一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
VLD2018-90,CPSY2018-100,RECONF2018-64(2019-01)

FPGA 上での部分再構成を使用したストリーム向けクロスバの実装と検証

川俣　裕一†　　木田　智大†　　柴田裕一郎†　　佐野健太郎††

† 長崎大学大学院工学研究科　〒852–8521 長崎市文教町 1–14
†† 理化学研究所 計算科学研究センター　〒650–0047 神戸市中央区港島南町 7–1–26
E-mail: †{kawamata,kida}@pca.cis.nagasaki-u.ac.jp, ††shibata@cis.nagasaki-u.ac.jp, †††kentaro.sano@riken.jp

あらまし　本論文では, 複数の FPGA を用いるマルチ FPGA クラスタ計算システムにおいて, 部分再構成を用いたネットワーククロスバの実装を提案する. 部分再構成によりクロスバモジュールを再構成することで, ネットワークルーティングを変更できる. 本稿の目的は通常回路と部分再構成回路の比較を行い, 部分再構成を使用したクロスバの資源使用量, 最大動作周波数を明らかにし, 検証, 評価を行い, 今後の課題について明らかにすることである. 結果として, 部分再構成を使用することで, 通常回路より ALM 資源の使用量を減らすことができた. 一方, 最大動作周波数はクロスバの入出力 bit 数が低い場合通常回路より高くなるが, 入出力 bit 数が高くなるにつれ低下し, 4096bit の時では通常回路と同程度となった. 現在は部分再構成に JTAG インタフェースを使用しているため, 今後内部メモリからの高速な部分再構成について検討し, システムに最適なクロスバ設計を検討していく必要がある.

キーワード　FPGA, Arria10, 部分再構成, Partial Reconfiguration,

Yuichi KAWAMATA†, Tomohiro KIDA†, Yuichiro SHIBATA†, and Kentaro SANO††

† Department of Computer and Information Sciences,
Graduate School of Engineering, Nagasaki University　1-14 Bunkyo-machi, Nagasaki, 852–8521 Japan
†† RIKEN Center for Computational Science　7–1–26 Minatojimaminamimachi, Kobe Chuo-ku, 650–0047 Japan
E-mail: †{kawamata,kida}@pca.cis.nagasaki-u.ac.jp, ††shibata@cis.nagasaki-u.ac.jp, †††kentaro.sano@riken.jp

1. 緒　　論

FPGA (Field Programmable Gate Array) は, アプリケーションに応じて最適なデータパスを構成できることから, 電力効率の良い専用計算機のプラットフォームとして従来から注目されてきた. 特に近年では集積度の向上に伴い, 浮動小数点数演算コアを搭載したり, 動作周波数を向上させるための配線アーキテクチャの工夫などが施された FPGA も登場しており, 高性能計算分野における応用にも期待が高まっている [1] [2].

特に, 格子状に配列されたデータに対して同じ形状のデータ参照を伴う演算処理を繰り返し適用するステンシル計算は, 様々な科学技術計算で利用されるデザインパターンの一種であるとともに, FPGA ベースのシステムにおいてストリーム的な枠組みで効率的に処理できることが知られている [3] [4] [5]. また, 演算パイプラインを直列に接続し, メモリアクセス 1 回あたりの演算回数を増やすことで, DRAM への要求メモリバンド幅を増やすことなく演算性能を高めることができる [6]. したがって, 複数の FPGA を相互接続した並列システムを構築した際にも, FPGA 間の接続バンド幅に制約されずに性能をスケールさせることができることから有望である [7].

FPGA を複数台用いたマルチ FPGA 計算システム構築するためには, FPGA 間で相互にデータを交換するための通信機構が必要となる. 多くのシステムでは, チップ内部のモジュール間通信に用いられるネットワークオンチップ (NoC) 用のスイッチ機構を拡張し, FPGA 間接続応用する研究が盛んに行われている [8] [9] [10] [11] [12] [13]. これらの NoC 由来のスイッチ機構は, パケットベースのルーティングを行うものが主流であり柔軟性に優れている. 一方で, ストリームベースの処理を行うマルチ FPGA クラスタ計算システムにおいては, アプリケーションによっては必ずしも図 1 の左図のように入力データを短時間のうちに異なる宛先にルーティングする必要はなく, 右図のようにある程度まとまったデータストリームとしてのルーティングをサポートすればよい. また, パケットごとに任意の宛先にルーティング可能な汎用クロスバはマルチプレクサの集合体として FPGA に実装されるため, 入出力数の増加とともに, 資源使用量, 周波数の面でも効率が低下すると考えられる. しかし, アプリケーションごとに固定のルーティングしか行えないということは柔軟性に欠ける. すなわち, マルチ FPGA システムにおけるク

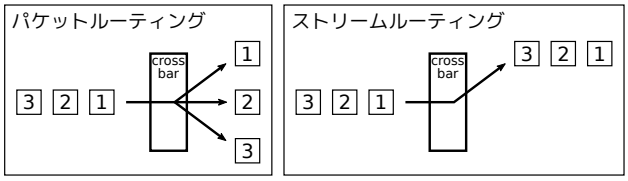

図 1　ルーティングの比較

ロスバには柔軟性と性能・効率の間にトレードオフ関係が存在し, 様々な設計上の選択肢が存在する.

そこで本論文では, 経路切り替えに FPGA の部分再構成技術を利用したストリームベースのネットワーククロスバを提案する. 部分再構成技術は FPGA の特定の回路を他の回路を動作させた状態で変更する技術である. 部分再構成を利用したクロスバは, 通常の汎用クロスバと比べて動的な柔軟性は制約されるものの, 資源使用量の減少及び最大動作周波数の向上が期待できる. これらのトレードオフ関係を明らかにし, 部分再構成をクロスバに適用する効果を評価するために, 通常のクロスバ回路と資源使用量, 最大動作周波数について比較するとともに, 部分再構成に要する再構成時間について評価する. 本論文では図 2 のような 2 次元トーラスによるマルチ FPGA システムを想定し, 図 3 のような外部 4 入出力＋内部 1 入出力の計 5 入力 5 出力を備えたクロスバについて評価, 検証を行う.

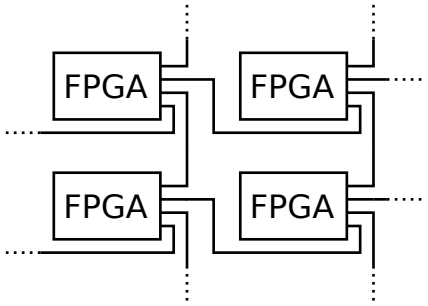

図 2　2 次元トーラス接続された FPGA

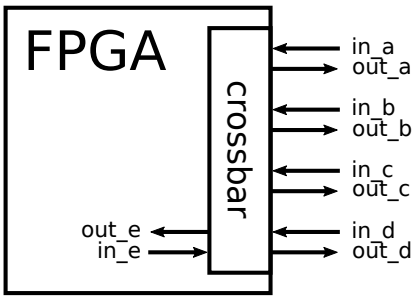

図 3　ストリーム接続例

本論文の構成は以下のとおりである. 2 節で Intel 環境下での FPGA の部分再構成技術について説明するとともに部分再構成デザインの設計手順について述べる. 3 節では本論文で検証した FPGA デザインの設計について説明する. 4 節で作成した FPGA デザインの評価及び考察を行い, 5 節において結論を示す.

2. Intel 環境下での部分再構成について

本研究では Intel 製 FPGA Arria10 で部分再構成 (Partial Reconfiguration: PR) を行う. 実験では構成用データ及び部分再構成用データの転送には JTAG を使用する.

2.1 部分再構成の概要

何度も回路を再構成出来る FPGA 上の回路において, 特定の領域の回路のみを, 他の回路を動作させたまま再構成させることを動的部分再構成という. 本研究では開発環境として Intel 社の Quartus Prime Version 18.1 Pro Edition を使用する.

2.2 部分再構成デザインの設計手順

Intel 社の FPGA における部分再構成回路の設計手順は以下の通りである [14].

1. 回路設計, 記述
2. Design Partition, LogicLock 領域の作成
3. Placement 領域, Routing 領域の確保
4. PR 制御 IP コアの追加
5. リビジョンの作成
6. Base リビジョンのコンパイル
7. qdb ファイルの書き出し
8. 各ペルソナのコンパイル

以降, 上記の各手順について述べていく.

2.2.1 回路設計, 記述

ベースとなる回路を設計, 記述する.

2.2.2 Design Partition, LogicLock 領域の作成

部分再構成するモジュールから Design Partition を作成する. また, 作成した Design Partition から Quartus の LogicLock 機能 [15] を用いて部分再構成するモジュールの配置を固定する. この固定する領域を LogicLock 領域と呼ぶ. 指定した位置にのみ指定したモジュールが配置配線されるため, 部分再構成モジュールは配置配線の自由度が低くなり, 配置を固定しない場合に比べると最大動作周波数の低下が考えられる. 本論文では LogicLock 領域を作成し, クロスバモジュールを固定したデザインを通常回路及び部分再構成回路と比較する.

2.2.3 Placement 領域, Routing 領域の配置

LogicLock 領域には Placement 領域と Routing 領域があり, Placement 領域, Routing 領域の位置と大きさを固定する. Placement 領域は部分再構成するモジュールを配置する領域であり, Routing 領域は Placement 領域は Placement 領域に接続する経路を配置することができる領域である. Routing 領域は回路によって必要な範囲が変わるが, Placement 領域より小さくすることはできない.

2.2.4 PR 制御 IP コアの追加

PR 制御機構は Quartus で IP コアを作成し使用する. PR 制御 IP は部分再構成を行う際に, FPGA 上に 1 つのみ必要となる [16]. PR 制御 IP のインタフェースを図 4 に示す.

図 4 において, nreset は PR 制御 IP コア用の非同期リセット信号入力である. clk は PR 制御 IP コア用のクロックである. 最大 100MHz まで対応する. pr_start は信号が 0 から 1 に変わっ

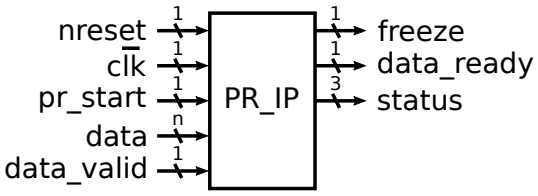

図 4　PR 制御 IP

た時部分再構成イベントを実行する. freeze 信号がロー (0) の時のみ, 次の pr_start 信号を受け取る. data には部分再構成用コンフィグレーションデータの入力を行う. データ幅は 1, 8, 16, 32bit が選択できる. data_valid は data ポートに有効なデータが入力されていることを示す. freeze は部分再構成実行中にハイ (1) の信号を出力する. data_ready は data ポートが data を受け取る準備ができていることを示す. status は部分再構成イベントの状態を示す 3bit のエラー出力である. なお, clk, pr_start, data, data_valid, および data_ready は JTAG インタフェース経由で部分再構成をする場合, PR 制御 IP が JTAG インタフェースと信号のやり取りを行うためこれらへの値の挿入は無視される.

2.2.5　リビジョンの作成

部分再構成設計フローにおいて Quartus ではプロジェクトリビジョン形式を用いる. Base リビジョンと Persona Implementation リビジョンの 2 つがある. Base リビジョンでは回路全体の設計を行い, Persona Implementation リビジョンでは部分再構成するモジュールについての設計を行う. 通常 Base リビジョンは 1 つであり, 複数の Persona Implementation リビジョンが存在する.

2.2.6　Base リビジョンのコンパイル

先に Base リビジョンのコンパイルを行う. このコンパイル操作には論理合成, 配置配線, タイミング確認, コンフィグレーションデータ生成などが含まれる.

2.2.7　qdb ファイルの書き出し

コンパイルした Base リビジョンを database ファイルとして出力する.

2.2.8　各ペルソナのコンパイル

部分再構成する機能モジュールをペルソナと呼ぶ. 先項で作成した qdb ファイルを使用し, 部分再構成用データを生成する.

3.　実装デザイン設計

本論文では図 5 のように通常のデザインの他にクロスバモジュールの位置を固定した Logic Lock (以降 LL) 回路とクロスバモジュールを部分再構成する Partial Reconfiguration (以降 PR) 回路を作成した.

今回作成した FPGA デザインでは, M20K Embedded Memory である SrcRAM からストリームデータを生成し, クロスバを通し DstRAM にストリームデータを格納する. また、制御用の CtrRAM からクロスバの制御を行う.

PR 回路以外のクロスバモジュールはフルクロスバを実装した. また、ストリームデータの bit 幅は 8bit, 32bit, 64bit, 256bit, 1024bit, 4096bit であり, クロスバは 5 入力 5 出力である.

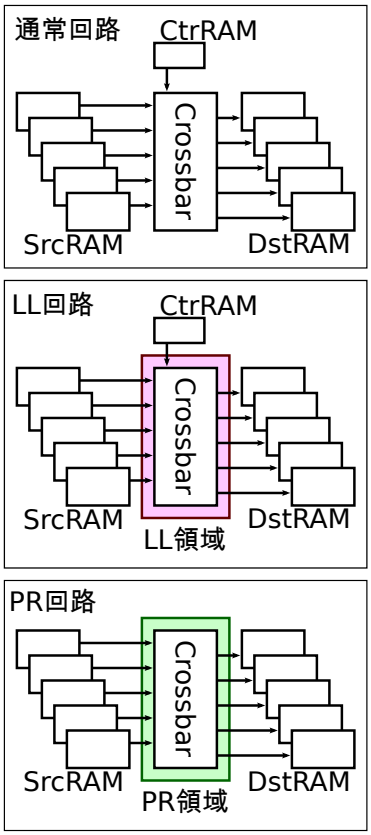

図 5　各回路の概略比較

3.1　フルクロスバの実装

フルクロスバの実装は図 6 のとおりである. 5 入力 5 出力で

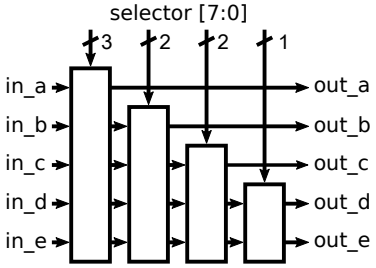

図 6　フルクロスバの実装

あり, 8bit の選択線によって入力と出力の接続を変更する. 上位 3bit が最も左のクロスバから 1 出力を選択肢, それ以外の出力は右隣のクロスバに順番を変更せずに接続される. このように 8bit の接続線を 3bit, 2bit, 2bit, 1bit に分けて各クロスバの選択線として用いる. この構造において, 出力が重複することはない.

3.2　ペルソナ

本論文では図 7 に示す 3 つのペルソナ (ST, RT, X) を作成した. PR クロスバ回路ではペルソナを交換することでルーティングを変更するため, クロスバ制御用の選択線は必要としない. 従って, PR 回路では選択線は実装しない.

ペルソナ ST では入力と出力が変更なしに接続される. 全ての FPGA においてこのペルソナを適用することは, 各 FPGA は独立することとなる. ペルソナ RT では最下を除き入力は 1 つ

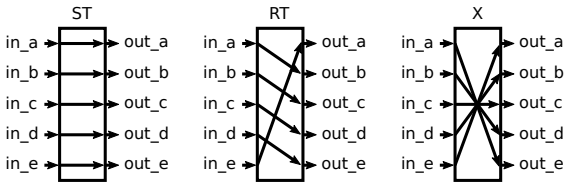

図 7　ペルソナの比較

下の出力に接続される. 最下の入力は最上の出力に接続される. 図 2 で考えると図 8 のようなシステムを構成することができる. ペルソナ X では入出力の上下が入れ替わって接続される.

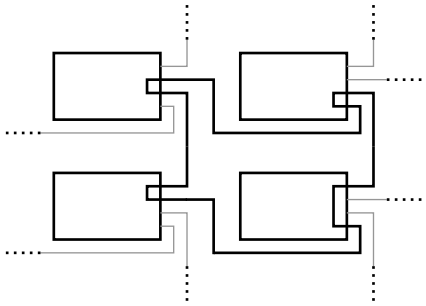

図 8　ペルソナ RT のルーティング例

3.3　Embedded Memory

本論文では RAM として M20K を使用した. M20K をシングルポート RAM として使用することで, In-System Memory Content Editor を使用でき, RAM の読み書きが行える. 本論文では In-System Memory Content Editor を使用して, SrcRAM にデータを書き込み DstRAM のデータを読み込むことで回路動作の確認を行った. また, 全ての入出力 bit 数においてワード数は 8 とした.

3.4　再構成等領域

本論文ではデザインの対照性を考慮し, 同入出力 bit 数の LL 回路と PR 回路では同じ領域位置に実装し, 同じ領域サイズを設定した. ロジックロック領域及び部分再構成領域については表 1 に示す. 64 bit までの回路については領域サイズを 10×10 とした. 256 bit, 1024 bit においては, 10×10 の領域内に配置配線することができなかったため, Y 方向に領域拡張した. 4096 bit においては X, Y 方向に領域を拡張し, 領域を設定できるように基準点 (Origin) の位置も変更した.

表 1　各領域比較

	Width	Height	Origin
8bit LL 回路	10	10	X88_Y8
8bit PR 回路	10	10	X88_Y8
32bit LL 回路	10	10	X88_Y8
32bit PR 回路	10	10	X88_Y8
64bit LL 回路	10	10	X88_Y8
64bit PR 回路	10	10	X88_Y8
256bit LL 回路	10	30	X88_Y8
256bit PR 回路	10	30	X88_Y8
1024bit LL 回路	10	110	X88_Y8
1024bit PR 回路	10	110	X88_Y8
4096bit LL 回路	28	210	X35_Y11
4096bit PR 回路	28	210	X35_Y11

4.　評価と考察

今回作成した回路の評価と考察を行う. 1024 bit RT, 4096 bit RT および 4096 bit X は配置配線を行うことができず実装できなかった. 評価環境を以下に示す.

- FPGA : Intel Arria10 115N2F45E1SG
- CPU : Intel Core(TM) i7-8700K
- MEM : DDR4 16GB
- OS : CentOS 7.5

4.1　最大動作周波数

今回作成した回路の最大動作周波数を図 9 および表 2 に示す.

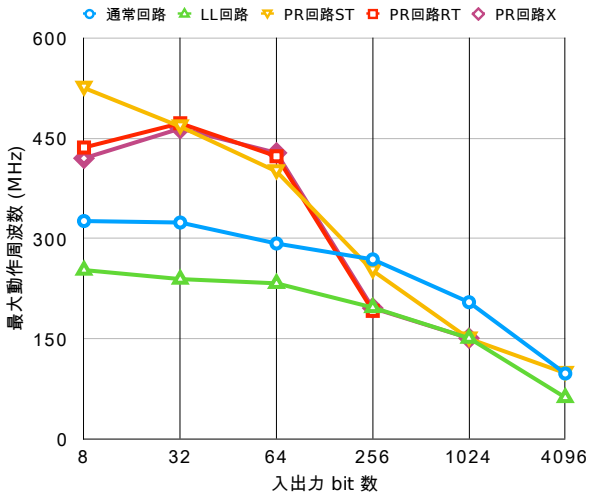

図 9　最大動作周波数の比較

表 2　最大動作周波数 (MHz)

入出力 bit 数	8	32	64	256	1024	4096
通常回路	326.26	324.04	292.65	268.89	204.79	98.12
LL 回路	252.91	239.41	233.05	196.89	151.01	62.25
PR 回路 ST	525.49	467.63	400.48	251.95	150.26	98.73
PR 回路 RT	436.30	472.59	423.19	191.86	-	-
PR 回路 X	420.17	464.68	428.27	196.00	150.99	-

64 bit 以下では PR で作成したクロスバの方が通常回路より最大動作周波数が高い結果となった. 一方, 256 bit 以上では通常回路を下回る結果となった. また LL 回路は全ての bit で通常回路を下回った. PR 回路ペルソナの違いにより最大動作周波数は異なり, bit 数が異なると最大動作周波数の順位は異なる結果となった.

これは PR 回路を作成する際にクロスバモジュールの配置を固定する必要があるため, PR 領域が十分に広く自由度が高い 64 bit までは最大動作周波数が上がり, それ以降は配線の自由度が下がるため最大動作周波数も低下したためと考えられる. また, 本論文ではデータストリームが RAM から RAM と配置でき

る箇所が決まっているため, RAM を用いないデザインでは配置配線の関係で更に高い最大動作周波数が期待できる.

4.2 使用資源量

今回作成したクロスバモジュールの ALM 使用量の比較を図 10 および表 3 に示す.

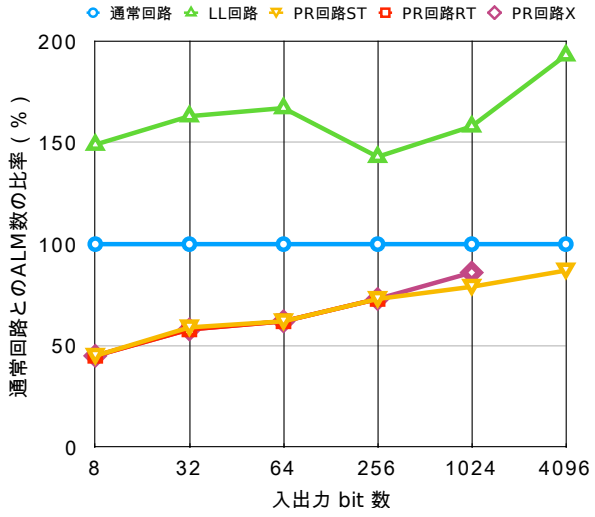

図 10　ALM 数の比較

表 3　クロスバモジュール資源使用量 (ALMs)

入出力 bit 数	8	32	64	256	1024	4096
通常回路	101	276	514	1911	6511	26686
LL 回路	150	450	858	2729	10300	51421
PR 回路 ST	45	164	321	1386	5122	23168
PR 回路 RT	45	160	321	1386	-	-
PR 回路 X	45	160	320	1386	5605	-

図 10 は通常回路の ALM 使用量を 100% とした時の比率をグラフ化している. 全ての bit 数で PR 回路は通常回路を下回り, LL 回路は通常回路を上回る結果となった. クロスバモジュールの ALM 使用量において PR 回路は入出力が 8bit の時で通常回路の約 45% であり, 比率は入出力 bit 数が増加するにつれて高くなり, 4096bit の時には約 87% となった. PR 回路ペルソナの違いにおいて ALM の使用量が異なる場合があったが, 基本的に PR 回路は入出力の違いしかないため, 大きな差が発生しなかったものと考えられる.

また, 今回作成した FPGA デザイン全体の使用資源量を表 4 に示す. PR 回路には通常回路に加えて PR 制御用 IP 回路が加わっているにも関わらず, PR 回路が全ての資源使用量で通常回路を下回っており ALM の使用量は入出力が 32bit の時, および 4096bit の時で通常回路の約 87% となった.

表 4　回路全体の資源使用量

	ALM	Register	RAM
32bit 通常回路	1138	1138	21
32bit LL 回路	1307	1149	21
32bit PR 回路 ST	988	1109	20
4096bit 通常回路	47950	42961	2051
4096bit LL 回路	73038	42361	2051
4096bit PR 回路 ST	41706	42912	2050

4.3 生成ファイルサイズ

今回作成したビットストリームファイルサイズの比較を表 5 に示す. sof ファイルは通常のコンフィグレーションの用い, rbf ファイルは部分再構成を行う際に用いる. そのため通常回路及び, LL 回路では rbf ファイルは生成されない.

表 5　ファイルサイズ (MB)

	sof	rbf : ST	rbf : RT	rbf : X
8bit 通常回路	36	-	-	-
8bit LL 回路	36	-	-	-
8bit PR 回路	36	5.9	5.9	5.9
32bit 通常回路	36	-	-	-
32bit LL 回路	36	-	-	-
32bit PR 回路	36	6.1	6.1	6.1
64bit 通常回路	36	-	-	-
64bit LL 回路	36	-	-	-
64bit PR 回路	36	6.1	6.1	6.1
256bit 通常回路	36	-	-	-
256bit LL 回路	36	-	-	-
256bit PR 回路	36	17	16	17
1024bit 通常回路	36	-	-	-
1024bit LL 回路	36	-	-	-
1024bit PR 回路	36	54	-	54
4096bit 通常回路	36	-	-	-
4096bit LL 回路	36	-	-	-
4096bit PR 回路	36	114	-	-

PR 領域が大きい 256bit 以降のデザインでは 64bit までのデザインより rbf ファイルサイズが大きい結果となった. 同じ領域でも 8 bit より 32 bit, 64 bit の方がファイルサイズが大きい結果となった. また, 同じ入出力 bit 数のデザインでもペルソナによって生成される rbf ファイルサイズに差異が見られた. どのデザインも sof ファイルは同じ大きさだった.

このことから, 広い領域を必要とする高 bit 入出力のクロスバにおいては生成される部分再構成用ビットストリームデータが大きくなるため, FPGA 内部の高速なメモリにデータを格納することが難しく, またそのデータを格納するために FPGA 資源を多量に消費する事が考えられるため, 本研究で作成するシステムへの実装には不向きであると判断できる.

4.4 再構成時間

再構成時間の比較を表 6 に示す. 表の数値は計測 5 回の平均値である. コンフィグレーション及び部分再構成は JTAG インタフェースを用いている. また時間計測には回路の変更だけでなく, Quartus 起動等のオーバーヘッドを含んでいる.

コンフィグレーションは sof ファイルサイズに差異が見られなかったことと同様に, 大きな差は見られなかった. 一方, 部分再構成においてはファイルサイズが大きい 256bit 以降で再構成時間の増加が見られ, 入出力が 64bit の時, 部分再構成時間は通

表 6　再構成時間 (秒)

	Configration	PR : ST	PR : RT	PR : X
8bit 通常回路	21.24	-	-	-
8bit LL 回路	21.02	-	-	-
8bit PR 回路	21.12	7.68	7.69	7.62
32bit 通常回路	20.99	-	-	-
32bit LL 回路	21.08	-	-	-
32bit PR 回路	21.12	7.93	8.23	7.70
64bit 通常回路	21.05	-	-	-
64bit LL 回路	21.21	-	-	-
64bit PR 回路	21.30	7.95	7.86	7.84
256bit 通常回路	21.18	-	-	-
256bit LL 回路	21.28	-	-	-
256bit PR 回路	21.26	18.20	19.00	19.06
1024bit 通常回路	21.25	-	-	-
1024bit LL 回路	21.28	-	-	-
1024bit PR 回路	21.37	56.65	-	56.46
4096bit 通常回路	21.36	-	-	-
4096bit LL 回路	21.36	-	-	-
4096bit PR 回路	21.36	163.34	-	-

常のコンフィグレーション時間の約 37% であり, 86%, 265% と増加していき 4096bit の時には約 765% となった.

今回は JTAG インタフェースを使用したが, 内部のメモリを使用することでより高速な部分再構成が可能であり, PR 制御 IP の理論最大性能である 3.2Gbps で部分再構成を行った場合, 5.9MB の 8bit クロスバモジュールペルソナを変更するのに要する時間は約 15 ミリ秒であると見積もることができる. また, 今回作成した最大のファイルサイズであった 4096bit クロスバペルソナは 115MB であり, 理論上約 285 ミリ秒で部分再構成可能であるが, この数値が本研究のシステムにとって妥当な数値かは今後検証していく必要がある.

5. 結　　論

本論文では, 部分再構成を用いたストリームデータ向けクロスバについて述べた. 通常のフルクロスバと比べて 32 bit, 4096 bit で 13% デザイン全体の ALM 使用数を低くすることができた. 最大動作周波数については, 部分再構成領域が十分に広く配置配線の自由度が高いデータ幅 64 bit までは通常回路より最大動作周波数が高く, 8 bit 時には最大で 1.6 倍となった. しかし, データ幅が 256 bit 以降になると最大動作周波数は通常回路を下回ることが多かった. また部分再構成時間は JTAG インタフェースを用いて行った結果, 最短でも約 8 秒であり今後内部メモリからの高速な部分再構成について検証を行っていく必要がある.

文　　献

[1] Kentaro Sano and Satoru Yamamoto, “FPGA-Based Scalable and Power-Efficient Fluid Simulation using Floating-Point DSP Blocks,” IEEE Transactions on Parallel and Distributed Systems, vol.28, pp.2823–2837, 2017.

[2] Czajkowski, Tomasz and Aydonat, Utku and Denisenko, Dmitry and Freeman, John and Kinsner, Michael and Neto, David and Wong, Jason and Yiannacouras, Peter and P. Singh, Deshanand, “From OpenCL to high-performance hardware on FPGAs,” Proceedings - 22nd International Conference on Field Programmable Logic and Applications, FPL 2012, pp.531–534, 08 2012.

[3] K. Sano, “FPGA-Based Systolic Computational-Memory Array for Scalable Stencil Computations,” High-Performance Computing Using FPGAs, pp.279–303, 2013.

[4] Y. Sato, Y. Inoguchi, W. Luk, and T. Nakamura, “Evaluating reconfigurable dataflow computing using the Himeno benchmark,” Proc. ReConFig, pp.1–7, 2012.

[5] H. Giefers, C. Plessl, and J. Förstner, “Accelerating Finite Difference Time Domain Simulations with Reconfigurable Dataflow Computers,” Proc. HEART, pp.33–38, 2013.

[6] K. Dohi, K. Okina, R. Soejima, Y. Shibata, and K. Oguri, “Performance modeling of stencil computing on a stream-based FPGA accelerator for efficient d esign space exploration,” IEICE Transactions on Information and Systems, vol.98–D, no.2, pp.298–308, 2015.

[7] K. Sano, Y. Hatsuda, and S. Yamamoto, “Multi-fpga accelerator for scalable stencil computation with constant memory bandwidth,” Parallel and Distributed Systems, IEEE Transactions on, vol.25, pp.695–705, 03 2014.

[8] Sagar Latti, “FPGA Implementation of Four Port Router for Network on Chip,” International Research Journal of Engineering and Technology (IRJET), vol.03, pp.887–880, 2016.

[9] Andreas Ehliar, Dake Liu, A Network on Chip based gigabit Ethernet router implemented on an FPGA, vol.03, SSoCC, 2006.

[10] Andreas Ehliar, Dake Liu, “An FPGA based open source Network-on-Chip architecture,” FPL, vol.03, pp.800–803, 2007.

[11] 史 発, Thiem Van Chu, 吉瀬 謙二，“FPGA アクセラレータを用いた大規模 NoC におけるルーティング手法の検討，” 情報処理学会第 79 回全国大会，pp.137–138，2017.

[12] Roman Gindin, Israel Cidon, Idit Keidar, “NoC-based FPGA: Architecture and routing,” NOCS, pp.253–264, 2007.

[13] David Bafumba-Lokilo, Yvon Savaria, Jean-Pierre David, “Generic Crossbar Network on Chip for FPGA MPSoCs,” IEEE, pp.269–272, 2008.

[14] Intel, AN 797: Partially Reconfiguring a Design on Intel Arria 10 GX FPGA Development Board, Intel, 2018.

[15] Altera, 10. デザイン・フロアプランの解析および最適化，Altera，2007.

[16] Intel, Intel Quartus Prime Pro Edition User Guide Partial Reconfiguration, Intel, 2018.