A Design Procedure for a Large-Scale Reconfigurable Data-Path

Honda, Hiroaki
Institute of Systems, Information Technologies and Nanotechnologies

Mehdipour, Farhad
Graduate School of Information Science and Electrical Engineering, Kyushu University | Institute of Systems, Information Technologies and Nanotechnologies

Kataoka, Hiroshi
Graduate School of Information Science and Electrical Engineering, Kyushu University | Institute of Systems, Information Technologies and Nanotechnologies

Inoue, Koji
Graduate School of Information Science and Electrical Engineering, Kyushu University | Institute of Systems, Information Technologies and Nanotechnologies

他

出版情報：情報処理学会研究報告, 2009-ARC-182. 2009 (14), pp.97-102, 2009-02-19. 情報処理学会
バージョン：

A Design Procedure for a Large-Scale Reconfigurable Data-Path

Hiroaki HONDA[†], Farhad MEHDIPOUR[††], Hiroshi KATAOKA[††], Koji INOUE[††], and Kazuaki MURAKAMI[††]

[†]Institute of Systems, Information Technologies and Nanotechnologies, Fukuoka 814-0001, Japan
[††]Graduate School of Information Science and Electrical Engineering Kyushu University, Fukuoka 819-0395, Japan

E-mail: dahon@isit.or.jp, {farhad, kataoka}@c.csce.kyushu-u.ac.jp, {inoue,murakami}@i.kyushu-u.ac.jp

Abstract Large-Scale Reconfigurable Data-Path (LSRDP) processor has been proposed for the reduction of required memory bandwidth in a high performance scientific computing. LSRDP micro-architecture design procedure and how it is exploited are presented. First, 24 benchmark Data Flow Graphs (DFGs) are extracted from 2nd order partial differential equations and electron repulsion integral calculations. LSRDP architectural specifications including height, width, and maximum connection length between floating point units are obtained through analyzing the statistics gathered from the compilation of DFGs.

Keyword Large-Scale Reconfigurable Data-Path, Data Flow Graph, Architecture Design

大規模再構成可能データパスプロセッサの設計手法

本田宏明[†], ファラハドメディプー[††], 片岡広志[†††], 井上弘士[††], 村上和彰[††]

[†]九州先端科学技術研究所, [††]九州大学システム情報科学研究院, [†††]九州大学システム情報科学府

概要 要求メモリバンド幅を抑えつつも高性能な科学技術計算を可能とするような，大規模再構成可能データパスプロセッサ （LSRDP）が提案されている．本稿ではこの LSRDP について，詳細なアーキテクチャ設計を行った．２階の偏微分方程式ならびに二電子積分からの24 個のベンチマークとなるデータフローグラフを使用し，マッピングツールの結果から統計的に LSRDP の高さ、幅、演算器間の最大結合距離を求めた．

Keyword 大規模再構成可能データパス，データフローグラフ，アーキテクチャ設計

1. Introduction

Computer systems based on parallel computer clusters with General-Purpose Processors (GPP) are often utilized for the high performance computing. Those parallel computers with GPPs account for a large share of the performance ranking in TOP500[1]. On the other hand, the hybrid architecture comprising an accelerator augmented to a GPP might be chosen for special purpose computations. The accelerator should be designed to feature small size, high performance, and low power consumption.

Recent examples of such accelerator are CSX600 PCI-X board[2], GRAPE-DR processor[3], Cell processor[4] which is heterogeneous multi-core processor and General Purpose computing on Graphic Processing Unit (GPGPU) calculations are often used by graphic accelerator chips[5]. Those accelerators commonly have Single Instruction Multiple Data stream (SIMD) mechanism for total architecture, or functional units.

Generally, a large memory bandwidth is demanded in conventional accelerators to perform calculations efficiently. Therefore, an on-chip memory can be utilized for reduction of the required memory bandwidth.

Recently, the Large-Scale Reconfigurable Data-path (LSRDP) processor is proposed by Murakami et al. [6-8] for reducing memory pressure itself on the system performance. The

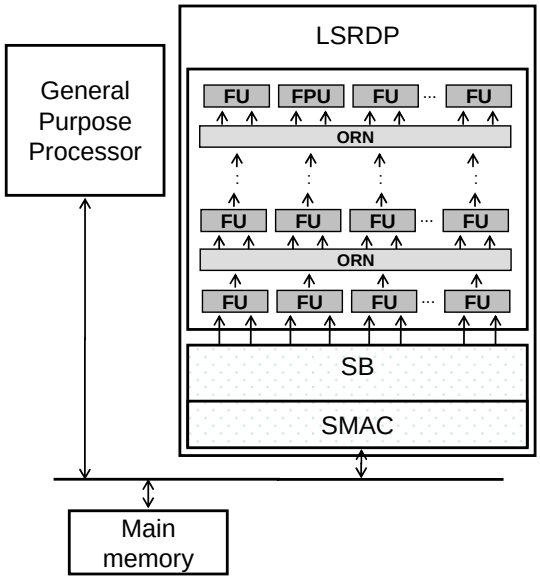

Fig.1: LSRDP system

LSRDP architecture comprises a two dimensional array of floating point Functional Units (FUs) and interconnection networks among FUs referred as Operand Routing Networks (ORNs). The main intuition behind the LSRDP is that the cascaded FUs can generate a final result without temporally memorizing intermediate data, therefore, the number of memory load/store operations corresponding to spill codes can be reduced (Fig. 1).

In the CREST SFQ-RDP project[9], developing algorithm for LSRDP and specifying LSRDP architecture configuration are the main tasks.

In this paper, LSRDP detailed architecture design procedure and the results are shown. In Section 2, LSRDP architecture is introduced. In Section 3, architecture designing methodology, benchmark application, and detailed architecture design scheme are explained. Section 4 presents the results of architecture design procedure. Finally, Section 5 concludes the paper.

2. Outline of LSRDP architecture

2.1. On chip design parameters

The LSRDP is shown in Fig.1. LSRDP system is constituted with LSRDP accelerator chip, GPP and main memory are connected through a shared bus to each other. Generally, LSRDP is configured as a two-dimensional array of FUs connected with flexible ORNs, Streaming Buffer (SB) which is the FIFO type buffer, and Streaming Memory Access Controller (SMAC) for data I/O. Each FU can be fed data through SMAC and SB to one or more FUs via ORN switches. Feedback data flow connections are not supported, which means that the flow of data in the FU array is only in one direction from input to output. The LSRDP should be an adaptable accelerator, because it is aimed to target various scientific applications. In order to satisfy this requirement, the LSRDP is featured with dynamically reconfiguring of the ORNs. Originally, the ORN consists of programmable switches. By means of setting the control signals provided with FUs and ORN switches, the function of the LSRDP can be configured at run time. Such flexibility makes it possible to implement various DFGs on the FU array.

In an LSRDP, a data flow graph (DFG) extracted from a target application program is mapped to the two dimensional FU array. Since the cascaded FUs can generate a final result without temporally memorizing intermediate data, we can reduce the number of memory load/store operations corresponding to spill codes. Therefore, memory bandwidth required to achieve a high performance can be reduced. Furthermore, since a loop-body mapped into the FU array is executed in a pipeline fashion, LSRDP can provide a high computing throughput.

2.2. On chip design parameters

In the LSRDP design stage, following architectural specifications have to be decided:

- Type and granularity of each functional unit (FU)
- Configuration of processing element (PE)
- LSRDP height and width
- Number of I/O ports
- Size of Operand Routing Networks (ORNs), especially maximum connection length between consecutive rows
- Layout of FU operation types
- Reconfiguration mechanism

In following sections above specifications are decided except Layout of FU and reconfiguration mechanism. Reconfiguration mechanism is strongly depends on detailed properties of the LSRDP micro-architecture.

3. Architecture designing scheme

3.1. Design methodology

Different strategies can be used for determining LSRDP architectural specifications in detail. Our approach is based on the quantitative analysis of DFGs of benchmark applications and their mapping results onto the LSRDP. Fig.2 shows the flow of the design

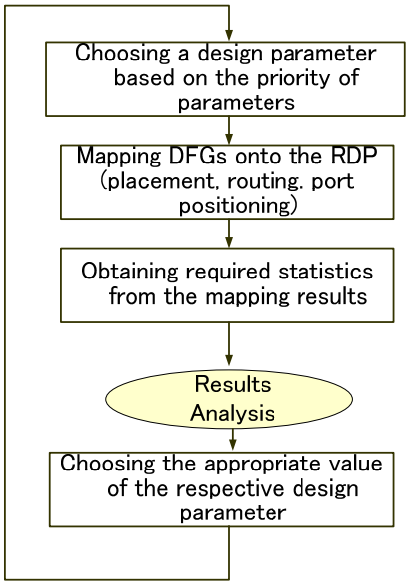

Fig.2: Design flow for LSRDP design based on the quantitative approach

stages. As the design flow is an iterative procedure of gathering statistics and analysis of results, therefore, the designer should decide the priority of specifications in the first step. Then, for determining each design parameter, applications (extracted DFGs) should be mapped on the LSRDP. There is no limitation in the initial architecture and the mapping process is performed without forcing any constraint. In the next stage, the results of mapping should be analyzed by the designer to decide an appropriate value for the intended parameter. This process is repeated to specify the entire specifications of the architecture. In the following sections more details on those procedures will be given.

3.2. Benchmark application

We utilized totally twenty four DFGs as a benchmark set to design the LSRDP architecture. Four applications are selected. One-dimensional heat and vibration equation, two-dimensional Poisson equation, and recursion calculation part of Electron Repulsion Integral (ERI) as a quantum chemistry application. All calculations are constructed based on ADD, SUB, and MUL operations.

Generally, 2-dimensional 2nd order partial differential equations with constant coefficients are categorized to three types: heat or diffusion equation, vibration equation and Poisson equation. Each equation has following canonical form, respectively:

$$\frac{\partial T(x,t)}{\partial t} = A\frac{\partial^2 T(x,t)}{\partial^2 x} \tag{1}$$

$$\frac{\partial^2 V(x,t)}{\partial t^2} = A\frac{\partial^2 V(x,t)}{\partial x^2} \tag{2}$$

$$\frac{\partial^2 u(x,y)}{\partial x^2} + \frac{\partial^2 u(x,y)}{\partial y^2} = f(x,y) \tag{3}$$

These equations are solved by finite difference method using following expressions[10]:

$$T(x_i,t_{j+1}) = D*T(x_i,t_j) + B*\left[T(x_{i-1},t_j) + T(x_{i+1},t_j)\right] \tag{4}$$

$$V(x_i,t_{j+1}) = D*V(x_i,t_j) + B*\left[V(x_{i-1},t_j) + V(x_{i+1},t_j)\right] + C*V(x_i,t_{j-1}) \tag{5}$$

$$u^{(n+1)}(x_i,y_j) = (1-\omega)*u^{(n)}(x_i,y_j) + \frac{\omega}{4}\left[u^{(n)}(x_{i-1},y_j) + u^{(n)}(x_{i+1},y_j) + u^{(n)}(x_i,y_{j-1}) + u^{(n)}(x_i,y_{j+1}) - h^2 f(x_i,y)\right] \tag{6}$$

Here, D, B, C and ω are constants. By using Eq.(6) which is an iterative equation (referred as successive over relaxation method), final *u(x,y)* is calculated as converged form.

In the next stage, benchmark DFGs are manually extracted from Eq. (4), (5) and (6), then mapped on the LSRDP by utilizing a mapping tool which will be explained in the following section. However, it is inefficient to map only small DFGs which are extracted directly from Eqs. (4)-(6). Therefore larger DFGs are generated through connecting the smaller ones.

For example, in heat equation, extracted DFG which corresponds to Eq.(4) can be shown as in Fig.4. This finite difference equation shows that the next point during the time evolution process: (x_i,t_{j+1}) is obtained by us-

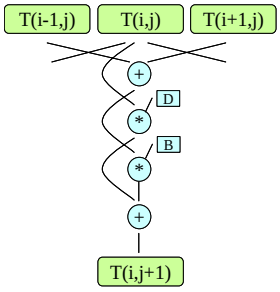

Fig.3: Data Flow Graph of minimum unit of heat equation

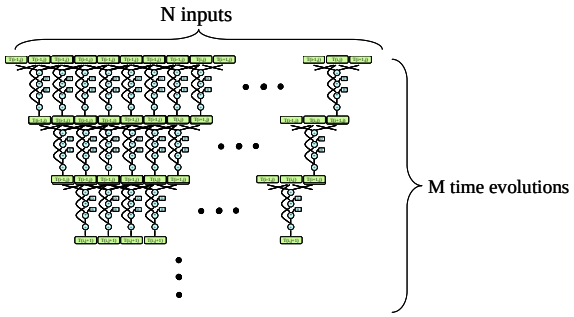

Fig.4:Connection of Data Flow Graphs

ing current three points: (x_i,t_{j+1}) (x_i,t_{j+1}) (x_i,t_{j+1}). If this equation is applied to the 1-dimentional $N+1$ points: $x_{i-N/2} \sim x_{i+N/2}$ and is iterated M times to calculate from t_j to t_{j+M}, then, this equation have to be iteratively applied $\sim N*M$ times. By extending that equation over the space and time dimensions, the final computation structure will correspond to the DFGs in Fig.7.

By connecting N DFGs over the space and M over the time directions, finally we obtain a large DFG with N inputs and $N\text{-}2^M$ outputs, $4*(2N\text{-}2^M)*M/2$ operations. In consequence, by implementing the obtained DFG on the LSRDP, numerous operations are possible to be executed in each local clock of the LSRDP pipelined architecture.

Similar methods of the above mentioned large DFG generation procedure are applicable to basic DFGs of vibration and Poisson equations.

For the electron repulsion calculation, recursion calculation parts[12] are described as Fig.5. In the formula, $(p_is,ss)^{(n)} \sim (p_ip_jp_kp_l)^{(n)}$ are objective integral values. $(ss,ss)^{(n)}$ is initial integral value, and all the other values are coefficients. Since each i,j,k,l index corresponds to space three components: x,y,z, each integral expression has multiple components. For example, $(p_ip_jp_kp_l)^{(n)}$ has 81 components and corresponding DFG has 81 outputs. Dividing DFG to smaller ones would be useful if the DFG size is lager than the number of resources in the LSRDP. In this case, vertical partitioning of DFG is practical to decrease the number of outputs and operations.

Numbers of final benchmark DFGs are summarized in Table 1.

Table 1: Numbers of DFGs for each applications

Application	Heat	Vibration	Poisson	ERI
# of DFG	6	7	3	8

$$
\begin{aligned}
(p_is,ss)^{(0)} &= PA_i(ss,ss)^{(0)} + WP_i(ss,ss)^{(1)} \\
(p_is,p_ks)^{(0)} &= QC_k(p_is,ss)^{(0)} + WQ_k(p_is,ss)^{(1)} \\
&\quad + \frac{\delta_{ik}Z_{ab}}{2}(ss,ss)^{(1)} \\
(p_ip_j,ss)^{(0)} &= PB_j(p_is,ss)^{(0)} + WP_j(p_is,ss)^{(1)} \\
&\quad + \delta_{ij}Z_a\{(ss,ss)^{(0)} - \rho Z_a(ss,ss)^{(1)}\} \\
(p_ip_j,p_ks)^{(0)} &= QC_k(p_ip_j,ss)^{(0)} + WQ_k(p_ip_j,ss)^{(1)} \\
&\quad + \frac{Z_{ab}}{2}\{\delta_{ik}(sp_j,ss)^{(0)} - \delta_{jk}(p_is,ss)^{(1)}\} \\
(p_ip_j,p_kp_l)^{(0)} &= QD_l(p_ip_j,p_ks)^{(0)} + WQ_l(p_ip_j,p_ks)^{(1)} \\
&\quad + \frac{Z_{ab}}{2}\{\delta_{ik}(sp_j,p_ks)^{(0)} - \delta_{jl}(p_is,p_ks)^{(1)}\} \\
&\quad + \delta_{kl}Z_b\{(p_ip_j,ss)^{(0)} - \rho Z_b(p_ip_j,ss)^{(1)}\} \\
(i,j,k,l &= x,y,z)
\end{aligned}
$$

Fig.5: Electron Repulsion Integral Recursion formula

3.3. LSRDP Compiler

Fig. 6 shows the proposed compilation flow for the LSRDP. The most important functionality of the compilation flow is to generating configuration bit-stream and an executable code for the reconfigurable processor including GPP and LSRDP. In the first stage a hw/sw partitioning is performed on the input application manually (like an approach introduced in the previous section) or by means of an automatic tool. Considering the LSRDP architectural specifications, DFGs are mapped on the LSRDP through placing DFG nodes on the PEs, routing interconnection as well as positioning input/output nodes on the proper positions. Configuration bit-stream corresponding to each one of DFGs can be generated after completion of the mapping stage. An executable code including non-critical segments of the application code and a piece of code for LSRDP interfacing has to be generated.

Mapping tool is developed as a part of

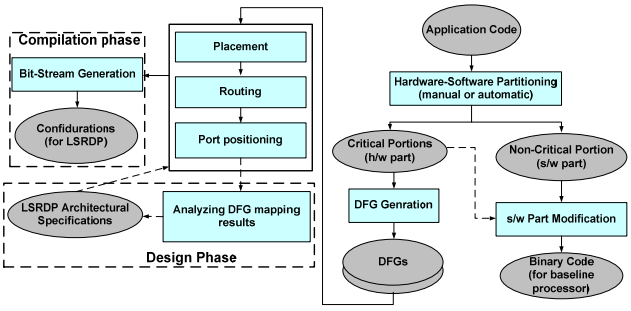

Fig.6: Outline of compiler flow

LSRDP compiler. In addition, it can be used for analyzing the mapping results as well as generating statistics required during the design procedure which all will help the designer to make a decision on the LSRDP architectural specifications

3.4. Architecture design

3.4.1. DFG classification

DFGs obtained though DFG extraction phase have different qualities with respect to their size, no of inputs and outputs and etc. Smaller implementations of a DFG should be tried while a DFG violates constraints. According to this classification four classes including Small (S), Medium (M), Large (L) and XLarge (XL) are constructed using the threshold numbers of FUs, inputs and outputs. For each group, following architecture design procedure is processed and final specifications are obtained.

3.4.2. Configuration of each processing element

Each Processing Element (PE) consists of FU and Transfer Unit (TU) which transfers data from previous to following row by skipping current row. Their different types of PE structure can be considered.

Type I: FU, TU, or FU + TU
Type II: FU, TU, FU + TU, or TU + TU
Type III: FU or TU.

In the design procedure, a suitable type is chosen after analyzing the mapping results.

3.4.3. Placement and routing

In order to minimize the LSRDP size in the mapping procedure, two different criteria were considered within the placement process: optimizing the total number of resources or the maximum connection length to minimize the ORN size. Connection length is defined as the horizontal distance between two PEs, which are located in consecutive rows and have data dependency as shown in Fig.7.

3.4.4. Port positioning

In the LSRDP it is assumed that I/O ports are located in the top and bottom borders. Between I/O ports and PEs in the first/last rows ORNs are located to make their connections possible. In this port positioning step, the main objective is to reducing the connection legnth of the ports and PEs.

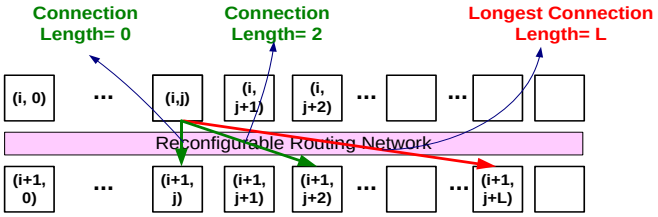

Fig.7: Definition of connection length

3.4.5. Connection length minimization

Since a considerable number of ORN should be used in the RDP architecture, reducing ORN size is an important challenge in the LSRDP design procedure. In order to optimize ORN size, mainly following three minimizing techniques are used: 1) leaving some unoccupied PEs during the placement process, 2) reducing maximum permissible width for placing DFG nodes and 3) some node is ripped-up and re-routing relative nodes. During the mapping process those methods should be attempted and the best result is chosen.

4. Results and Discussion

From the preliminary placement and routine mapping results, we select type II as the structure of PEs: FU, TU, FU+TU, or TU+TU. This type is most flexible one for routing. Following results are obtained under the type II configuration of PE.

Table 2 shows the results obtained through the design procedure as well. Numbers of FUs, Inputs, and Outputs represent the maximum size of the benchmark DFGs for each class. Maximum and average values of width and height are reported in the following columns. Column 7th denotes the total number of PEs required for implementing applications. The last column shows the maximum and average number of required TUs as well.

Table 3 shows final results of optimizing maximum connection lengths. From those results, maximum values are almost two times

Table 2: Results of LSRDP design procedure

	# of FUs	# of Inputs	# of Outputs	Width (max/avg)	Height (max/avg)	Total # of PEs (max/avg)	Extra TUs (max/avg)
RDP-S	128	19	12	26/14.9	10/6.7	98/51.7	56/23.5
RDP-M	512	19	12	26/17.1	16/9.3	170/77.8	92/37.1
RDP-L	1024	38	24	58/40	24/14.4	730/260.1	428/141.3
RDP-XL	> 1024	64	52	122/45.3	25/12.4	1217/350.4	1065/240

Table 3: Final results of Maximum Connection Length

	MCL max/avg
RDP-S	9/4
RDP-M	9/5
RDP-L	19/9.3

larger than the average values. Connection lengths distribution analysis shows 79% of connections are less than 2 and 89% of connections are less than the average value (4). Hence, only a small fraction of connection results in a larger maximum connection length.

5. Conclusions

In this paper, Large-Scale Reconfigurable Data-Path (LSRDP) micro-architecture design procedure and their results are shown. First, 24 benchmark Data Flow Graphs (DFGs) are manually extracted from time evolution heat and vibration equation, Poisson equation, as well as electron repulsion integral calculations. From the analyses of mapping results of benchmark DFGs, LSRDP height, width, and maximum connection length between floating point units are obtained.

Acknowledgments

This research was supported in part by Core Research for Evolutional Science and Technology (CREST) of Japan Science and Technology Corporation (JST).

References

[1] TOP500 Supercomputer, http://www.top500.org/.
[2] ClearSpeed Processor, http://www.clearspeed.com/.
[3] Cell Broadband Engine, http://cell.scei.co.jp/index_j.html.
[4] J. Makino, K. Hiraki and M. Inaba, GRAPE-DR: 2-Pflops massively-parallel computer with 512-core, 512-Gflops processor chips for scientific computing, SC07 2007.
[5] J.D. Owens, D. Luebke, N. Govindaraju, M. Harris, J. Krger, A.E. Lefohn, and T. Purcell, A Survey of General-Purpose Computation on Graphics Hardware, Computer Graphics Forum, 26(1), pp. 80-113, March 2007.
[6] H. Kataoka, H. Honda, F. Mehdipour, Koji Inoue, and Kazuaki Murakami, Performance Evaluation of a Large-Scale Reconfigurable Data-Path Utilized for Scientific Application, IPSJ SIG Technical Reports, VLD2007-133, CPSY2007-76, RECONF 2007-79, pp.1-6, October 2008 (in Japanese).
[7] K. Shimasaki, T. Nagano, H. Honda, F. Mehdipour, K. Inoue, and K. Murakami, On-chip Network Architecture for Large Scale Reconfigurable Data-Path, IPSJ SIG Technical Reports, 2007-ARC-173, pp. 115-120, June 2007 (in Japanese).
[8] H. Honda, *et al.*, Large-Scale Reconfigurable Data-Path Processor using Single Flux Quantum circuit, ICCMSE, 2007. 09.
[9] Japan Science and Technology Agency's CREST program, "Low-power, high-performance, reconfigurable processor using single flux quantum circuits".
[10] W.H. Press, B.P. Flannery, S.A. Teukolsky, and T.W. Vetterling, Numerical Recipes in C, Cambridge University Press, 1988.
[11] S. Obara and A. Saika, Efficient recursive computation of molecular integrals over Cartesian Gaussian Functions, J. Chem. Phys., Vol.84, pp.3963, 1986.