POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea Magistrale

Titolo

Relatori:
Prof. Mariagrazia Graziano
Prof. Marco Vacca
Prof. Maurizio Zamboni

Candidata:
Gina Jiang

2017

Acknowledgments

Table of contents

List of figures

Chapter 1

Magnetostatic field calculation 3D

In general, the magnetostatic field is connected to the magnetization $[\overrightarrow{M}]$ through the magnetizing tensor $[\mathbf{TG}]$.

The evaluation of the magnetostatic field of a given cell requires the summation on all cells of the mesh due to the the long range of the dipole interaction.

$$\overrightarrow{H}(ijk) = -M_s \sum_{i'=1}^{N_x} \sum_{j'=1}^{N_y} \sum_{k'=1}^{N_z} \begin{bmatrix} TG_{xx} & TG_{xy} & TG_{xz} \\ TG_{yx} & TG_{yy} & TG_{yz} \\ TG_{zx} & TG_{zy} & TG_{zz} \end{bmatrix}_{(i-i',j-j',k-k')} \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix}_{(i',j',k')} \tag{1.1}$$

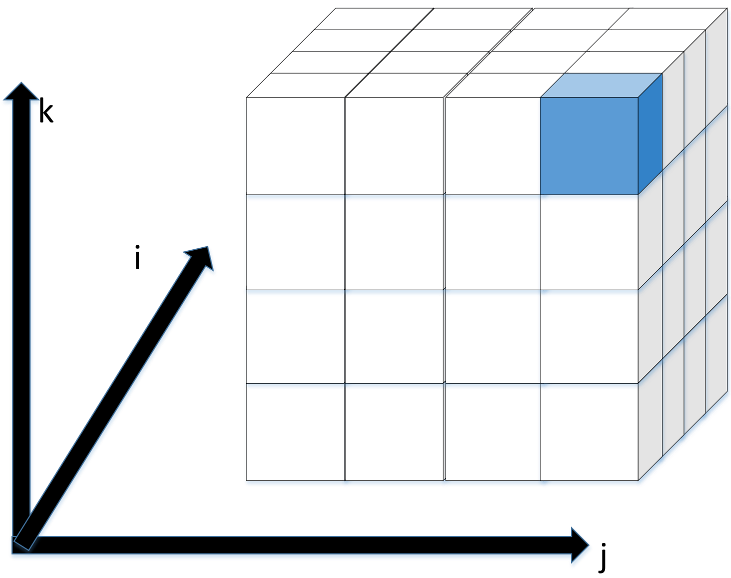

Figure 1.1: The selected cell is supposed to sum all the magnetostatic field contributions between it and all the other cells

1.1 Product TG and M

Each cell $(i'j'k')$ has a finite state machine that evaluates the magnetostatic field contribution between the cell *ijk* and the cell *i'j'k'*.

$$\overrightarrow{h}(i-i',j-j',k-k') = \begin{bmatrix} TG_{xx} & TG_{xy} & TG_{xz} \\ TG_{yx} & TG_{yy} & TG_{yz} \\ TG_{zx} & TG_{zy} & TG_{zz} \end{bmatrix}_{(i-i',j-j',k-k')} \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix}_{(i',j',k')}$$

.

The matrix $\mathbf{TG}_{(i-i',j-j',k-k')}$ is the magnetizing tensor between the 2 cells, while $\overrightarrow{M}_{(i',j'k')}$ is the magnetization of the cell i',j',k'.

The FSM is composed by the following states:

- **READ_MX**: save m_x value
- **READ_MY**: save m_y value
- **READ_MZ**: save m_z value
- **READ_TGxx**: read TG_{xx} value and computing $op1 = TG_{xx} \cdot m_x$
- **READ_TGxy**: read TG_{xy} value and computing $op2 = TG_{xy} \cdot m_y$
- **READ_TGxz**: read TG_{xz} value and computing $h_x = TG_{xz} \cdot m_z + op1 + op2$
- **READ_TGyx**:read TG_{yx} value and computing $op1 = TG_{yx} \cdot m_x$
- **READ_TGyy**:read TG_{yy} value and computing $op2 = TG_{yy} \cdot m_y$
- **READ_TGyz**: read TG_{yz} value and computing $h_y = TG_{yz} \cdot m_z + op1 + op2$
- **READ_TGzx**:read TG_{zx} value and computing $op1 = TG_{zx} \cdot m_x$
- **READ_TGzy**:read TG_{zy} value and computing $op2 = TG_{zy} \cdot m_y$
- **READ_TGzz**: read TG_{zz} value and computing $h_z = TG_{zz} \cdot m_z + op1 + op2$

1.2 Logic Plane

This component group all the cells in a given plane, and enable the computation of

$$\overrightarrow{h}(i-i',j-j',k-k^*) \qquad \forall i' < N_x,\ j' < N_y,\ k = k^*$$

in parallel.

1.3 Sum all

The *sum_all* is a component that given a fixed k^* (i.e. a given plane) evaluates the magnetostatic field of all cells in a fixed plane. The input is given by the *logic plane*

$$\sum_{i'=1}^{N_x}\sum_{j'=1}^{N_y}\begin{bmatrix} TG_{xx} & TG_{xy} & TG_{xz} \\ TG_{yx} & TG_{yy} & TG_{yz} \\ TG_{zx} & TG_{zy} & TG_{zz} \end{bmatrix}_{(i-i',j-j',k-k^*)} \begin{bmatrix} m_x \\ m_y \\ m_z \end{bmatrix}_{(i',j',k^*)}$$

By looking to the formula 1.1, we have to evaluate the sum for each plane, so we store this sum and wait for the next plane to be computed and added to the previous sum.

Of course when we reach the k^{th} plane, we stop the sum and we set a signal to tell that the calculation is finished.

Bibliography

[1] Performance driven FPGA design with an ASIC perspective. Andreas Ehliar, 2009

[2] K.-C. Wu and Y.-W. Tsai, "Structured asic, evolution or revolution?"in ISPD ?04: Proceedings of the 2004 international symposium on Physical design. New York, NY, USA: ACM, 2004, pp. 103?106.

[3] I. Kuon and J. Rose, "Measuring the gap between fpgas and asics",in Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 2007.

[4] Actel, "Igloo low-power flash fpgas handbook,"2008. [Online]. Available: http://www.actel.com/documents/IGLOO_HB.pdf

[5] T. Tuan, S. Kao, A. Rahman, S. Das, and S. Trimberger, "A 90nm lowpower fpga for battery-powered applications", in Proceedings of the 2006 ACM/SIGDA 14th international symposium on Field programmable gate arrays, 2006, pp. 3-11

[6] 2013 by Yong Cao, Referencing UIUC ECE408/498AL Course Notes (Lecture 10 - Virginia Tech)

[7] Prefix sum - Parallel algorithm - Wikipedia

[8] Patrick Cozzi,University of Pennsylvania

[9] Qingqing Dang, Shengen Yan, Ren Wu, A fast integral image generation algorithm on GPUs, 2014 IEEE

[10] 2015, Mario Cofano, Mariagrazia Graziano, Maurizio Zamboni, Design of a Logic-in-Memory architecture for massive parallel algorithms

[11] PRACTICAL FAST 1-D DCT ALGORITHMS WITH 11 MULTIPLICATIONS, Christoph Loeffler, Adriaan Lieenberg, and George S. Moschytz

[12] 2015, Filasieno Francesco, Garino Simone, Garlando Umberto, Progettazione di Sistemi a Basso Consumo : Array Sistolico Riconfigurabile

[13] LLM Integer Cosine Transform and Its Fast Algorithm Chi-Keung Fong, Student Member, IEEE, and Wai-Kuen Cham, Senior Member, IEEE

[14] High Order Integer Transform Design for Video Compression,Jie Dong, Member, IEEE, and Yan Ye, Member, IEEE

[15] DIGITAL IMAGE FILTERING By Fred Weinhaus

[16] Integrated system architecture, lecture notes by E. Raviola, 2017

[17] A Guide to Numerical Methods for Transport Equations, Dmitri Kuzmin

[18] Analysis and Design of an Optimized HW Accelerator for Protein Alignment, Gianvito Urgese, Mariagrazia Graziano, Maurizio Zamboni, September 2012, Politecnico di Torino

[19] A New Non-Restoring Square Root Algorithm and Its VLSI Implementations, Yamin Li and Wanming Chu, International Conference on Computer Design (ICCD?96), October 7?9, 1996, Austin, Texas, USA