Chapter 3.3.3), and the implementation of the BiCG-Stab matrix solver scheme on the hardware circuit (Chapter 3.4) is discussed. Finally, the FIT dataflow machine intended for 2-D magneto-static field simulation is designed through the hardware description language VHDL (Chapter 3.4.2).

4.1.1 FIT Scheme for 2-D Magneto-static Field

The behavior of magneto-static field can be described by the vector Poisson's Equation,

$$\nabla \times \left(\frac{1}{\mu}\nabla \times \mathbf{A}\right) = \mathbf{J} \tag{4.1.1}$$

$$\mathbf{B}=\nabla \times \mathbf{A} \tag{4.1.2}$$

where, μ represents the magnetic permeability, **A** represents the vector potential, **J** represents the current density, and **B** represents the magnetic induction. In case of 2-D magneto-static field, which means $\mathbf{B}_z = 0$, the following equation holds,

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_x \\ \mathbf{B}_y \\ \mathbf{B}_z \end{pmatrix} = \begin{pmatrix} \frac{\partial A_z}{\partial y} - \frac{\partial A_y}{\partial z} \\ \frac{\partial A_x}{\partial z} - \frac{\partial A_z}{\partial x} \\ \frac{\partial A_y}{\partial x} - \frac{\partial A_x}{\partial y} \end{pmatrix} = \begin{pmatrix} \frac{\partial A_z}{\partial y} \\ -\frac{\partial A_z}{\partial x} \\ 0 \end{pmatrix} \tag{4.1.3}$$

It can be found out from the above equation that there is only the z-component of magnetic vector potential **A** in the 2-D case. The integral form of the Equation (4.1.1) is as follows,

$$\oint_c \left(\frac{1}{\mu}\nabla \times \mathbf{A}\right) \cdot d\mathbf{l} = I \tag{4.1.4}$$

where, I represents the current. The Finite Integration Technique (FIT) scheme for 2-D magneto-static field simulation is considered. With discretization performed in a grid space for the above Equation (4.1.4) as shown in Fig. 4.1.1. the following equation can be obtained,

$$\Delta l(H_{xi,j-1} + H_{yi,j} - H_{xi,j} - H_{yi-1,j}) = J_{i,j}\Delta l^2 \tag{4.1.5}$$

where $H_{xi,j}$ and $H_{yi,j}$ are the magnetic field components around a grid (i,j), respectively. According to Equation (4.1.4) along the integral path C in Fig.4.1.1, Equation (4.1.6) can be rewritten as,

$$c_{i,j}^0 A_{zi,j} - c_{i,j}^1 A_{zi,j-1} - c_{i,j}^2 A_{zi+1,j} - c_{i,j}^3 A_{zi,j+1} - c_{i,j}^4 A_{zi-1,j} = J_{i,j}\Delta l^2 \tag{4.1.6}$$

where,

$$c_{i,j}^{1}=\frac{1}{2\mu_{i,j-1}}+\frac{1}{2\mu_{i-1,j-1}},c_{i,j}^{2}=\frac{1}{2\mu_{i,j-1}}+\frac{1}{2\mu_{i,j}},$$
$$c_{i,j}^{3}=\frac{1}{2\mu_{i,j}}+\frac{1}{2\mu_{i-1,j}},c_{i,j}^{4}=\frac{1}{2\mu_{i-1,j}}+\frac{1}{2\mu_{i-1,j-1}},\qquad(4.1.7)$$
$$c_{i,j}^{0}=c_{i,j}^{1}+c_{i,j}^{2}+c_{i,j}^{3}+c_{i,j}^{4}$$

To construct the matrix equation based on the discretization Equation (4.1.6) for all grid points as shown in Fig.4.1.2. The final FIT matrix equation can be obtained. Then, the information about distribution of the magnetic field can be obtained by solving the matrix equation in Fig.4.1.2.

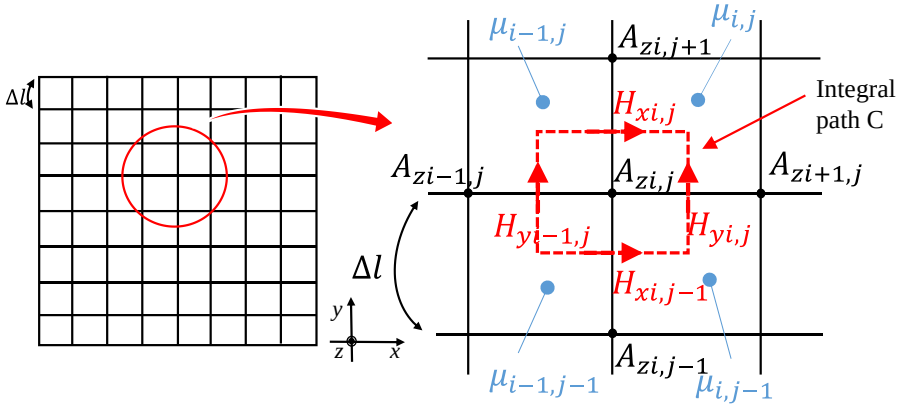


Fig. 4.1.1 FIT discretization in 2-D grid space

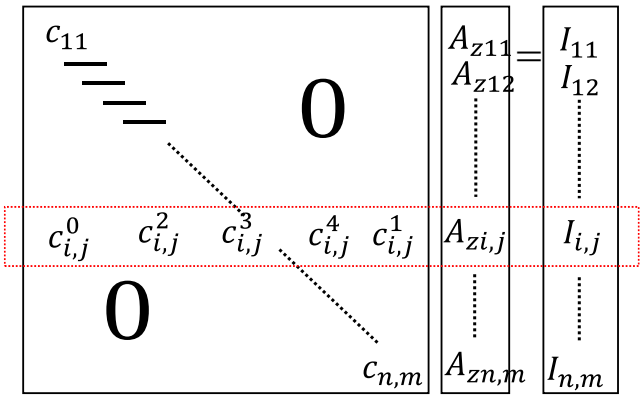


Fig. 4.1.2 FIT scheme matrix equation

4.1.2 Hardware Circuit of BiCG-Stab Matrix Solver

To solve the large sparse matrix equation, the BiCG-Stab scheme is adopted. In addition, detail digital circuits of the BiCG-Stab matrix solver based on dataflow architecture are designed.

In this study, the iterative process of BiCG-Stab scheme to solve matrix equation of $A\mathbf{x}=\mathbf{b}$ is expressed as follows,

$\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$, $\mathbf{x}_0$ is an initial guess,

$\mathbf{r}_0^*, (\mathbf{r}_0^*, \mathbf{r}_0) \neq 0, e.g., \mathbf{r}_0^* = \mathbf{r}_0$, set $\beta_{-1} = 0$,

for n=1, 2, 3……until $\|\mathbf{r}_n\| \leq \varepsilon\|\mathbf{b}\|$,

begin

$$
\begin{aligned}
&\mathbf{p}_n = \mathbf{r}_n + \beta_{n-1}(\mathbf{p}_{n-1} - \zeta_{n-1}A\mathbf{p}_{n-1}) && \text{(i)}\\
&\alpha_n = \frac{(\mathbf{r}_0^*, \mathbf{r}_n)}{(\mathbf{r}_0^*, A\mathbf{p}_n)} && \text{(ii)}\\
&\mathbf{t}_n = \mathbf{r}_n - \alpha_n A\mathbf{p}_n && \text{(iii)}\\
&\zeta_n = \frac{(A\mathbf{t}_n, \mathbf{t}_n)}{(A\mathbf{t}_n, A\mathbf{t}_n)} && \text{(iv)}\\
&\mathbf{x}_{n+1} = \mathbf{x}_n + \alpha_n\mathbf{p}_n + \zeta_n\mathbf{t}_n && \text{(v)}\\
&\mathbf{r}_{n+1} = \mathbf{t}_n - \zeta_n A\mathbf{t}_n && \text{(vi)}\\
&\zeta_n = \frac{(A\mathbf{t}_n, \mathbf{t}_n)}{(A\mathbf{t}_n, A\mathbf{t}_n)} && \text{(vii)}
\end{aligned}
\qquad (4.1.8)
$$

end

In the BiCG-Stab scheme (4.1.8), initial value should be set, before the iterative process ((i) ~ (vii)) of (4.1.8). After the setting of an appropriate initial value of $\mathbf{x}_0$, the initial residual $\mathbf{r}_0$ and the initial shadow residual $\mathbf{r}_0^*$, the iterative process ((i) ~ (vii)) of BiCG-Stab starts. Also, the iterative process ((i) ~ (vii)) is repeated until the residual $\mathbf{r}_{n+1}$ converges and become sufficiently small.

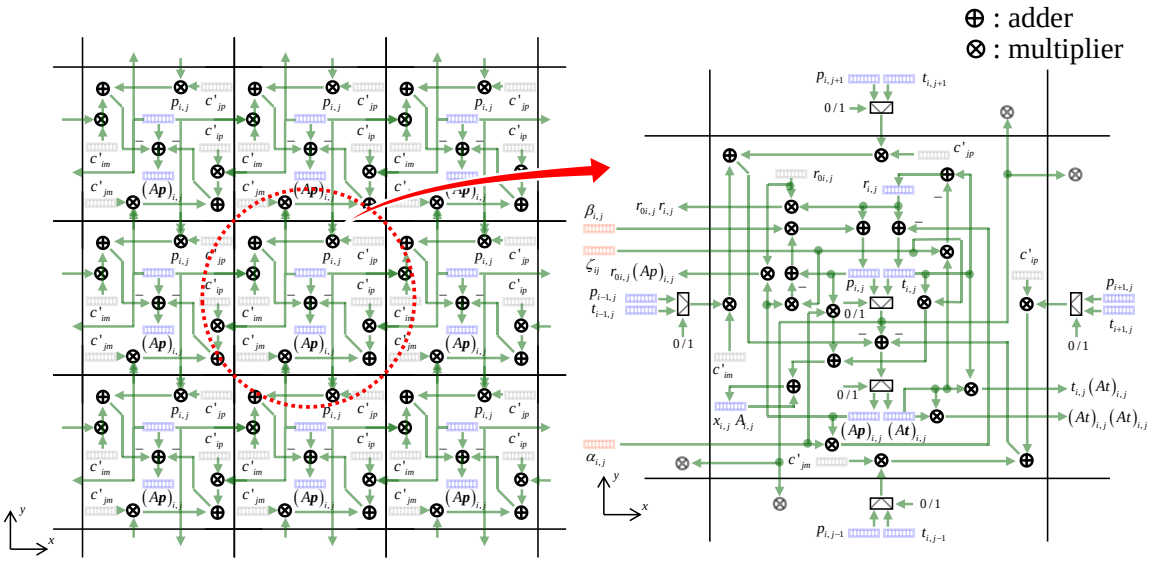
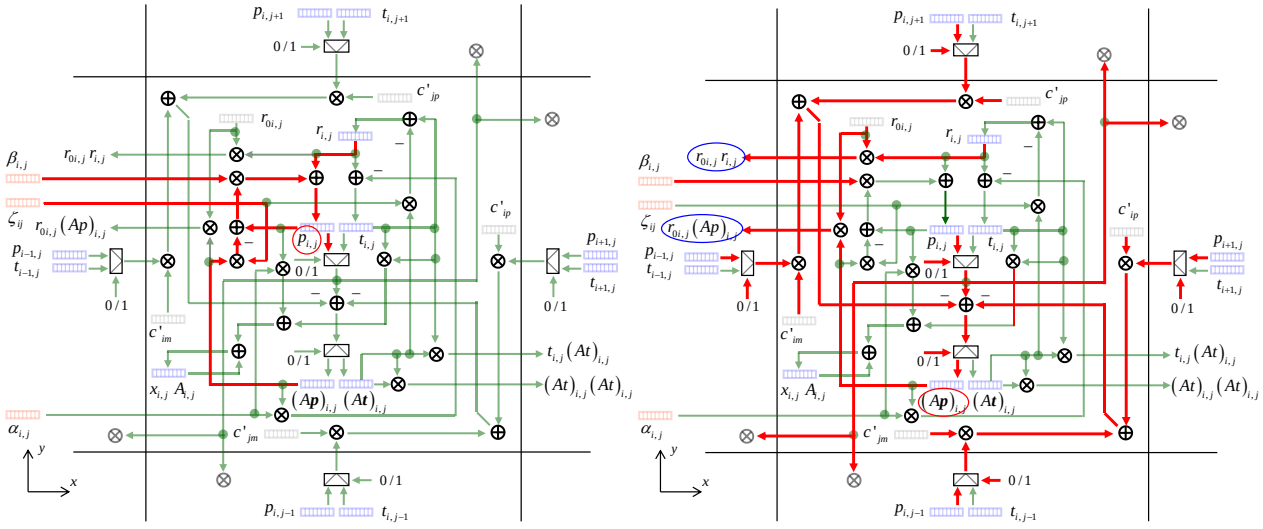


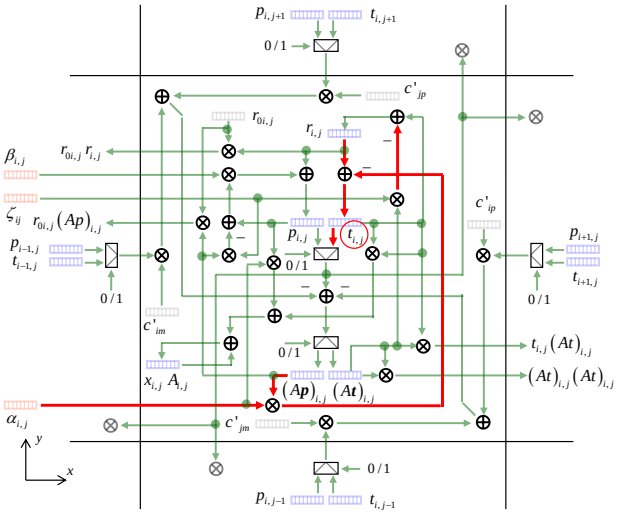
Fig. 4.1.3 FIT scheme grid circuit based on BiCG-Stab scheme

In the design of dedicated computer, one of our purposes is potability, which means, a smaller size hardware is required. Since the use of floating-point arithmetic in the hardware circuits can result in a large size hardware, it is considered to use the fixed-point arithmetic with integer format to instead of the floating-point arithmetic. In addition, during the process of BiCG-Stab iteration, the range of dynamic change of the value

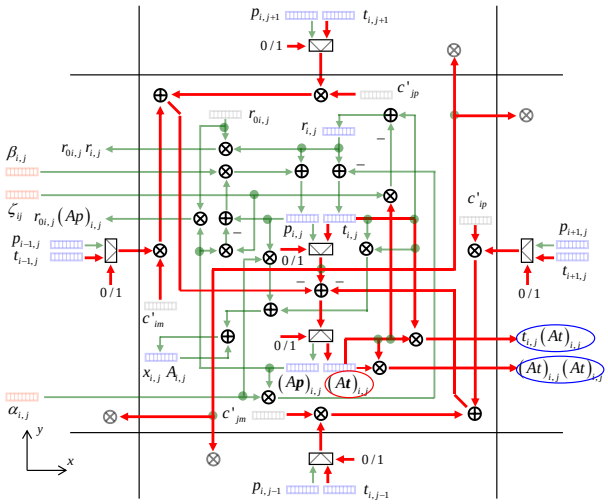


(a) Iteration process (i) in (4.1.8)

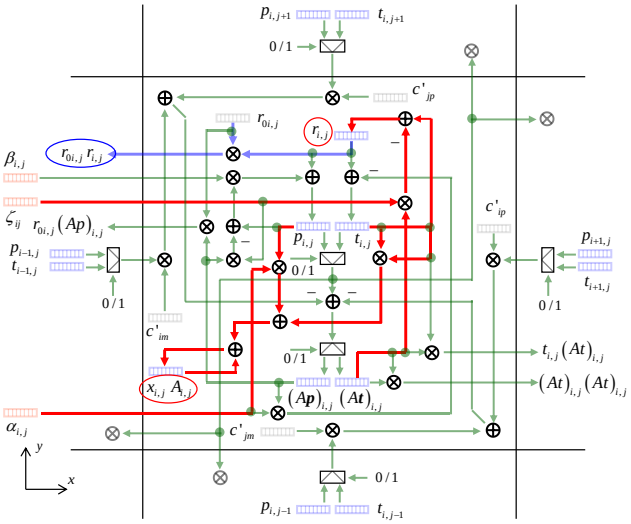
(b) One component of $A\mathbf{p}_n$



(c) Iteration process (iii) in (4.1.8)



(d) One component of $A\mathbf{t}_n$



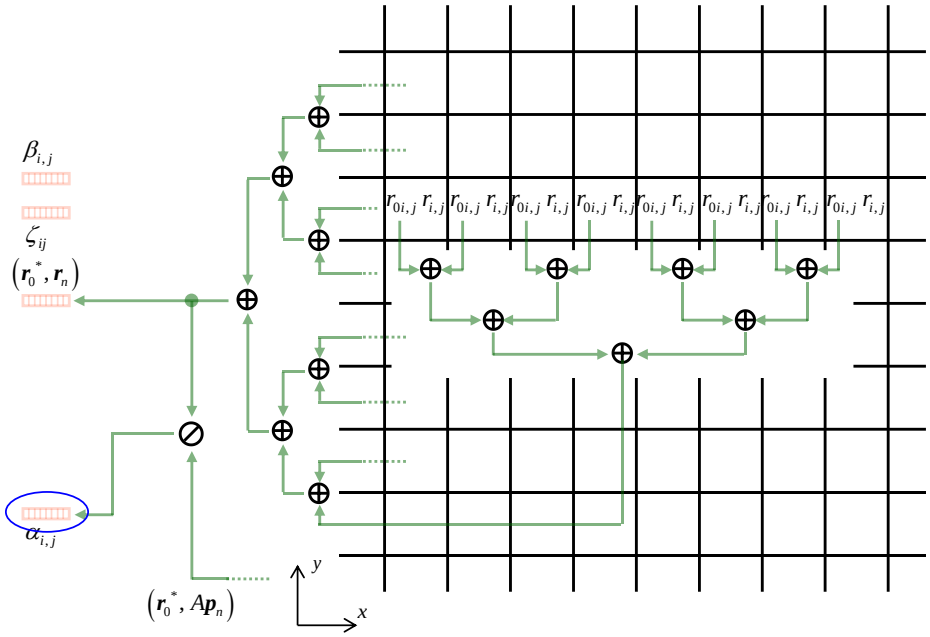
(e) Iteration process (v) and (vi) in (4.1.8)

Fig.4.1.4 Individual circuit operations of BiCG-Stab scheme

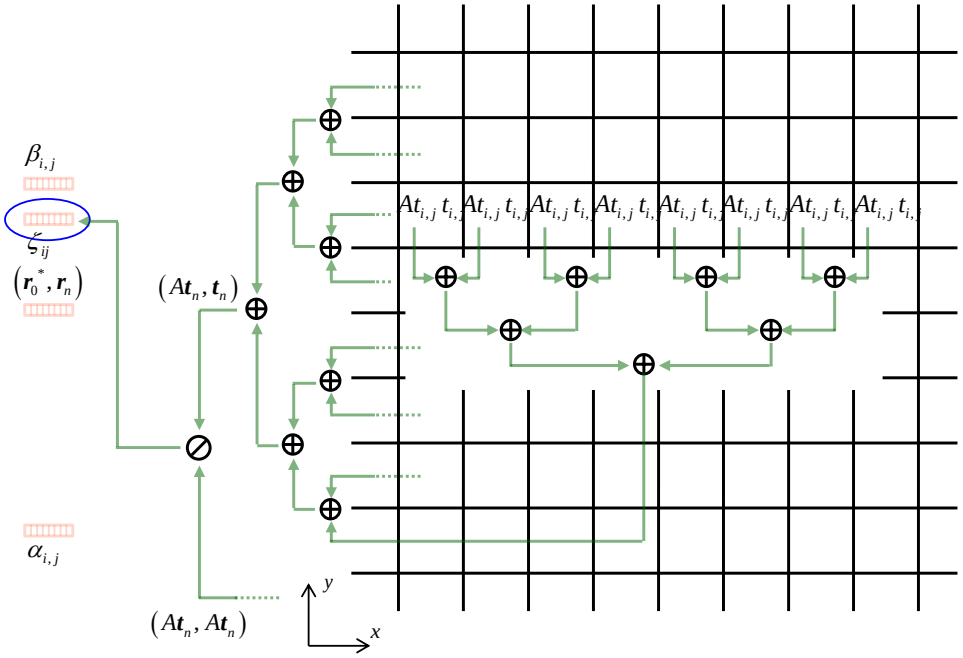
exceeds 10^{22}, to keep the sufficient precision in the BiCG-Stab matrix solver, we employ 200-bit registers to store all unknown values. The overview of FIT scheme grid circuit based on BiCG-Stab matrix solver is shown in Fig4.1.3. Then, the detailed circuits of the unit FIT grid based on BiCG-Stab scheme are explained, as shown in Fig.4.1.4. In the BiCG-Stab matrix solver circuits, all of the unknown values ($\mathbf{p}_n$, $\mathbf{t}_n$, $\mathbf{x}_n$, $\mathbf{r}_n$, $A\mathbf{p}_n$, $A\mathbf{t}_n$) of

the BiCG-Stab iterative process ((i) ~ (vii)) in (4.1.8) and the coefficients $(c_{i,j}^0, c_{i,j}^1, c_{i,j}^2, c_{i,j}^3, c_{i,j}^4)$ in (4.1.6) are stored in the registers at each grid, and these registers are connected with each other according to the arithmetic circuit. The highlighted parts of Fig.4.1.4 (a), (c) and (e) correspond to iterative process (i), (iii) and (vii), while these logic circuits can execute the arithmetic operations in parallel with one clock cycle. It should be noted that the highlighted parts of Fig.4.1.4 (b) and (d) are one component of matrix-vector products $A\mathbf{p}_n$ and $A\mathbf{t}_n$, while the one component of $A\mathbf{p}_n$ and $A\mathbf{t}_n$ is executed by the logic circuit with one clock cycle. In addition, the other components in all the grids space of $A\mathbf{p}_n$ and $A\mathbf{t}_n$ can be executed in parallel by synchronizing with one clock cycle.

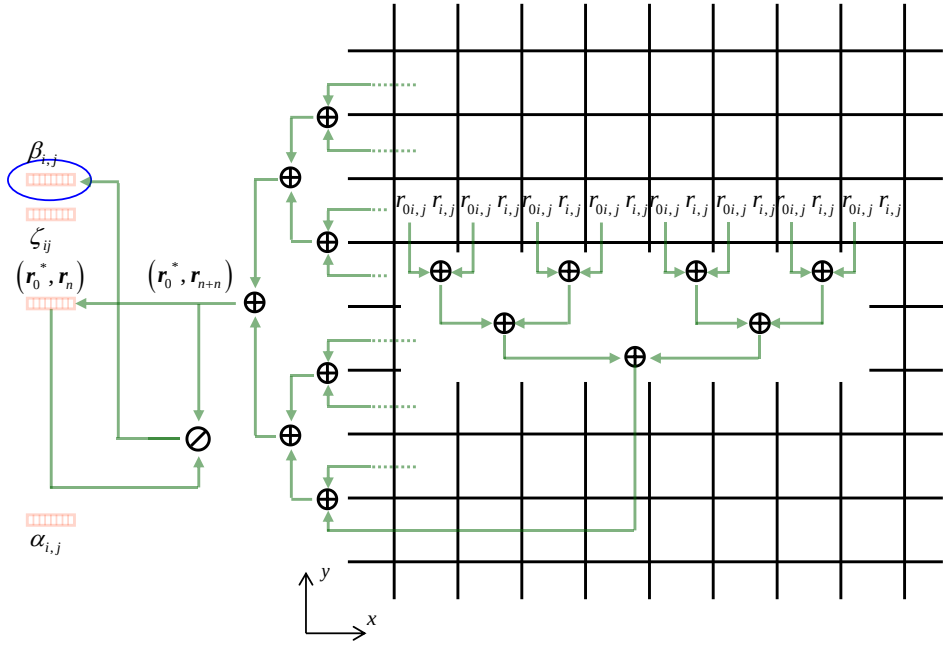
The inner product $(\mathbf{r}_0^*, \mathbf{r}_n)$, $(\mathbf{r}_0^*, A\mathbf{p}_n)$, $(A\mathbf{t}_n, \mathbf{t}_n)$, $(A\mathbf{t}_n, A\mathbf{t}_n)$ is collected after each component is executed at the individual grid circuit of Fig.4.1.3. Then, these multiplications of each grid circuit are summed up to compute α_n, ζ_n and β_n as shown



(a) Inner product $(\mathbf{r}_0^*, \mathbf{r}_n)$, $(\mathbf{r}_0^*, A\mathbf{p}_n)$ for α_n in (4.1.8)



(b) Inner product $(A\mathbf{t}_n, \mathbf{t}_n)$, $(A\mathbf{t}_n, A\mathbf{t}_n)$ for ζ_n in (4.1.8)



(c) Inner product $(\mathbf{r}_0^*, \mathbf{r}_n)$, $(\mathbf{r}_n, \mathbf{r}_{n+1})$ for β_n in (4.1.8)

Fig. 4.1.5 Summation of inner product component

in Fig.4.1.5, corresponding to the iterative process (ii), (iv) and (vii) in (4.1.8), respectively. So far, the design of detailed circuit of BiCG-Stab matrix solver has been explained. Next, the whole configuration of FIT dataflow machine for 2-D magneto-static

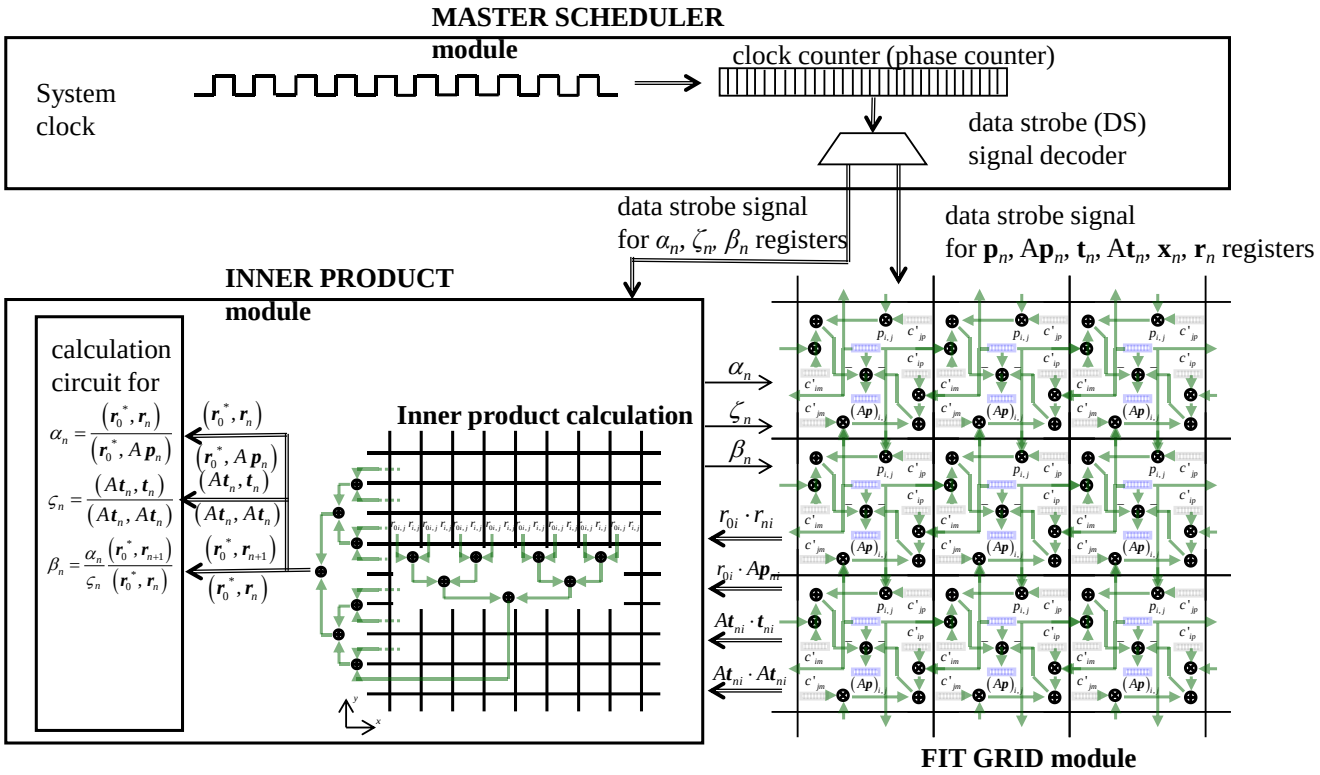


Fig.4.1.6 The whole configuration of FIT dataflow machine

field simulation on hardware circuit will be introduced.

Fig.4.1.6 shows the whole configuration of FIT dataflow machine. This dataflow machine consists of 3 parts: MASTER SCHEDULER module, INNER PRODUCT module and FIT GRID module. As explained, the FIT GRID module (Fig.4.1.3) executes the BiCG-Stab iterative process ((i) ~ (vii)) in (4.1.8) for the FIT matrix equation (Fig.4.1.2). The INNER PRODUCT module collects the multiplications of $r_{0i,j}r_{i,j}$, $r_{0i,j}A\mathbf{p}_{i,j}$, $t_{i,j}A\mathbf{t}_{i,j}$, $A\mathbf{t}_{i,j}A\mathbf{t}_{i,j}$, and these multiplications are summed up to compute the inner product of $(\mathbf{r}_0^*, \mathbf{r}_n)$, $(\mathbf{r}_0^*, A\mathbf{p}_n)$, $(A\mathbf{t}_n, \mathbf{t}_n)$, $(A\mathbf{t}_n, A\mathbf{t}_n)$ with one clock cycle. Then the α_n, ζ_n and β_n are computed and sent back to FIT GRID module for the computations of (i), (iii), (v) and (vii) in (4.1.8). The execution of FIT GRID module and INNER RPODUCT module is controlled by the data strobe (DS) signals from MASTER

SCHEDULER module.

4.1.3 Division Circuit with Multiple Precision Integer Format

The hardware implementation of the FIT method scheme is not suitable for the floating-point calculations with a large hardware size, as mentioned in Chapter 3.4.2. Also, the hardware implementation of the FIT scheme and matrix calculations in fixed-point arithmetic with integer format calculations is essential. In particular, the division calculations using the BiCG-Stab method require at least double-precision calculations, which must be performed through integer format operations without degrading precision. In addition, VHDL can not support the integer calculations beyond 32 bits. In this research, it is considered to design a division circuit with multiple precision integer formats on the hardware.

In this research, we examine a circuit that executes arbitrary multiple-precision integer division in one clock without depending on precision, which is based on this standard 32-bit integer division in VHDL. The division circuit proposed in this research is explained using the example of division between 96 bits (equivalent to 29 decimal digits) shown in Fig.4.1.7. Firstly, the 96-bit divisor (den) and dividend (num) are divided into 16-bit units and express as follows:

$$\text{num}=n0 \times 2^{80} + n1 \times 2^{64} + n2 \times 2^{48} + n3 \times 2^{32} + n4 \times 2^{16} + n5 \times 2^{0} \tag{4.1.9}$$

and

$$\text{den}=d0 \times 2^{80} + d1 \times 2^{64} + d2 \times 2^{48} + d3 \times 2^{32} + d4 \times 2^{16} + d5 \times 2^{0} \tag{4.1.10}$$

that is to say, this division num/den=(ND) can be written as follows,

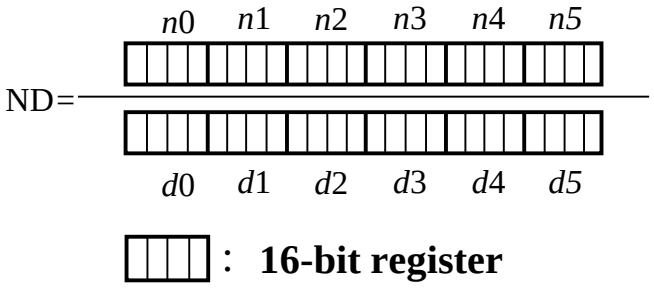


Fig.4.1.7 16-bit register allocation for multiple precision division

$$\mathrm{ND} = \frac{\mathrm{num}}{\mathrm{den}} \tag{4.1.11}$$

Then, the first term $d0 \times 2^{80}$ is divided by the numerator and denominator at the same time, to obtain the following equatuion,

$$\mathrm{ND} = \frac{\dfrac{\mathrm{num}}{d0 \times 2^{80}}}{\dfrac{\mathrm{den}}{d0 \times 2^{80}}} \tag{4.1.12}$$

For the numerator, VHDL standard 32-bit division can be used to calculate $(n0 \times 2^{16} + n1)/d0$,

$$n0 \times 2^{16} + n1 \equiv nq(1) \times d0 + nr(1) \tag{4.1.13}$$

where, the quotient $nq(1)$ and the reminder $nr(1)$ are calculated. Then, the remainder $nr(1)$ (16-bit or less) with the next lower 16-bit $n2$ forms a new 32-bit value, $(nr(1) \times 2^{16} + n2)$, and this value is used to divide $d0$ as follows,

$$nr(1) \times 2^{16} + n2 \equiv nq(2) \times d0 + nr(2) \tag{4.1.14}$$

where, the quotient $nq(2)$ and the reminder $nr(2)$ are calculated, as shown in Fig.4.1.8. By repeating this process, the numerator $\frac{\mathrm{num}}{d0 \times 2^{80}}$ of (4.1.12) can be rewritten as,

$$\begin{aligned} N = (&nq(1) \times 2^{64} + nq(2) \times 2^{48} + nq(3) \times 2^{32} + nq(4) \times 2^{16} \\ &+ nq(5) \times 2^{0}) \times 2^{-80} \end{aligned} \tag{4.1.15}$$

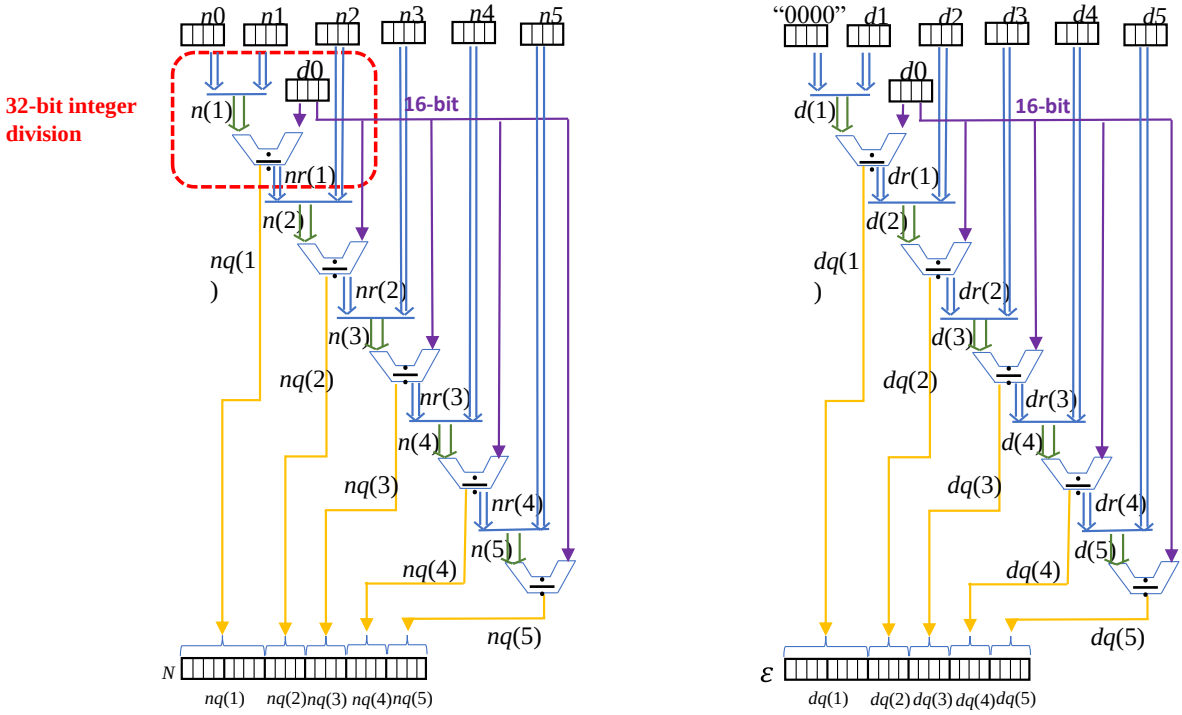


Fig.4.1.8 Multiple precision division circuit

and similarly, when calculating the second term in the denominator of (4.1.12), we can get,

$$\varepsilon = (dq(1) \times 2^{64} + dq(2) \times 2^{48} + dq(3) \times 2^{32} + dq(4) \times 2^{16} + dq(5) \times 2^{0}) \times 2^{-80} \tag{4.1.16}$$

where dq(i) represents the quotient (16 bits) of each 32-bit division. From the above, the (4.1.12) can be rewritten as,

$$\mathrm{ND} = \frac{N}{1+\varepsilon} \tag{4.1.17}$$

where, $\varepsilon \ll 1$, so that (4.1.17) can be expanded as follows,

$$\mathrm{ND} = N \times (1-\varepsilon)(1+\varepsilon^2+\varepsilon^4+\varepsilon^6+\cdots) \tag{4.1.18}$$

By truncating the infinite series with sufficient precision required for the computation, the division in (4.1.11) can be performed with only 32-bit division. For example, it is sufficient to use up to the term that has the precision equivalent to double precision for

BICG-Stab matrix calculation. At that time, since the series of division processing in Fig. 4.1.8 does not include any registers and is composed of combinational circuits in principle, this division processing can be executed in one clock. Also, in the variants from (4.1.11) to (4.1.12), the value of the denominator den is not necessarily large enough to occupy the entire register, especially when most of the high-order bits are zero, *d*0 are all 0 in 16 bits, and (4.1.12) cannot be calculated. For this reason, in general, the denominator is first carried until the most significant bit (MSB) of the denominator den becomes non-zero before processing (4.1.12). That is to say, the shift left by the number of zeros is in the high-order bits. Then, it is stored in another register (Fig.4.1.8). In Fig.4.1.9, there are k bits of 0 in the high-order bits of the denominator den, and the left-justified value is the temporary divisor (dent),

$$\text{dent}=dt0\times 2^{(96-(k+16))}+dt1\times 2^{(96-(k+32))}+dt2\times 2^{(96-(k+48))}+ dt3\times 2^{(96-(k+48))}+dt4\times 2^{0}+ \tag{4.1.19}$$

and then, the (4.1.12) can be rewritten as,

$$\text{ND}=\frac{\dfrac{\text{num}}{dt0\times 2^{(96-(k+16))}}}{1+\dfrac{dt}{dt0\times 2^{(96-(k+16))}}} \tag{4.1.20}$$

By applying the process after (4.1.13) above to (4.1.20), which corresponds to (4.1.17),

$$\text{ND}=\frac{NT\times 2^{-(96-(k+16))}}{1+\varepsilon t\times 2^{-(96-(k+16))}} \tag{4.1.21}$$

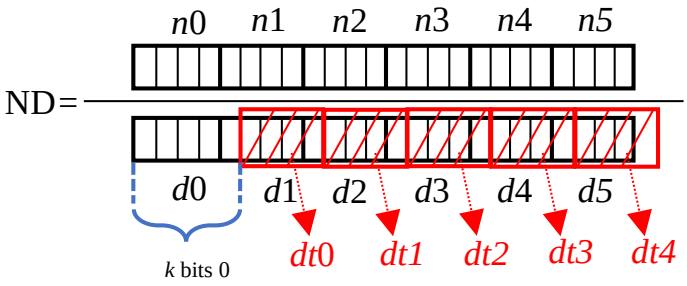


Fig.4.1.9 Register allocation for multiple precision division when denominator k bits is zero

where, $NT = \frac{\text{num}}{dt0}$ and $\varepsilon t = \frac{\text{dt}}{dt0}$.

4.1.4 Numerical Examples

As mentioned in the last section, DS control signals are generated in the MASTER SCHEDULER module. Fig.4.1.10 shows an example of the DS signals, it takes 8 clock cycles for one iteration of BiCG-Stab in (4.1.8) to complete, and it costs a total of 272

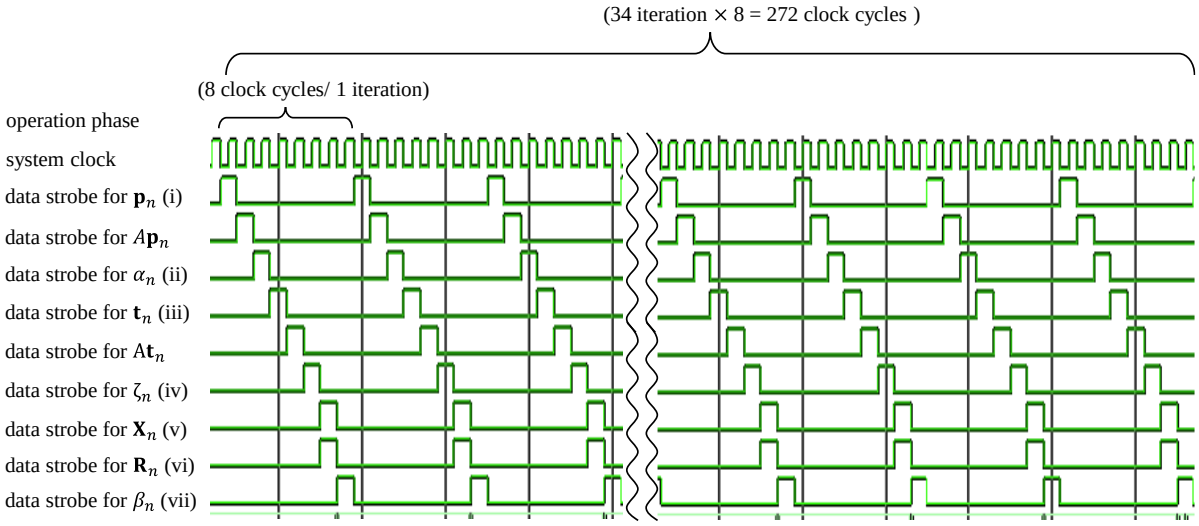


Fig.4.1.10 Data strobe control signals generated at MASTER SCHEDULER module

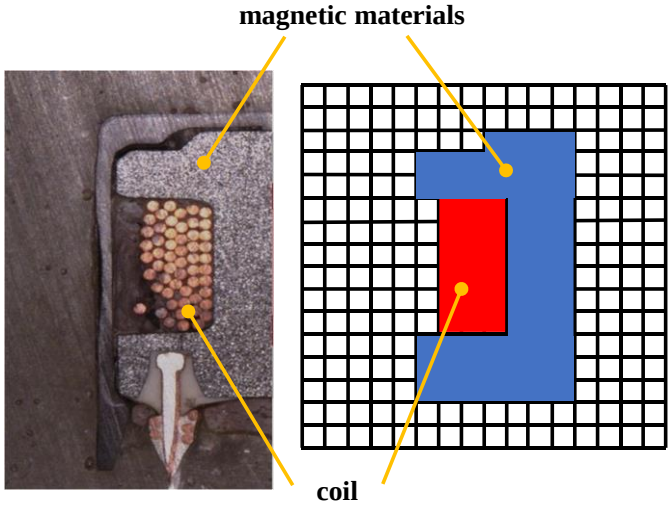
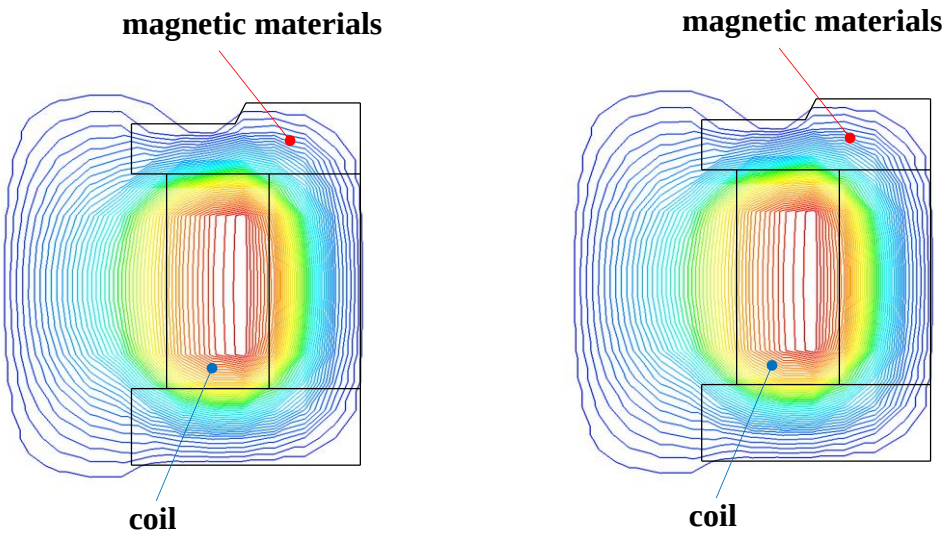


Fig.4.1.11 The 2-D numerical model for inductor



(a) C language (b) VHDL simulation

Fig.4.1.12 Simulation result of magnetic fields force

clock cycles until to the residual $\mathbf{r}_n$ converges.

In addition, the FIT dataflow machine intended for 2-D magneto-static field simulation is designed by using hardware description language, namely VHDL. Fig.4.1.11 shows a 2-D numerical model for the inductor, and this 2-D numerical model is discretized in the grids space with 16×16 grids, the outer boundary was set to be 0. Then the magnetic vector potential distribution of the 2-D numerical model is simulated by C language and VHDL simulation, revealing that the results of C language for simulation and VHDL simulation have a good appointment. That means, the designed logic circuit of BiCG-Stab matrix solver for FIT scheme can operate correctly. Furthermore, if we implement this designed dataflow machine on 50-MHz FPGA, it will cost about $6\mu s$ to obtain the result. then the computation time for C language based on a standard PC with Core i5 CPU will cost about 40ms. That means, FIT dataflow machine has more than 6000 times performance.

4.2 Dedicated Computer for 3-D Electro-static Field Simulation

In the last section, the design of dataflow machine for 2-D magneto-static field is discussed. In this section, the design of FIT dataflow machine for 3-D electro-static field simulation is explained. As mentioned in Chapter 3.1.1, the Poisson's equation is the basis of electro-static field. Then, the calculation of 3-D electro-static field is simulated based on FIT (Chapter 3.2.2). After a discretization of the whole grids space of 3-D electro-static field, the distribution of electric potential in the form of matrix equation is obtained. To solve this matrix equation, a same iterative method as 2-D magneto-static field case is considered, that is, BiCG-Stab scheme. In addition, a "sliced 3-D architecture" is considered for 3-D field simulation performed in FDTD dataflow machine (Chapter 2.3.2). Also, the implementation of the BiCG-Stab matrix solver scheme on the hardware circuit (Chapter 3.4) is discussed. Finally, the FIT dataflow machine for 3-D electro-static field simulation is designed by using the hardware description language VHDL (Chapter 3.4.2).

4.2.1 FIT Scheme for 3-D Electro-static Field

The behavior of electro-static field can be expressed as the following Poisson's Equation for the scalar potential $\emptyset$,

$$\nabla \cdot (\varepsilon \nabla \emptyset) = -\rho \tag{4.2.1}$$

$$\mathbf{E} = -\nabla \emptyset \tag{4.2.2}$$

where, ε represents the permittivity, $\emptyset$ represents the scalar potential, ρ represents the charge density, and **E** represents the electric field intensity. Considering that the case of

3-D electro-static field is discussed, the following equation holds,

$$\mathbf{E} = \begin{pmatrix} \mathbf{E}_x \\ \mathbf{E}_y \\ \mathbf{E}_z \end{pmatrix} = \begin{pmatrix} -\dfrac{\partial \emptyset}{\partial x} \\ -\dfrac{\partial \emptyset}{\partial y} \\ -\dfrac{\partial \emptyset}{\partial z} \end{pmatrix} \tag{4.2.3}$$

Then, the integral form of the Equation (4.2.1) is presented as follows,

$$\oiint_S (\varepsilon \nabla \emptyset) \cdot d\mathbf{S} = -\oiiint_v \rho dv \tag{4.2.4}$$

After the discretization in a grid space in FIT scheme for the above Equation (4.2.4) as shown in Fig.4.2.1. The following equation can be obtained,

$$\Delta S(D_{x2} + D_{y2} + D_{z2} - D_{x1} - D_{y1} - D_{z1}) = -\rho_{i,j,k}\Delta v \tag{4.2.5}$$

where D_x, D_y and D_z are referred to as the electric displacement. Then, the above Equation (4.2.5) can be rewritten as follows,

$$\begin{aligned} c^0_{i,j,k}\emptyset_{i,j,k} - c^1_{i,j,k}\emptyset_{i+1,j,k} - c^2_{i,j,k}\emptyset_{i,j+1,k} - c^3_{i,j,k}\emptyset_{i,j,k+1} - c^4_{i,j,k}\emptyset_{i-1,j,k} \\ - c^5_{i,j,k}\emptyset_{i,j-1,k} - c^6_{i,j,k}\emptyset_{i,j,k-1} = -\rho_{i,j,k}\Delta v \end{aligned} \tag{4.2.6}$$

where,

$$\begin{aligned} c^1_{i,j,k} &= \frac{\varepsilon_{i,j,k} + \varepsilon_{i,j,k-1} + \varepsilon_{i,j-1,k-1} + \varepsilon_{i,j-1,k}}{4}, \\ c^2_{i,j,k} &= \frac{\varepsilon_{i,j,k} + \varepsilon_{i,j,k-1} + \varepsilon_{i-1,j,k-1} + \varepsilon_{i-1,j,k}}{4}, \\ c^3_{i,j,k} &= \frac{\varepsilon_{i,j,k} + \varepsilon_{i,j-1,k} + \varepsilon_{i-1,j-1,k} + \varepsilon_{i-1,j,k}}{4}, \\ c^4_{i,j,k} &= \frac{\varepsilon_{i-1,j,k} + \varepsilon_{i-1,j,k-1} + \varepsilon_{i-1,j-1,k-1} + \varepsilon_{i-1,j-1,k}}{4}, \\ c^5_{i,j,k} &= \frac{\varepsilon_{i,j-1,k} + \varepsilon_{i,j-1,k-1} + \varepsilon_{i-1,j-1,k-1} + \varepsilon_{i-1,j-1,k}}{4}, \\ c^6_{i,j,k} &= \frac{\varepsilon_{i,j,k-1} + \varepsilon_{i,j-1,k-1} + \varepsilon_{i-1,j-1,k-1} + \varepsilon_{i-1,j,k-1}}{4}, \\ c^0_{i,j,k} &= c^1_{i,j,k} + c^2_{i,j,k} + c^3_{i,j,k} + c^4_{i,j,k} + c^5_{i,j,k} + c^6_{i,j,k} \end{aligned} \tag{4.2.7}$$

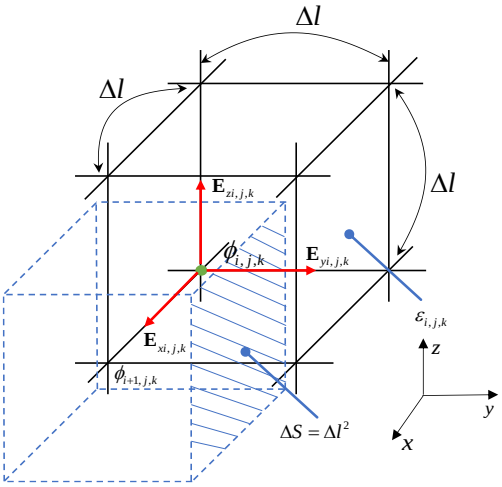


Fig. 4.2.1 FIT discretization in 3-D grid space

$$\begin{bmatrix} c_0 & c_0 & c_0 & & & 0 & & \\ & \ddots & \ddots & \ddots & & & & \\ & & c_4 & c_5 & c_6 & c_0 & c_1 & c_2 & c_3 \\ & & & \ddots & \ddots & \ddots & & & \\ 0 & & & & & & c_0 & c_0 & c_0 \end{bmatrix} \begin{bmatrix} \phi_{0,0,0} \\ \vdots \\ \phi_{i,j,k} \\ \vdots \\ \phi_{ni,nj,nk} \end{bmatrix} = \begin{bmatrix} -\rho_{0,0,0} \\ \vdots \\ -\rho_{i,j,k} \\ \vdots \\ -\rho_{ni,nj,nk} \end{bmatrix}$$

Fig. 4.2.2 FIT scheme matrix equation

To construct the matrix equation based on the discretization Equation (4.2.6) for all grid points as shown in Fig.4.2.2. The final FIT matrix equation is obtained. And then the information about distribution of the magnetic field can be obtained by solving the matrix equation in Fig.4.2.2.

4.2.2 Hardware Circuit of BiCG-Stab Matrix Solver

In this case for 3-D electro-static field simulation, the same iterative method as in the case of 2-D magneto-static field simulation is adopted. The unit grid circuit of BiCG-Stab

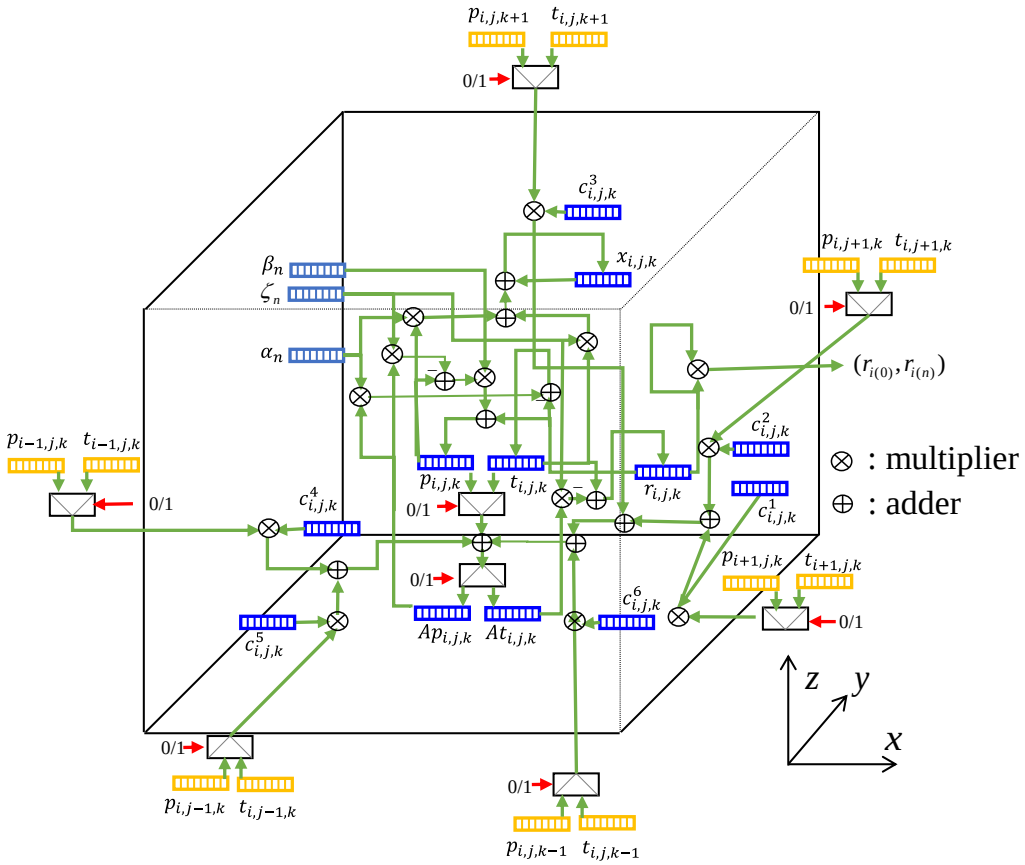


Fig. 4.2.3 Unit grid circuit of BiCG-Stab matrix solver for 3-D electrostatic field

matrix solver for 3-D electro-static field is shown in the Fig.4.2.3. Then if many unit grid circuits are interconnected all over the 3-D grids space, such as the allocation of 2-D magneto-static field, this allocation can indeed achieve an extremely high-performance computation. However, such the 3-D grid circuits will result in a large hardware size, and it is impossible to be implemented in a single LSI (Large-Scale Integrated Circuits).

To avoid a very large hardware size, a "3-D sliced structure" is considered and implemented in a 3-D FDTD dataflow machine for microwave simulation (Chapter 2.3.2). The basic "3-D sliced structure" is shown in Fig. 4.2.4. There are 3 gird circuits arranged vertically, and they are called "arithmetic 3 grid circuit". The upper and lower grid circuits only have registers to store the unknown values ($\mathbf{p}_n, \mathbf{t}_n, \mathbf{x}_n, \mathbf{r}_n, A\mathbf{p}_n, A\mathbf{t}_n$) of iterative

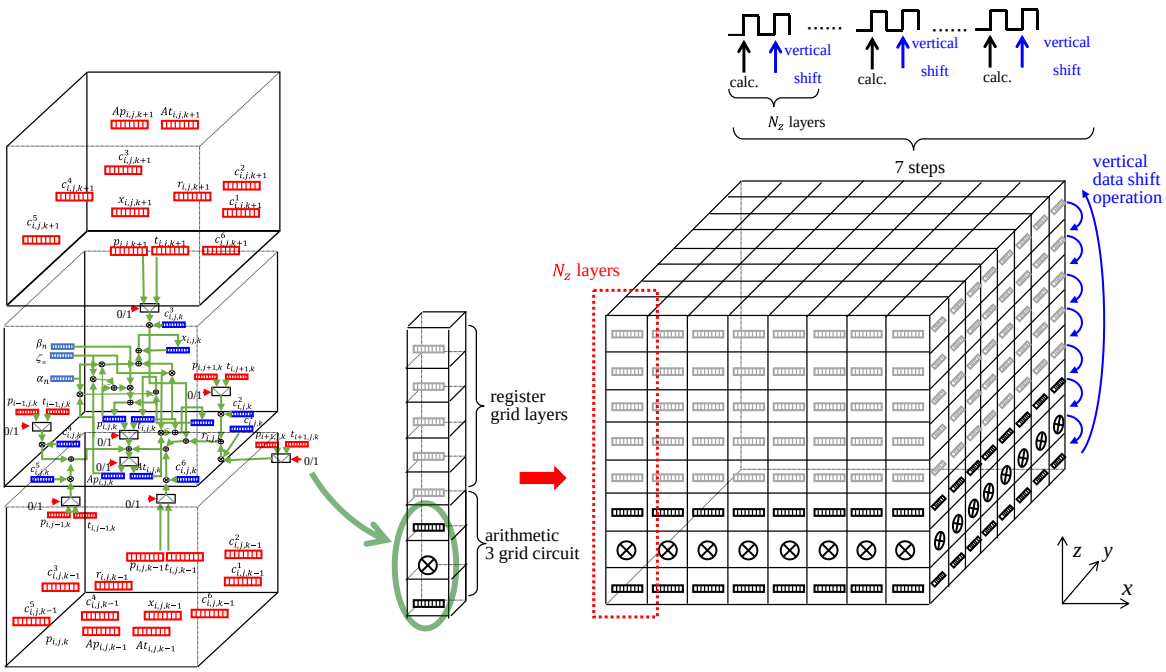


Fig. 4.2.4 Sliced 3-D dataflow architecture

process in BiCG-Stab scheme (4.1.8) and the coefficients value $(c^0_{i,j,k}, c^1_{i,j,k}, c^2_{i,j,k}, c^3_{i,j,k}, c^4_{i,j,k}, c^5_{i,j,k}, c^6_{i,j,k})$ of (4.2.7). The middle grid circuit execute the iterative process of BiCG-Stab scheme in (4.1.8). This "arithmetic 3 grid circuit" is located at the bottom, with the register grid layer added on the top to form a "Sliced 3-D dataflow architecture" as shown in Fig.4.2.4. This "Slice 3-D dataflow architecture" expands horizontally to all 3-D grids space as shown in Fig. 4.2.4. The calculation and vertical shift are alternated to execute the iterative process of BiCG-Stab throughout the 3-D grids space. For example, for N_z layers, after the middle grid circuit of "arithmetic 3 grid circuit" executes the iterative process (i) in (4.1.8), the updated value shift down to the lower grid, and the registers of upper grid shift down to the middle grid circuit to execute (i) in (4.1.8) until the value of every layer can be updated in all over 3-D grids space.

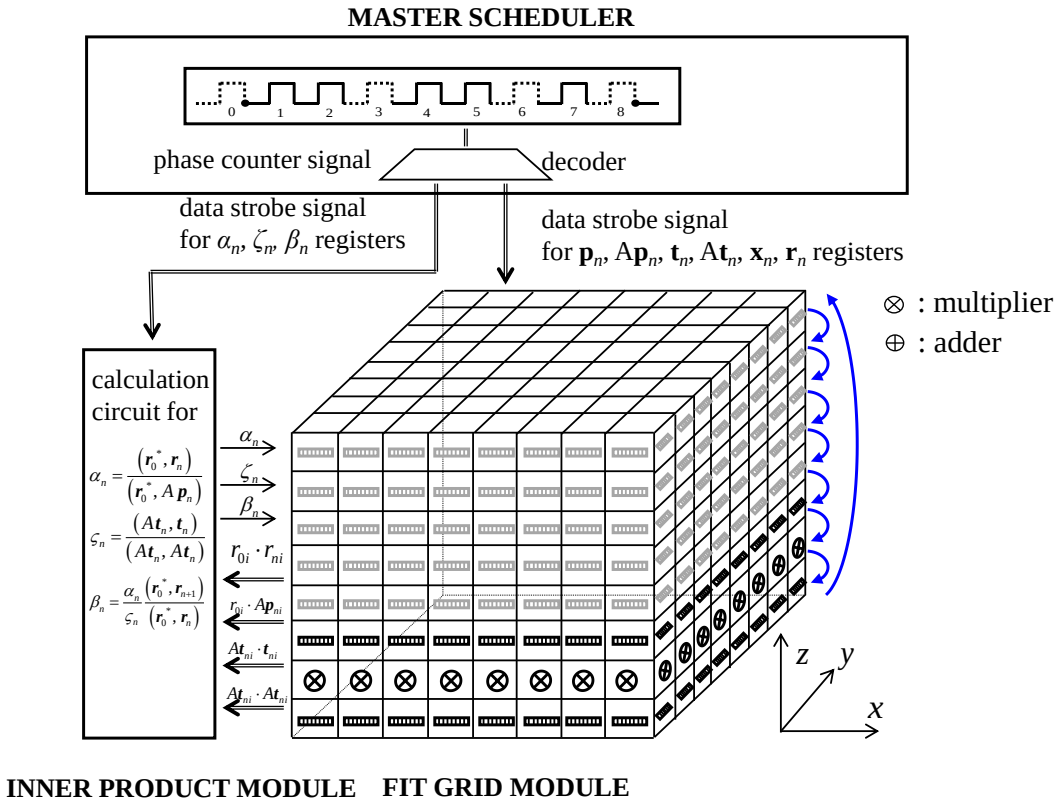


Fig. 4.2.5 The whole configuration of 3-D FIT dataflow machine

4.2.3 Numerical Examples

The whole configuration of FIT dataflow machine for 3-D electrostatic field simulation as shown in Fig.4.2.5, which consists of 3 parts, FIT GRID module, INNER PRODUCT module and MASTER SCHEDULER. These three parts have same function as the whole

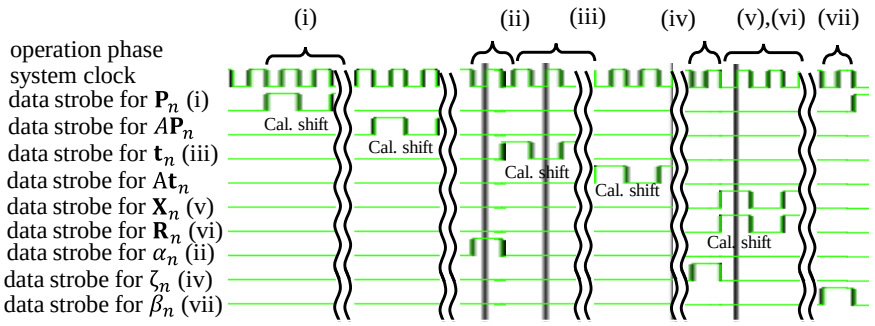


Fig. 4.2.6 VHDL simulation for MASTER SCHEDULER

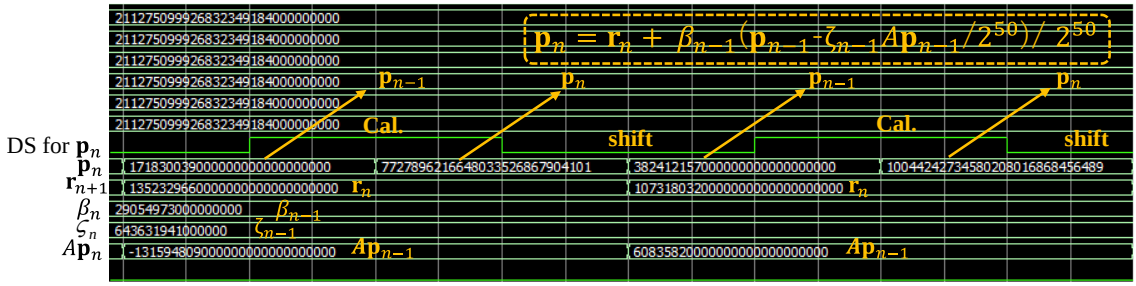


Fig. 4.2.7 VHDL simulation result for one row of 3-D grids space

configuration of 2-D magneto-static field simulation, FIT GRID module executes the iterative process of BiCG-Stab scheme, INNER PRODUCT module executes the inner product calculations of α_n, ζ_n, β_n and then sends back to the FIT GRID module, and the MASTER SCHEDULER module generates DS signals to control the execution of FIT GRID module and INNER PRODUCT module. The VHDL simulation of MASTER SCHEDULER for one iteration process of BiCG-Stab scheme (4.1.8) is shown in Fig.4.2.6. In one iteration of BiCG-Stab in all the 3-D grids space, it takes $(2N_z \times 5 + 3N_z + 3)$ clock cycles.

The VHDL simulation of one row for the $\mathbf{p}_n$ of (i) in (4.1.8) is taken as an example as shown in Fig.4.2.7. When the DS for $\mathbf{p}_n$ at a high level, the middle layer of "arithmetic 3 grid circuit" executes the computation for $\mathbf{p}_n$ of (i) in (4.1.8). When the DS for $\mathbf{p}_n$ at a low level, the unknown values ($\mathbf{p}_n$, $\mathbf{t}_n$, $\mathbf{x}_n$, $\mathbf{r}_n$, $A\mathbf{p}_n$, $A\mathbf{t}_n$) of the iterative process in (4.1.8) and the coefficients ($c^0_{i,j,k}$, $c^1_{i,j,k}$, $c^2_{i,j,k}$, $c^3_{i,j,k}$, $c^4_{i,j,k}$, $c^5_{i,j,k}$, $c^6_{i,j,k}$) of (4.2.7) are shift down to the lower layer. Then, it can be confirmed that the computation for $\mathbf{p}_n$ is executed correctly according to the (i) in (4.1.8). That is to say, the designed "Sliced 3-D dataflow architecture" is carried out as normal.

4.3 Dedicated Computer for Eddy Current Field Simulation

In this section, the conceptual design of dataflow machine for 2-D eddy current field simulation and its specific circuit design including matrix solver is explained. As mentioned in Chapter 3.1.3, the basic equation for the behavior of eddy current fields were introduced. Then, the calculation of 2-D eddy current fields is simulated based on FIT (Chapter 3.2.2). After a discretization of the whole grids space of 2-D eddy current fields, we obtain the distribution of magnetic potential in the form of matrix equation. To solve this matrix equation, it is considered to use the same iterative method as 2-D magneto-static fields, that is BiCG-Stab (Chapter 3.3.3). Also, the implementation of the BiCG-Stab matrix solver scheme on the hardware circuit (Chapter 3.4) is discussed. Finally, the FIT dataflow machine for 2-D eddy current field simulation is designed by using the hardware description language VHDL (Chapter 3.4.2).

4.3.1 FIT Scheme for 2-D Eddy Current Field

The behavior of eddy current field can be described by using the following vector Poisson's Equation for the scalar potential **A**,

$$\nabla \times \left(\frac{1}{\mu}\nabla \times \mathbf{A}\right) = -\sigma\frac{\partial A_z}{\partial t} + J_{0z} \tag{4.3.1}$$

where, μ represents the permeability, σ represents the conductivity of conductor, J represents current density. After discretization in a grid space in FIT scheme for the above Equation (4.3.1) as shown in Fig.4.2.1. The following equation can be obtained,

$$\left(\frac{\sigma}{\Delta t}+\frac{C^1+C^2+C^3+C^4}{2}\right)A_{zi,j}^{n+1}$$
$$-\frac{C^1}{2}A_{zi,j-1}^{n+1}-\frac{C^2}{2}A_{zi+1,j}^{n+1}-\frac{C^3}{2}A_{zi,j+1}^{n+1}-\frac{C^4}{2}A_{zi-1,j}^{n+1}$$
$$=\left(\frac{\sigma}{\Delta t}-\frac{C^1+C^2+C^3+C^4}{2}\right)A_{zi,j}^{n} \tag{4.3.2}$$
$$-\frac{C^1}{2}A_{zi,j-1}^{n}-\frac{C^2}{2}A_{zi+1,j}^{n}-\frac{C^3}{2}A_{zi,j+1}^{n}-\frac{C^4}{2}A_{zi-1,j}^{n}+\mathbf{J}$$

where,

$$C^1=\frac{1}{2\mu_{i-1,j-1}}+\frac{1}{2\mu_{i,j-1}},C^1=\frac{1}{2\mu_{i,j-1}}+\frac{1}{2\mu_{i,j}},$$
$$C^3=\frac{1}{2\mu_{i,j-1}}+\frac{1}{2\mu_{i-1,j}},C^4=\frac{1}{2\mu_{i-1,j}}+\frac{1}{2\mu_{i-1,j-1}} \tag{4.3.3}$$

where $A_{zi,j}^{n}$ represents the n-th time step value of z-component of the vector potential at i-th, j-th grid for x and y directions. According to the above Equation (4.3.2), the (n+1)-th time step value of $A_{zi,j}^{n+1}$ at i-th, j-th grid can be calculated when the previous time step value of $A_{zi,j}^{n}$ is obtained. Then, to construct the discretized equation of (4.3.2) for all grids space and time axis, the matrix equation is obtained, as shown in the Fig.4.3.1. In the FIT scheme for eddy current fields, the behavior of eddy current fields is described by Equation (4.3.1). Then, in addition to being discretized in space, it is also discretized in time domain. After the discretization, and to construct the discretized equation for all grids space and time axis in matrix form, the matrix equation can be obtained (Fig.4.3.1).

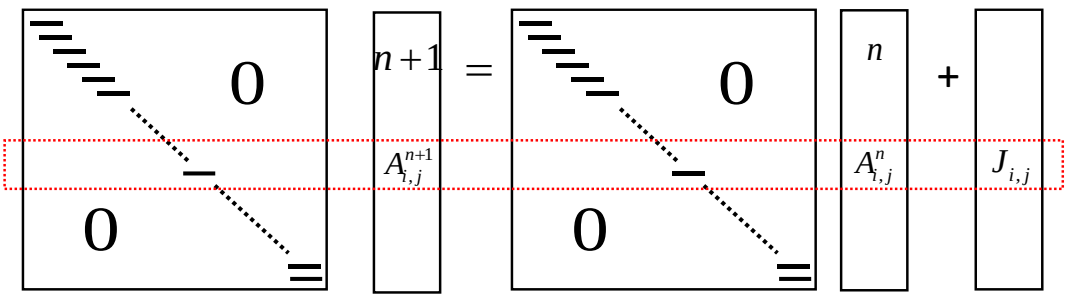


Fig. 4.3.1 FIT scheme matrix equation

And then, the right hand side of the matrix equation (Fig.4.3.1), except the given value current J, there is also the n-th time step value, after addition of these two components, the result of $(n+1)$-th time step can be calculated. Then, the matrix form similar to the matrix of magneto-static field case can be found (Fig.4.1.2).

4.3.2 Hardware Circuit of BiCG-Stab Matrix Solver

To solve the large sparse matrix equation, the iterative method, BiCG-Stab scheme is considered. In addition, the detailed arithmetic circuits of the BiCG-Stab matrix solver based on dataflow architecture are designed. Because the discretized matrix equation of 2-D eddy current fields is the same as the discretized matrix equation of 2-D magneto-static fields, the circuit connection of iteration process for eddy current fields is also the same as in the case of magneto-static fields, as shown in Fig.4.3.2. It has been known that, the circuit connection of iteration process of BiCG-Stab scheme for 2-D eddy current

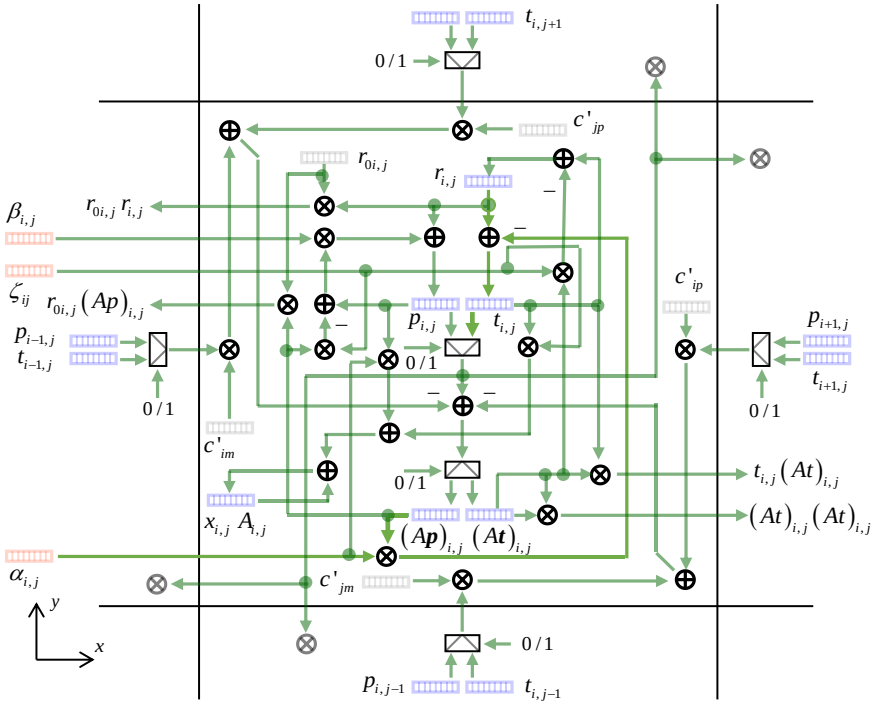


Fig. 4.3.2 Circuit connection for the iterative process of BiCG-Stab scheme

fields is the same as in the case of 2-D magneto-static fields. Then for the entire arithmetic circuit in the unit grid is discussed in detail. As shown in Fig.4.3.3, on the right hands side of the matrix equation, there are two components, not only the given value **J**, but also the result of previous time step $\mathbf{A}^n$. After addition of these two components, the vector $\boldsymbol{v}$ is

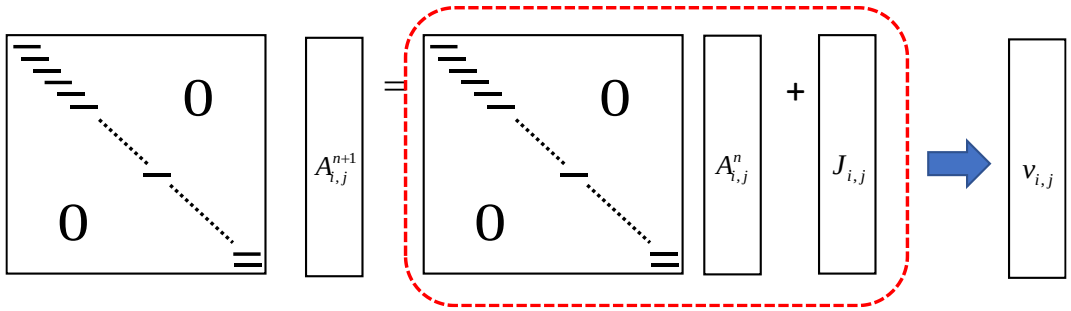


Fig. 4.3.3 FIT scheme matrix equation

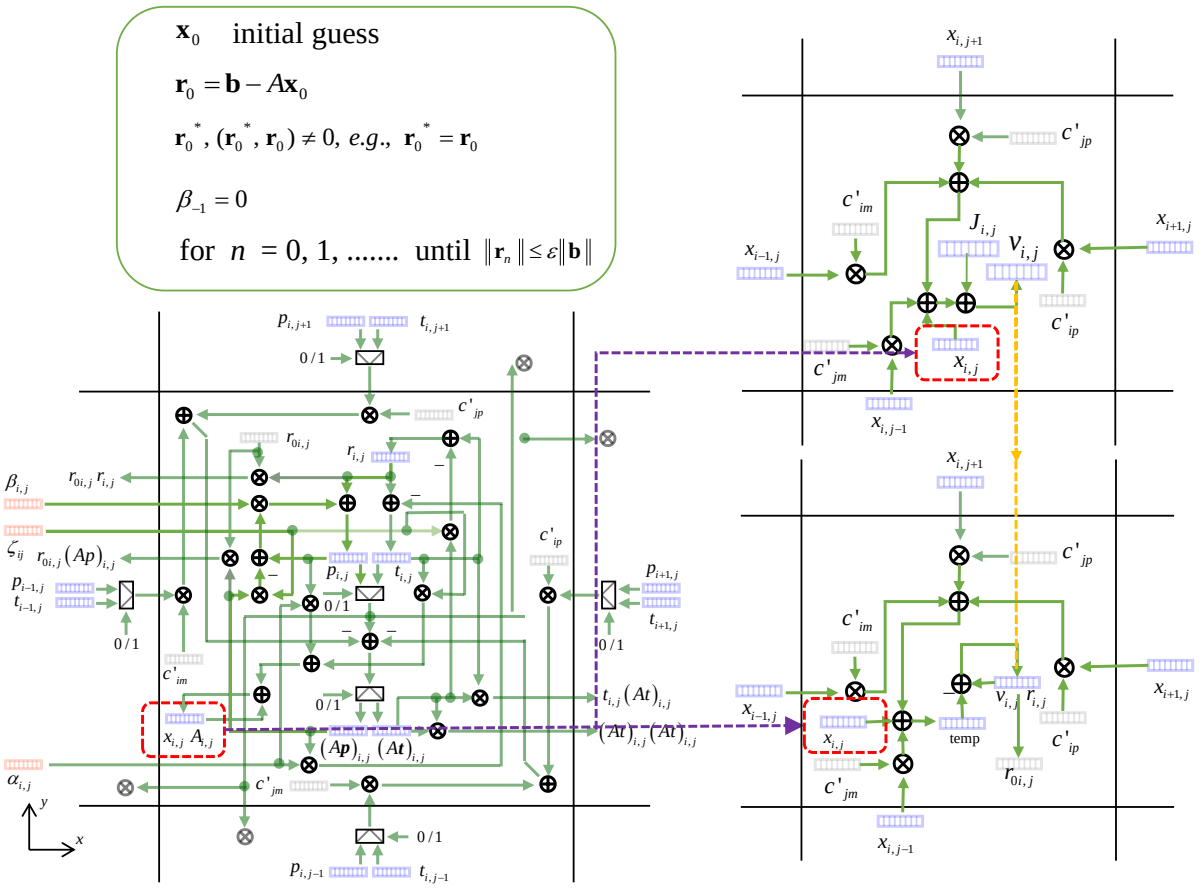


Fig. 4.3.4 Circuit connection for the initial value setting of BiCG-Stab scheme

obtained. Then, the dataflow machine executes the iteration process similar to the case of 2-D magneto-static fields. In addition, the right-hand side of the matrix equation is reconstructed, and the initial value of BiCG-Stab scheme is also need to be reset. Therefore, it is considered to design two more circuits for calculating the reconstruction on the right-hand side, and setting the initial value, as shown in Fig.4.3.4. After the last iteration process is completed (the bottom left of Fig.4.3.4), the last time step value $\mathbf{A}^n$ is updated to calculate the right-hand side of matrix equation (the bottom right of Fig.4.3.4). Then, the initial value is reset (the upper right of Fig.4.3.4).

4.2.3 Numerical Examples

The whole configuration of FIT dataflow machine for 2-D eddy current field simulation

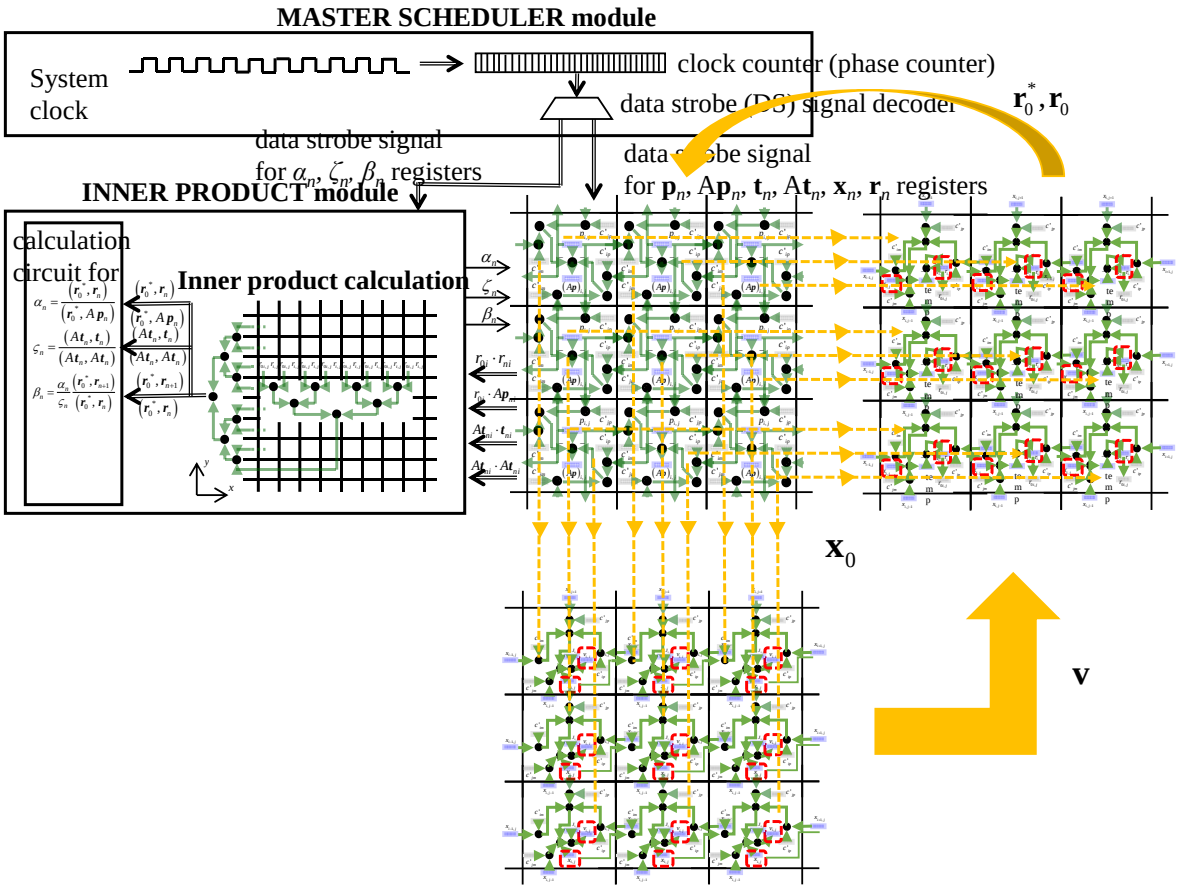


Fig. 4.3.5 The whole configuration of 2-D eddy current fields dataflow machine

as shown in Fig.4.3.5, which consists of three parts, the FIT GRID module including the circuit of right-hand side of matrix equation reconstructed and the circuit of initial value reset, the INNER PRODUCT module and the MASTER SCHEDULER. These three parts have the same function as the whole configuration of 2-D magneto-static field simulation, FIT GRID module executes the iterative process of BiCG-Stab scheme, the INNER PRODUCT module executes the inner product calculations of α_n, ζ_n, β_n and sends back to the FIT GRID module, the MASTER SCHEDULER module generates DS signals to control the execution of FIT GRID module and INNER PRODUCT module. The

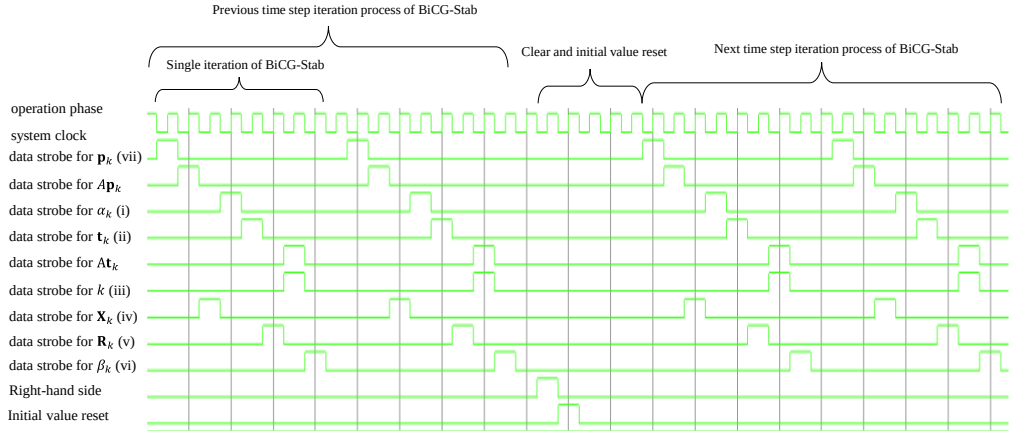


Fig. 4.3.6 VHDL simulation for MASTER SCHEDULER

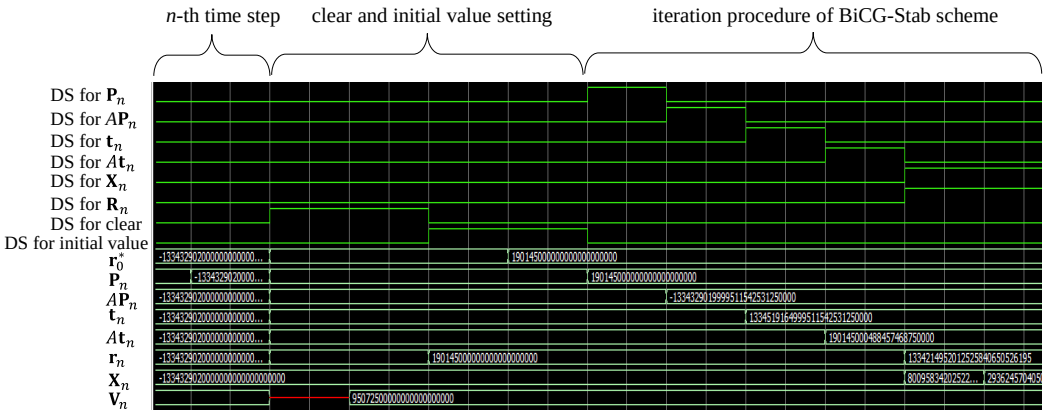


Fig. 4.3.7 VHDL simulation result for clear and initial value setting and one iteration

VHDL simulation of MASTER SCHEDULER for one iteration process of BiCG-Stab scheme (4.1.7) is shown in Fig.4.3.6. In one iteration of BiCG-Stab in all the 2-D grids space, it takes 12 clock cycles, and the process of reconstructing and resetting the initial value takes 2 clock cycles.

During VHDL simulation, the clear and initial setting process is conducted to confirm that the circuits of reconstruction and initial value reset are performed correctly. In addition, VHDL simulation of one iteration process of BiCG-Stab scheme is performed to confirm the circuits of iteration process of BiCG-Stab shceme in unit grid can be carried out correctly. Also, these two parts between clear and initial value setting with iteration process of BiCG-Stab scheme can be connected smoothly and correctly to perform the arithmetic circuits.

Chapter 5

Summary

In this research, we mainly focus on the development of FIT dataflow machine based on the BiCG-Stab scheme, and the design of overview architecture of dedicated computer. Then, VHDL simulation is performed to confirm the correctness of designed logic circuit. In summary, we have accomplished the following things:

(1) As one part of the development of portable, low-power HPC technology for electromagnetic field simulation through the dataflow architecture dedicated computer method, we investigate an arbitrary multiple-precision integer division circuit for high-speed matrix calculations in order to further expand the application fields. An arbitrary multiple-precision integer division can now be executed in one clock.

(2) We develop a FIT dataflow machine for the simulation of 2-D magneto-static fields. The detailed dataflow circuits of the BiCG-Stab matrix solver for the FIT matrix equation are designed, and the whole configuration of the entire system of the dedicated computer is designed, including the FIT GRID module, the INNER PRODUCT module, and the MASTER SCHEDULER module. In addition, the dataflow machine of the FIT dedicated computer for 2-D magneto-static field simulation is designed by using the hardware description language, VHDL. The result of VHDL simulation is compared with that of C software simulation to confirm the correctness and validity of the designed dataflow machine.

(3) We also discuss the FIT dataflow machine for the simulation of 3-D electro-static

fields. Also, the specific dataflow circuits of the BiCG-Stab matrix solver for the FIT matrix equation are designed, and the whole configuration of the entire system of the dedicated computer is designed, including the FIT GRID module, INNER PRODUCT module, and the MASTER SCHEDULER module. In addition, to reduce the hardware size of dataflow machine to achieve a higher performance machine, it is considered to use the "Sliced 3-D dataflow architecture", and the single BiCG-Stab iteration process for all the 3-D grids space takes $(2N_z \times 5 + 3N_z + 3)$ clock cycles. Then the logic circuit of the vertical grid circuit of 3-D FIT dataflow machine is designed through VHDL to confirm that the designed logic circuits for the BiCG-Stab scheme are implemented correctly.

(4) We discussed the 2-D FIT dataflow machine for the simulation of eddy current fields. And the detailed logic circuits based on the BiCG-Stab scheme are designed. In addition, the unit grid of BiCG-Stab matrix solver includes three parts, the iteration process circuit, the circuit for reconstruction of the right-hand side of FIT matrix equation, and the initial value setting circuit. It is confirmed by the VHDL simulation that the circuits designed via VHDL for the BiCG-Stab matrix solver in unit grid are carried out correctly. We will soon proceed to the VHDL simulation for all grids, and then evaluate performance of the 2-D FIT dataflow machine for eddy current fields simulation.

References

[1] T. Weiland, "Time domain electromagnetic field computation with finite difference methods", International Journal of Numerical Modelling, 9: 295-319, 1996.

[2] T. Weiland, "A discretization method for the solution of Maxwell's equations for six-component fields", Electronics and Communication (AE), 31:116, 1977.

[3] M. Clemens and T. Weiland, Discrete Electromagnetism with the Finite Integration Technique, Progress In Electromagnetics Research, PIER 32, 65-87, 2001

[4] M. Walter, I. Munteanu, "FIT for EMC," Proc. of the 17th International Zurich Symposium on Electromagnetic Compatibility, , 15-17, 2006.

[5] K. Krishnakumar, “Micro-genetic algorithm for stationary and non-stationary function optimization,” in Proc. SPIE 1196, Intell. Control Adapt. Syst., 289–296,1989

[6] M. P. Bendsoe, “Optimal shape design as a material distribution problem,” Struct. Optim., 1:193–202, 1989.

[7] O. Sigmund and J. Petersson, “Numerical instabilities in topology optimization, a survey on procedures dealing with checkerboards, mesh-dependencies and local minima,” Struct. Optim., 16:68–75, 1998

[8] A. Canova, M. Repetto. Magnetic design optimization and objective function approximation[J]. IEEE Trans. Magn.,39(5):2154-2162, 2003.

[9] F. Campelo, F.G. Guimaraes, H. Igarashi, and J.A. Ramirez, “A clonal selection algorithm for optimization in electromagnetics,” IEEE Trans. Magn., 41(5):1736–1739, 2005.

[10] K. Watanabe, F. Campelo, Y. Iijima, et al. Optimization of Inductors Using Evolutionary Algorithms and Its Experimental Validation[J]. IEEE Trans. Magn., 46(8):3393-3396, 2010.

[11] S. Shimokawa, H. Oshima, K. Shimizu, et al. Fast 3-D Optimization of Magnetic Cores for Loss and Volume Reduction[J]. IEEE Trans. Magn., 54(11):1-4, 2018.

[12] K. Itoh, H. Nakajima, H. Matsuda, et al. Development of Small Dielectric Lens for Slot Antenna Using Topology Optimization with Normalized Gaussian Network[J]. IEICE Transactions on Electronics, E101.C(10):784-790, 2018.

[13] H. Sasaki, H. Igarashi. Topology Optimization Accelerated by Deep Learning[J]. IEEE Trans. Magn., 55(6):1-5, 2019.

[14] S. Doi, H. Sasaki, H. Igarashi. Multi-Objective Topology Optimization of Rotating Machines Using Deep Learning[J]. IEEE Trans. Magn., 55(6):1-5,2019.

[15] Y. Otomo, H. Igarashi, Y. Hidaka, et al. 3-D Topology Optimization of Claw-Pole Alternator Using Gaussian-Basis Function With Global and Local Searches[J]. IEEE Trans. Magn., 56(1):1-4, 2019.

[16] F. Ye, H. Igarashi. Topology Optimization of Metamaterial Using Gaussian-Basis Functions[J]. Journal of Advanced Simulation in Science and Engineering, 6(1):149-156, 2019.

[17] Y. Otomo, H. Igarashi. 3-D topology optimization of magnetic cores for wireless power transfer with double-sided winding coils[J]. International Journal of Applied Electromagnetics and Mechanics, 60:S115-S123, 2019.

[18] H. Sakamoto, K. Okamoto, H. Igarashi. Fast Analysis of Rotating Machine Using Simplified Model-Order Reduction Based on POD[J]. IEEE Trans. Magn., 56(2):1-4, 2020.

[19] J. Asanuma, S. Doi, H. Igarashi. Transfer Learning Through Deep Learning: Application to Topology Optimization of Electric Motor[J]. IEEE Trans. Magn., 56(3):1-4, 2020.

[20] S. Hiruma, M. Ohtani, S. Soma, et al. Novel Hybridization of Parameter and Topology Optimizations: Application to Permanent Magnet Motor[J]. IEEE Trans. Magn., 57(7):1-4, 2021.

[21] Mark, J. R. "A dedicated VLSI architecture for finite-difference time domain calculations." Conf. Proc. of 8th Annual Review of Progress in Applied Comput. Electromagnetics, 1992. 1992.

[22] P. Placidi, L. Verducci, G. Matrella, L. Roselli, and P. Ciampolini, A Custom VLSI Architecture for the Solution of FDTD Equations, IEICE Trans.Electron., E85-C(3) :572-577, 2002.

[23] R.N. Schneider, M.M. Okoniewski,,L.E. Turner, “Finite -difference time -domain method in custom hardware”, Microwave and Wireless Components Letters, IEEE, 12(1) :488 -490,2002.

[24] Schneider, Ryan N., Laurence E. Turner, and Michal M. Okoniewski. "Application of FPGA technology to accelerate the finite-difference time-domain (FDTD) method." Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays. 2002.

[25] Verducci, L., et al. "A feasibility study about a custom hardware implementation of the FDTD algorithm." Proc. of The 27th General Assembly of the URSI (2002).

[26] H. Kawaguchi, K. Takahara, D. Yamauchi, “Design Study of Ultra -high Speed Microwave Simulator Engine”, IEEE Trans. Magn.,38(2): 689 -692,2002.

[27] J.P. Durbano, F.E. Ortiz, J.R. Humphrey, M.S. Mirotznik , D.W. Prather, Hardware Implementation of a Three-Dimensional Finite -Difference Time -Domain Algorithm, IEEE Antennas and Wireless Propagation Letters,2:54 -57,2003.

[28] S. Matsuoka, K. Ohmi and H. Kawaguchi, Study of a Microwave Simulation Dedicated Computer, FDTD/FIT Data Flow Machine, IEICE Trans. Electron., E86 -C(11)2199 -2206,2003.

[29] Chen, Wang, et al. "An FPGA implementation of the two-dimensional finite-difference time-domain (FDTD) algorithm." Proceedings of the 2004 ACM/SIGDA 12th international symposium on Field programmable gate arrays. 2004.

[30] Durbano, James P., and Fernando E. Ortiz. "FPGA-based acceleration of the 3D finite-difference time-domain method." 12th Annual IEEE symposium on field-programmable custom computing machines. IEEE, 2004.

[31] Schneider, Rvan N, Michal M. Okoniewski, and Laurence E. Turner. "A software-coupled 2D FDTD hardware accelerator [electromagnetic simulation]." IEEE Antennas and Propagation Society Symposium, 2004. Vol. 2. IEEE, 2004.

[32] Suzuki, Hidetoshi, et al. "FPGA Implementation of FDTD Algorithm." 2005 Asia-Pacific Microwave Conference Proceedings. Vol. 5. IEEE, 2005.

[33] Curt, Petersen F., et al. "Enhanced functionality for hardware-based FDTD accelerators." The Applied Computational Electromagnetics Society Journal (ACES) (2007): 29-46.

[34] Kawaguchi, Hideki, et al. "Improved architecture of FDTD/FIT dedicated computer for higher performance computation." IEEE transactions on magnetics 44.6 (2008): 1226-1229.

[35] H. Kawaguchi, Y. Fujita, Y. Fujishima, S. Matsuoka, “Improved, Architecture of FDTD/FIT Dedicated Computer for Higher Performance, Com putation”, IEEE Trans. Magn.,44(6):1226-1229,2008.

[36] Y. Fujita, H. Kawaguchi, “Full Custom PCB Implementation of FDTD/FIT Dedicated Computer”, IEEE Trans. Magn.,45(3): 1100 -1103,2009.

[37] K. Sano, Y. Hatsuda, L. Wang, and S. Yamamoto, "Performance Evaluation of Finite-Difference Time-Domain (FDTD) Computation Accelerated by FPGA-based Custom Computing Machine", Interdisciplinary Information Sciences,15(1):67-78,2009.

[38] Sypek, Piotr, Adam Dziekonski, and Michal Mrozowski. "How to render FDTD computations more effective using a graphics accelerator." IEEE Transactions on Magnetics 45.3 (2009): 1324-1327.

[39] 遠藤 翔, 園田 潤, 佐藤 源之. "FDTD 法の FPGA 実装における時空間パイプラインによる高速化." 電子情報通信学会論文誌 B 92.1 (2009): 243-249.

[40] Sano, Kentaro, et al. "Performance evaluation of finite-difference time-domain (FDTD) computation accelerated by FPGA-based custom computing machine." Interdisciplinary Information Sciences 15.1 (2009): 67-78.

[41] Fujita, Yuya, and Hideki Kawaguchi. "Full-custom PCB implementation of the FDTD/FIT dedicated computer." IEEE transactions on magnetics 45.3 (2009): 1100-1103.

[42] Y. Fujita and H. Kawaguchi, Development of Improved Memory Architecture FDTD/FIT Dedicated Computer Based on SDRAM for Large Scale Microwave Simulation, International Journal of Applied Electromagnetics and Mechanics, 32(3):145-157,2010.

[43] Fujita, Yuya, and Hideki Kawaguchi. "Development of improved memory architecture FDTD/FIT dedicated computer based on SDRAM for large scale microwave simulation." International Journal of Applied Electromagnetics and Mechanics 32.3 (2010): 145-157.

[44] Y. Fujita and H. Kawaguchi, Development of Improved Memory Architecture FDTD/FIT Dedicated Computer Based on SDRAM for Large Scale Microwave

Simulation, International Journal of Applied Electromagnetics and Mechanics, 32(3):145-157,2010.

[45] Y. Fujita, H. Kawaguchi, “Development of Portable High Performance Computing System by Parallel FDTD Dedicated Computers”, 18th International Conference on the Computation of Electromagnetic Fields (2011, Sydney, Australia).

[46] Fujita, Yuya, and Hideki Kawaguchi. "Development of portable high performance computing system by parallel FDTD dedicated computers." *Proc. 18th Int. Conf. Comput. Electromagn. Fields.* 2011.

[47] H. Kawaguchi and S. Matsuoka, Conceptual Design of 3D FDTD Dedicated Computer with Dataflow Architecture for High Performance Microwave Simulation, IEEE Tran. Magn., 51(3) ,2015.

[48] H. Kawaguchi, Improved Architecture of FDTD Dataflow Machine for Higher Performance Electromagnetic Wave Simulation, IEEE Tran. Magn., 52(3) ,2016.

[49] H. Kawaguchi, S. Matsuoka, Implementation of Microwave Simulation at Dispersive Material in Dataflow Architecture FDTD Dedicated Computer, IEEE Tran. Magn.,54(3),2018.

[50] H. Kawaguchi, Design Study of Domain Decomposition Operation in Dataflow Architecture FDTD/FIT Dedicated Computer, IEICE Trans. Electron., E101-C(1): 20-25,2018.

[51] C.X. Wang, S. Ota, H. Kawaguchi, Conceptual Design of Dataflow Machine for Magnetostatic Field Simulation, Proceedings of the 2021 International Conference

on Electromagnetics in Advanced Applications (ICEAA), (2021, August, Hawaii, USA), ID:694, pp.223.

[52] Sugimoto, D. "GRAPE: A parallel computer dedicated to astrophysical many-body problems." Parallel Computing 25.13-14 (1999): 1663-1676.

[53] Hamada, Tsuyoshi, et al. "A comparative study on ASIC, FPGAs, GPUs and general purpose processors in the O (N^ 2) gravitational N-body simulation." 2009 NASA/ESA Conference on Adaptive Hardware and Systems. IEEE, 2009.

[54] Narumi, Tetsu, Atsushi Kawai, and Takahiro Koishi. "An 8.61 Tflop/s molecular dynamics simulation for NaCl with a special-purpose computer: MDM." Proceedings of the 2001 ACM/IEEE conference on Supercomputing. 2001.

[55] Hamada, Tsuyoshi, et al. "PROGRAPE-1: A programmable, multi-purpose computer for many-body simulations." Publications of the Astronomical Society of Japan 52.5 (2000): 943-954.

[56] Hamada, Tsuyoshi, and Naohito Nakasato. "PGR: a software package for reconfigurable super-computing." International Conference on Field Programmable Logic and Applications, 2005. IEEE, 2005.

[57] Makino, Junichiro, Kei Hiraki, and Mary Inaba. "GRAPE-DR: 2-Pflops massively-parallel computer with 512-core, 512-Gflops processor chips for scientific computing." Proceedings of the 2007 ACM/IEEE conference on Supercomputing. 2007.

[58] Kawaguchi, Hideki, Shin Kubo, and Hiroaki Nakamura. "Orbital angular momentum of vortex fields in corrugated cylindrical waveguide hybrid mode." IEEE Microwave and Wireless Technology Letters 33.2 (2022): 118-121.

Acknowledgements

When I wrote this acknowledgement, WHO declared the end of the COVID-19 emergency. My story began in January 2020 and is coming to an end in the summer of 2023. Time has been slow, so slow that every year when spring arrives I wonder if the COVID-19 will be over next spring. Time flies, and the COVID-19 is almost over, but I have to say goodbye.

Looking back on the three years, I often feel lucky. I guess I really saved the universe in my last life so that I could meet my supervisor in this life, a supervisor who is as warm as a friend and family, Kawaguchi sensei. Words are powerless to express my gratitude, but I still want to express my most sincere gratitude to Kawaguchi sensei here. I am grateful to Kawaguchi sensei for his endless patience, tolerance and careful guidance in my research, and for developing my logical thinking ability both in my research and life. In addition, I am also grateful to Kawaguchi sensei for the care he gave me in my daily life. Although I was studying abroad alone, I never felt lonely but surrounded by warmth.

In addition, I owe my sincere gratitude to the students in our laboratory, Takahashi san, Anda kun, and everyone who helped me a lot with my research and life

Finally, I would like to thank to my family. Thank you to my mom for always supporting me unconditionally in any decision I make and giving me the courage to do any what I want to do. I would like to my sister for acting as a chat “tool” whenever I couldn’t get through to my mom and my boyfriend online and accompany with me walk through the boring journey home. And thank you to my boyfriend for being there every day for his company during the past three years we haven’t seen each other.

List of Publications

Peer-reviewed Papers

[1] Chenxu Wang, Hideki Kawaguchi, Kota Watanabe, "Study of FIT Dedicated Computer with Dataflow Architecture for High Performance 2-D Magneto-static Field Simulation", IEICE Transactions on Electronics, Vol.E106-C, April (2023).

[2] Chenxu Wang, Hideki Kawaguchi, "Design Study of FIT Dataflow Machine for High Performance 3-D Electrostatic Field Simulation", International Journal of Applied Electromagnetics and Mechanics, vol.71, no. S1, pp. S203-S210. May (2023).

[3] WANG CHENXU, 川口　秀樹, 渡邊　浩太, “高速行列計算利用のための任意多倍長精度整数型除算 LSI 回路の検討”, 電子情報通信学会 和文論文誌 C, Vol.J106-C, No.10. 10.2023.

International Conference Proceedings

[1] Chenxu Wang, Seiya Ota, Hideki Kawaguchi, "Conceptual Design of Dataflow Machine for Magnetostatic Field Simulation", 2021 International Conference on Electromagnetics in Advanced Applications (ICEAA), 2021.8.

[2] Chenxu Wang, Hideki Kawaguchi, "Design study of FIT dataflow machine for 3D electrostatic field simulation",20th International Symposium on Applied Electromagnetics Mechanics (ISEM), 2022.6.

[3] Chenxu Wang, Hideki Kawaguchi, “Conceptual Design of FIT Dataflow Machine for 2-D Time Domain Eddy Current Field Simulation”, 24th International Conference on the Computation of Electromagnetic Fields (COMPUMAG 2023), 2023.6.

Conference Proceedings

[1] Chenxu Wang, Seiya Ota, Hideki Kawaguchi, "Design study of BiCG-Stab matrix solver circuit for FIT scheme based on dataflow architecture", The 40th JSST Annual International Conference on Simulation Technology in Japan, 2021.9.

[2] Chenxu Wang, Hideki Kawaguchi, "Conceptual Design of Dataflow Machine for 2-D eddy current fields Simulations", The 41th JSST Annual International Conference on Simulation Technology in Japan, 2022,9.

[3] Wang Chenxu, Ota Seiya, Kawaguchi Hideki, "Design study of hardware circuit of FIT scheme based on dataflow architecture for 2D magnetostatic field simulation", 電磁界理論研究会, EMT-21-071, 2021.11.

[4] Wang Chenxu, Kawaguchi Hideki, "Conceptual design of FIT scheme dataflow machine for 3D electrostatic field simulation", 電磁界理論研究会, EMT-22-023 2022.1.

[5] Chenxu WANG, Hideki KAWAGUCHI, "Conceptual Design of FIT Dedicated Computer with Dataflow Architecture for 2-D Magnetostatic Field Simulation", 第 30 回　MAGDA　カンファレンス　in 広島, 2021.12.

[6] Chenxu WANG, Hideki KAWAGUCHI, " Generation of millimeter-wave vortex field by using corrugated waveguide and FDTD analysis of propagation in magnetized plasma", 第 31 回　MAGDA　カンファレンス　in 鹿児島, 2022.11.