3K-7

Design of Synchronization Mechanism to Conquer the Clock Oscillator Variation for High Performance Stencil Computation Accelerator

Ryohei Kobayashi † Shinya Takamaeda-Yamazaki †‡ Kenji Kise †
†Graduate School of Information Science and Engineering Tokyo Institute of Technology
‡JSPS Research Fellow (DC1)

1 Introduction

Stencil computation is one of the typical scientific computing kernels[1]. Various accelerators to solve stencil computation at high speed are designed by using multiple high end FPGAs[2].

We have developed an effective stencil computation accelerator by using multiple FPGAs, which employs 2D-mesh architecture connecting multiple small FPGAs[3]. On the process of the development, there is a trouble that the system generates an illegal computation result when the multiple FPGA nodes are used. The cause is clock period variation.

This paper describes a quantitative evaluation result of clock variations for every FPGA node and the design and implementation of a mechanism to operate the synchronization of data.

2 Parallel Stencil Computation by using Multi-FPGA

Figure 1 shows a typical pattern of 2D stencil computation. k in the figure represents time-step. Each circle represents a value of grid-point and each value of grid-point at next time-step (k+1) is computed by using the values of its four adjacent grid-points at current time-step (k). (i, j) represents coordinate of grid-point. Two buffers, V0 and V1, are used for the computation. As shown in Figure 1, V1[i][j] is updated by the summation of four values. The each value is obtained by multiplying weighting factor by one adjacent grid-points (V0[i-1][j], V0[i][j-1], V0[i][j+1], V0[i+1][j]). Finally, every grid-point is updated for the next time-step.

As shown in Figure 1, the data set of stencil computation is divided into several blocks according to the number of vertical and horizontal array of FPGAs. Each data block is assigned to each FPGA. The computation on each FPGA uses the assigned data and the boundary data of each block shared. The necessary boundary data of the adjacent FPGAs have to be sent to. In Figure 1, a group of grid-points (4×4) is assigned one FPGA, an arrow represents communication to the neighbor FPGA. Gray regions represent the data subset communicated to other FPGAs.

3 Clock Period Variation

Each FPGA node in our system is equipped with the clock oscillator (CSX-750PB(B)) with operation frequency 40MHz, and the frequency stability of the oscillator is ±50ppm. ±50ppm of the frequency stability means the clock period variation per 1,000,000 cycles is within ±50 cycles.

As a preliminary evaluation, we evaluated the clock period variation in each FPGA node. The clock period

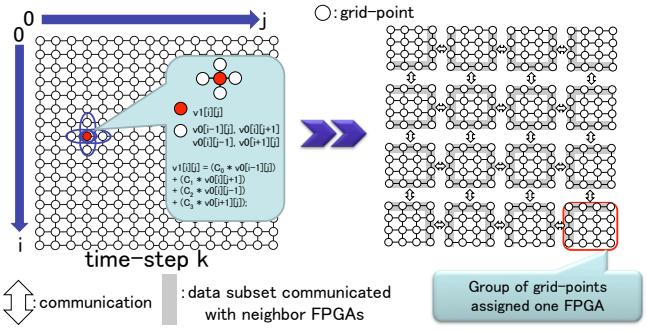

Figure 1: Parallel Stencil Computation by using Multi-FPGA.

variation is cited as the cause of trouble that the system generates an illegal computation result when the multiple FPGA nodes are used. We used ScalableCore system to measure the clock period variation. ScalableCore system emulates an environment for many-core processors, and the many-core processors are synchronized with the concept named virtual cycle. In order to examine the frequency stability by using this mechanism, we coded the application in C, which is working in many-core processors on the ScalableCore system.

In this paper, we measured the clock period variation of each FPGA node by running the implemented program, and varied measurement time for each measurement. We targeted the FPGA array (8×8) and the results are shown in Figure 2, Figure 3 and Table 1.

Figure 2 shows the clock period variation depending on measurement time (20sec). The Z axis in these figures represents the clock period variation per 1,000,000 cycles. The X and Y axes represent the position coordinate of each FPGA node. As shown in Figure 2, the clock period variation per 1,000,000 cycles keeps within ±50 cycles. This result means that it is guaranteed that the frequency stability of the oscillator is ±50ppm.

Table 1 shows that the worst clock period variation and standard deviation of the clock variation in each measurement time. The Time, Worst Value and Standard Deviation in Table 1 represent measurement time, the worst clock period variation and standard deviation of the clock variation on FPGA array (8×8) respectively. As shown in Table 1, the Worst Value and the Standard Deviation did not change nearly though we changed measurement time. This result means that the clock period variation does not depend on measurement time.

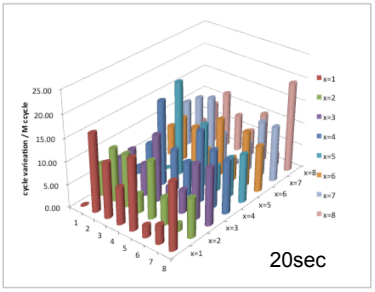

Figure 2: Clock Period Variation (Measurement Time :20sec).

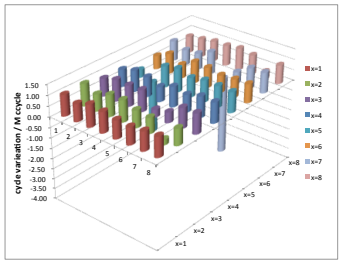

Figure 3: Relative Clock Period Variation.

Table 1: Measurement Result

Time[sec]	Worst Value[ppm]	Standard Deviation
20	20.47 (x=3, y=5)	4.73
40	20.47 (x=3, y=5)	4.68
80	20.47 (x=3, y=5)	4.73
160	20.59 (x=3, y=5)	4.77
320	20.66 (x=3, y=5)	4.79

Figure 3 shows the result which is the clock period variation (320sec) divided by the variation (20sec). The X, Y axes in Figure 3 represent the same in Figure 2. The Z axis in the figure represents the clock period variation per 1,000,000 cycles with respect to the result (20sec). As shown in Figure 3, the clock period variation in almost all the nodes shows about 1 [ppm]. This result also shows that the clock period variation does not depend on measurement time.

From these results, it is clear that the system must be designed considering the clock period variation which almost does not change on time axis.

4 Design and Implementation of Synchronization Mechanism

We designed and implemented the mechanism to synchronize all FPGA nodes in order to operate the system successfully. The mechanism absorbs the clock period variation.

We describe the design of synchronization mechanism and Figure 4 (a) shows it. We define FPGA node (A) in Figure 4 (a) as Master node. Each FPGA node executes stencil computation in synchronism with the signal transmitted from Master node. The nodes other than Master node stall stencil computation until receiving a synchronization signal.

The synchronization signal is generated by Master node in a period of $\alpha + \beta$ and sent to the other nodes. The α represents the number of cycles required to execute stencil computation between one "Iteration". In this paper, the Iteration is defined a sequent process to compute all the grid-points at a time-step. The β represents the margin to absorb the clock period variation of each FPGA node. The β represents the margin to absorb the clock period variation of each FPGA node. This margin must be the value which hides the worst clock period variation in α cycles.

We describe the implementation of synchronization mechanism and Figure 4 (b) shows it. The α and β in Figure 4 (b) are same as the α and β in Figure 4 (a). The FPGA nodes other than Master node, which receive a synchronization signal, send the synchronization signal to right and down FPGA nodes. Therefore, all FPGA nodes are synchronized to Master node. The FPGA node receiving the synchronization signal restarts to execute stencil computation after waiting several cycles. The reason why the FPGA node waits several cycle is to prevent chattering.

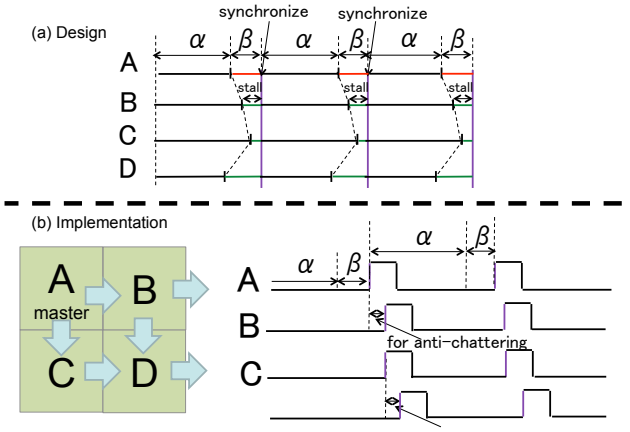

Figure 4: Design and Implementation of Synchronization Mechanism.

5 Conclusion

This paper describes a quantitative evaluation result of clock variations for every FPGA node and the design and implementation of a mechanism to operate the system successfully. The future work is to evaluate the performance of the accelerator by using multiple FPGAs.

Acknowledgment

This work is supported in part by Core Research for Evolutional Science and Technology (CREST), JST.

References

[1] Kaushik Datta et al. Stencil computation optimization and auto-tuning on state-of-the-art multicore architectures. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, SC '08, pp. 4:1–4:12, Piscataway, NJ, USA, 2008. IEEE Press.

[2] K. Sano et al. Scalable streaming-array of simple soft-processors for stencil computations with constant memory-bandwidth. In *Field-Programmable Custom Computing Machines (FCCM), 2011 IEEE 19th Annual International Symposium on*, pp. 234 –241, may 2011.

[3] R. Kobayashi et al. Towards a low-power accelerator of many fpgas for stencil computations. In *Networking and Computing (ICNC), 2012 Third International Conference on*, pp. 343 –349, dec. 2012.