L-4

Hardware Acceleration of an Implicit FDTD Method

*Tomoaki Ichikawa [1],Shinichiro Ohnuki[2]

Abstract: The finite-difference time-domain method is widely used as one of powerful computational techniques, however the computational cost becomes very expensive for nano-scale electromagnetic problems because of the Courant-Friedrich-Lewy stability condition. In this paper, we investigate hardware acceleration using Graphics Processing Units for an implicit method.

1. Introduction

The finite-difference time-domain (FDTD) method is a popular algorithm to solve electromagnetic problems. This method is useful, however the selectable maximum time-step becomes very small for nano-scale problems because of the Courant-Friedrich-Lewy (CFL) stability condition. To avoid CFL condition, we investigate the Alternating-Direction (ADI) FDTD method[1] known as one of implicit methods and use GPUs to cut down the computational cost by the hardware acceleration.

GPUs are specialized processors to deal with 3D Graphic rendering and suitable for parallel computing, since they have many computational cores. Recently, techniques using GPUs for the general purpose computing have been reported[2][3]. In this paper, we study availability of the ADI FDTD method accelerated by GPUs for fast electromagnetic simulation.

2. Formulation

We study hardware acceleration of the ADI FDTD method by GPUs. The ADI FDTD method is described as the following two steps. The first step can be expressed by

$$\begin{bmatrix} \alpha & -\beta & 0 & \cdots & \cdots & 0 \\ -\beta & \alpha & -\beta & 0 & & \vdots \\ 0 & -\beta & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -\beta & 0 \\ \vdots & & \ddots & -\beta & \alpha & -\beta \\ 0 & \cdots & \cdots & 0 & -\beta & \alpha \end{bmatrix} \cdot \begin{bmatrix} H_y^{n+1/2}(0,j) \\ H_y^{n+1/2}(1,j) \\ \vdots \\ H_y^{n+1/2}(i,j) \\ \vdots \\ H_y^{n+1/2}(i_{\max},j) \end{bmatrix} = \begin{bmatrix} H_y^{n}(0,j) \\ H_y^{n}(1,j) \\ \vdots \\ H_y^{n}(i,j) \\ \vdots \\ H_y^{n}(i_{\max},j) \end{bmatrix} \quad (1)$$

The left-hand side coefficient matrix and the right-hand side vectors H_y^n are known, and $H_y^{n+1/2}$ vectors are unknown. We can obtain the unknown vector to multiple the inverse matrix of left-hand side coefficient matrix by both sides. In this step, we compute only x direction of matrix elements. The second step is given by

$$\begin{bmatrix} \alpha & -\beta & 0 & \cdots & \cdots & 0 \\ -\beta & \alpha & -\beta & 0 & & \vdots \\ 0 & -\beta & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -\beta & 0 \\ \vdots & & \ddots & -\beta & \alpha & -\beta \\ 0 & \cdots & \cdots & 0 & -\beta & \alpha \end{bmatrix} \cdot \begin{bmatrix} H_x^{n+1}(i,0) \\ H_x^{n+1}(i,1) \\ \vdots \\ H_x^{n+1}(i,j) \\ \vdots \\ H_x^{n+1}(i,j_{\max}) \end{bmatrix} = \begin{bmatrix} H_x^{n+1/2}(i,0) \\ H_x^{n+1/2}(i,1) \\ \vdots \\ H_x^{n+1/2}(i,j) \\ \vdots \\ H_x^{n+1/2}(i,j_{\max}) \end{bmatrix} \quad (2)$$

The second step is the same as the first step of the ADI FDTD method. The left-hand side vectors H_x^{n+1} are corresponding to the first step of $H_y^{n+1/2}$ and the right-hand side vectors $H_x^{n+1/2}$ are corresponding to the first step of H_y^n. In the second step, we compute only y direction of matrix elements. After computing both steps, we can obtain the result which is corresponding to that by an explicit FDTD method at one time-step. In the matrix calculations, we do not have to calculate all matrix components because these are sparse matrices. For fast parallel computing by GPUs, we divide the whole computational field into small blocks which are distributed to all GPU cores.

3. Numerical results

We analyze wave propagation in 2-D free space by the ADI FDTD method. The step sizes of computational field Δx and Δy are 0.01 [m]. The incident wave is a sinusoidal wave and the frequency is 400 MHz. The time-step Δt is selected to satisfy the following CFL

1: Graduate Course of Electrical Engineering, Graduate School of Science and Technology, Nihon University.

2: Department of Electrical Engineering, College of Science and Technology, Nihon University

stability condition:

$$\Delta t \le \Delta t_{CFL} = \frac{1}{c_0\sqrt{(1/\Delta x)^2 + (1/\Delta y)^2}} \qquad (3)$$

where c_0 is speed of light. The value of Δt (=11.7[ps]) is set to a half of Δt_{CFL}.

Figure 1 shows the computational time of the ADI FDTD method. Circles indicate computational time using CPU which is Intel® Core™2Quad 2.83GHz and squares indicate computational time using GPU which is NVIDIA Geforce9600GT. When the size of computational field is $1024\Delta x$ x $1024\Delta y$, the computation using GPU is 25 times faster than that using CPU.

Figure 2 shows the electric field intensity obtained by the ADI FDTD method using CPU and GPU. The both curves agree very well.

Figure 3 shows the wave propagation for varying the time-steps Δt, $2\Delta t$, $5\Delta t$, and $10\Delta t$. All the results are in good agreement. We confirm that the computational time is inversely proportional to the time-steps.

4. Conclusions

We investigate the potential of hardware acceleration for the ADI FDTD method using GPUs. The computational time is 25 times faster than that using a conventional CPU. We also verify that reliable simulation can be performed when the time-step is selected 5 times as large as that satisfied CFL stability condition.

5. Reference

[1] T. Namiki, "A New FDTD Algorithm Based on Alternating-Direction Implicit Method", *IEEE Trans. Microw. Theory Tech.*, vol.47, No.10, 1998.

[2] T. Ichikawa and S. Ohnuki, The Institute of Electrical Engineers of Japan Symposium, 1-013, p21, 2010. (in Japanese)

[3] M. Acuna, T. Aoki, "Multi-GPU Computing and Scalability for Real-Time Tsunami Simulation", IPSJ Symposium Series., Vol.2010, No.1, 2010.

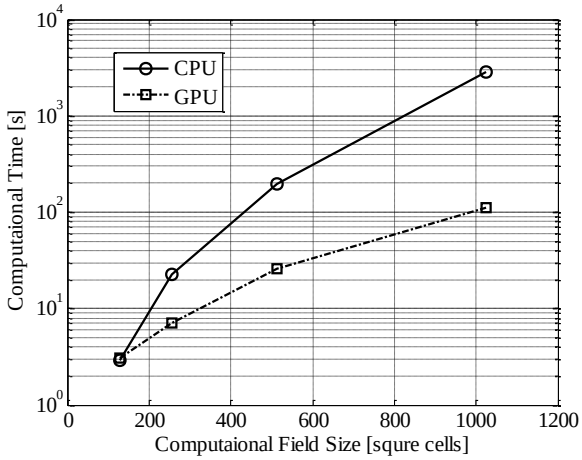

Figure 1. Computational time.

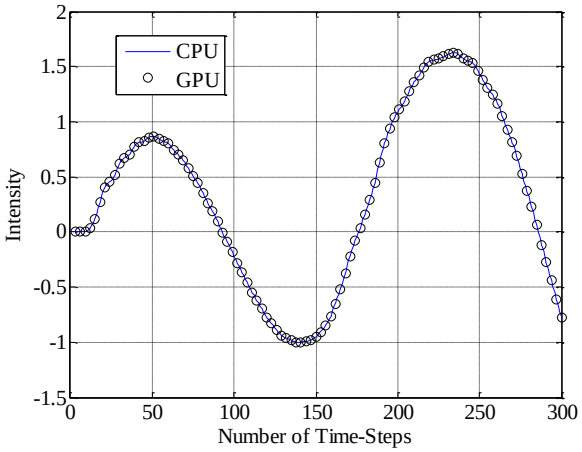

Figure 2. Time domain response computed by CPU and GPU.

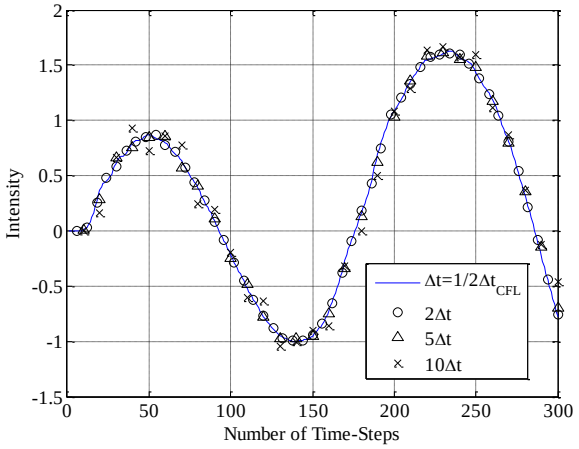

Figure 3. Time domain response for various time-steps.