

Elastic CGRA における EoP 信号とデータのための同期プリミティブの提案

田畑 順正[†] Boma Adhi^{††} Omkar Bhilare^{†††} Hamas Waqar^{†††} Omar Ragheb^{††††}

佐野 健太郎^{††} Jason Anderson^{†††} 上野 知洋^{††} 長名 保範[†]

[†] 熊本大学 〒860-8555 熊本県熊本市中央区黒髪 2-2-1

^{††} 理化学研究所 〒650-0047 兵庫県神戸市中央区港島南町 7-1-26

^{†††} University of Toronto 10 King's College Road, Toronto, ON M5S 3G4, Canada

^{††††} Fujitsu Consulting (Canada) Inc. 1600-155 University Ave. Toronto, ON M5H 3B7, Canada

E-mail: junsei@st.cs.kumamoto-u.ac.jp, osana@kumamoto-u.ac.jp

あらまし Elastic CGRA は、データ到着に応じて計算と通信を進めることで、可変遅延を含む処理に対して高い柔軟性を有する。一方で、圧縮データストリームのようにパケットの境界情報を伴うデータを扱う場合には、データ本体だけでなく End-of-Packet (EoP) 信号も対応するデータと整合を保ったまま伝搬させる必要がある。そこで本稿では、Elastic CGRA において EoP 信号とデータを同期させるプリミティブ Special Join を提案する。Special Join は、入力側に Join と出力側に Eager Fork を組み合わせることで、EoP とデータの一対一対応の保持、先着入力に対するバックプレッシャによる保持、下流ストール時の整合維持、既存のエラスティックなハンドシェイクとの整合を実現する。さらに、Elastic RIKEN CGRA を対象として CGRA-ME 上に Special Join を実装し、マッパーが複数出力、複数ビット幅のプリミティブを扱えるようにした。機能検証の結果、Special Join は EoP と対応するデータの整合性を保持したまま、後段へ正しく受け渡せることを確認した。

キーワード CGRA, CGRA-ME, データ圧縮, データストリーム, EoP

1. はじめに

計算機の性能はしばしば外部メモリ帯域に律速される。CGRA は演算器の空間並列性とパイプライン処理により高いスループットを実現できる一方十分なデータ供給が得られない場合にはその性能を十分に活かせない。とりわけ Elastic CGRA は、データの到着に応じて計算と通信が進行するデータフロー型のアーキテクチャであり、ハンドシェイク信号によって可変遅延やバックプレッシャを扱えるため、ストリーム処理との親和性が高い [1][2][3]。

メモリアクセス帯域の不足を緩和する手法として、先行研究では FPGA ベースのストリーム計算を対象に、浮動小数点数値データ列に対するハードウェアベースの帯域圧縮機構を提案している [4]。同手法は予測ベースの損失レス圧縮・伸長に加え、複数チャネルを単一ストリームへ束ねるマルチチャネルの SerDes を備え、実効帯域の拡張によってアプリケーション性能を向上できることを示した。したがって、同様のデータ圧縮機構を CGRA ベースのストリーム実行基盤へ導入することは有望である。

一方で、この種の圧縮データストリームを Elastic CGRA 上で扱うには、データ本体だけでなく、ストリームの区切りを示すパケット境界情報も正しく扱う必要がある。とくに、多チャネル圧縮では各チャネルの圧縮率が一様ではなく、時間的にも変動するため、単純な Time Division Multiplexing では十分では

ない。先行研究では、利用可能な帯域を各チャネルへ動的に割り当てながら圧縮されたデータブロックを直列化・復元する必要があることを示している [4]。このとき、後段で正しく復元するためには、どこまでが 1 つの圧縮ブロックあるいは 1 つのストリーム単位に対応するのかを示す境界情報が必要となる。本稿では、この境界情報のうち終端側を示す EoP に着目する。

そこで本稿では、Elastic CGRA において EoP 信号とデータを同期させるプリミティブ Special Join を提案する。Special Join は CGRA 内部に組み込み可能な同期機構であり、EoP と対応するデータの整合性を保ったまま後段のアプリケーションへ受け渡すことを目的とする。さらに、Elastic RIKEN CGRA を対象として CGRA-ME 上に Special Join を実装し、複数出力、複数 bit 幅、1bit Network に対応させたうえで機能検証を行った。その結果、図 1 の通り 1 bit Network を含む RIKEN CGRA 上に Special Join を実装し EoP とデータの整合性を保ったまま、後段へ正しく受け渡せることを確認した。

本論文の構成は以下のとおりである。第 2 章で Elastic RIKEN CGRA、CGRA-ME、および圧縮データストリームと EoP 同期の課題について述べる。第 3 章で提案する Special Join の目的、構造を示す。第 4 章で CGRA-ME 上での実装について述べ、第 5 章で機能検証結果を示す。最後に、第 6 章でまとめと今後の展望を述べる。

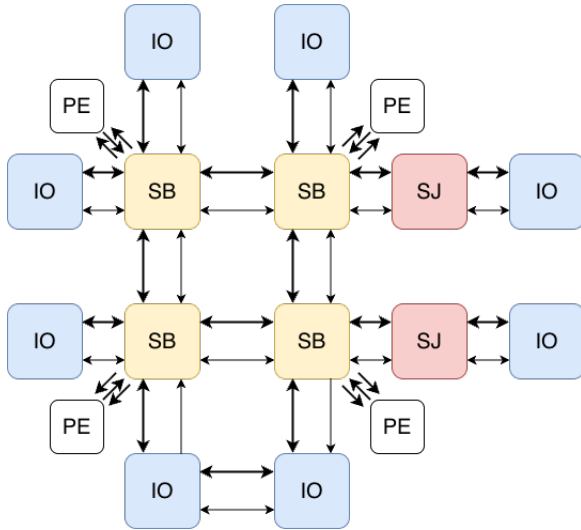


図 1 Special Join を含んだ Elastic RIKEN CGRA

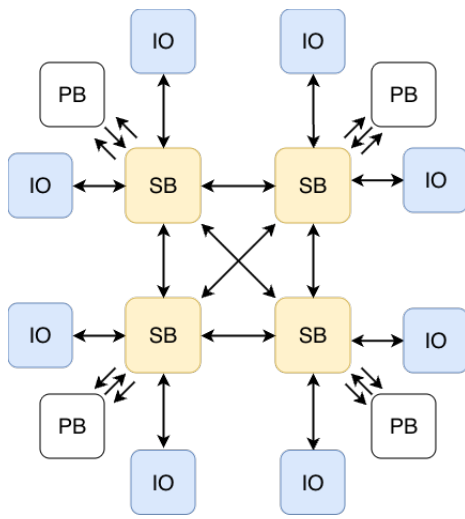


図 2 Elastic RIKEN CGRA

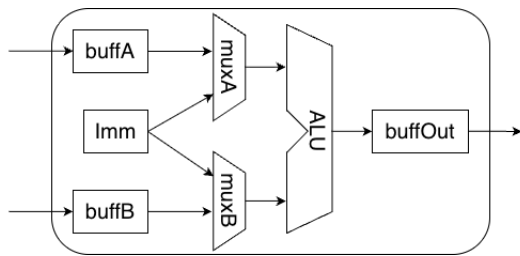


図 3 Elastic RIKEN CGRA の PE

2. 背景

2.1 Elastic RIKEN CGRA

Elastic CGRA は、データフローパラダイムに基づく CGRA であり、データの到着に応じて計算および通信が進行する。送信側はデータと valid を提示し、受信側は受信不能時に stop を返すことでバックプレッシャを与える。このレイテンシ非依存な性質により、外部メモリアクセスや入力依存の演算遅延など、サイクル単位で固定しにくい遅延を含む処理にも対応しやすい [1] [5]。

図 2 に本研究で対象とする Elastic RIKEN CGRA の全体構成を示す。Elastic RIKEN CGRA は、2 次元アレイ状に配置された Processing Element (PE) と、それらを接続する Switch Block (SB) から構成される。各 PE は図 3 に示すように、ALU、入力および出力の Elastic Buffer、ならびに Immediate register を備える。また、各 PE に隣接して SB が配置され、SB は上下左右および対角方向の隣接 SB と接続され、これをクロスバと呼ぶ。アレイ周囲にはデータストリームの入出力を担う I/O ブロックが配置される [6] [7]。

本研究で対象とする実装では、32 bit 幅のクロスバとは別に、1 bit 幅の制御信号を伝搬する経路を 1bit Network と呼ぶ。EoP はこの 1bit Network 上を流れる制御情報として扱う。外部ストリーム側のインタフェースはデータ、valid、stop、EoP を持ち、CGRA 内部ではデータ、valid、stop を基本とする。そのため、外部から与えられる EoP を CGRA 内部で扱うためには、32 bit のクロスバと 1bit Network の間で整合を保つ同期機構が必要となる。

2.2 CGRA-ME

CGRA-ME (CGRA-Modeling and Exploration) は、CGRA アーキテクチャおよびその CAD 手法の研究を目的としたオープンソースのフレームワークである [8]。CGRA-ME 2.0 では、アプリケーション、CGRA アーキテクチャ記述、および制約条件を入力とし、対象アーキテクチャへのマッピングを行う。アプリケーションは計算集約なループやカーネルからデータフローグラフ (DFG) へ変換され、アーキテクチャは CGRA-ME が提供する C++ API を用いて、PE やインタコネクトなどの構成要素を記述する [8]。

CGRA-ME はマッピング結果として対象アーキテクチャの構成ビットストリームを生成するとともに、生成アーキテクチャの Verilog RTL も出力する。これにより、機能シミュレーションや物理実装を含む一連の検証が可能となる。図 4 に本研究で用いる CGRA-ME のツールフローを示す。

本研究では、このフレームワーク上に SpecialJoin を新たなエラスティック・プリミティブとして実装し、生成された RTL とテストベンチを用いて機能検証を行う。

2.3 圧縮データストリームと EoP 同期の課題

外部メモリ帯域はストリーム計算の性能を律速する主要因であり、高い演算並列性を備える CGRA においても、十分なデータ供給が得られなければ性能向上は頭打ちとなる。先行研究では、浮動小数点数値データ列に対する予測ベースの損失レス圧縮器・伸長器を提案するとともに、複数チャネルの圧縮データを単一ストリームへ束ねるマルチチャネル SerDes を導入し、実効帯域の拡張によって FPGA ベースのストリーム計算性能を改善できることを示した [4]。したがって、同様の圧縮データストリームを CGRA ベースの計算基盤へ導入することは、メモリ帯域律速を緩和する有効な方向性である。

重要なのは、マルチチャネル圧縮では各チャネルの圧縮率が一律ではなく、時間的にも変動する点である。したがって、圧縮後のデータは固定長の元データ列としてではなく、可変長の圧縮ブロック列として扱う必要がある。後段で正しく復元する

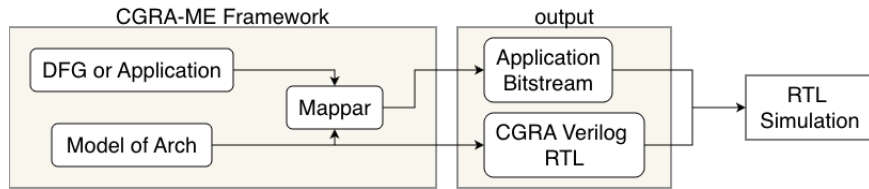


図4 CGRA-ME のツールフロー

ためには、どこまでが同一のデータブロック、あるいは同一のストリーム単位に対応するのかを示すパケットの境界情報が必要となる。

本研究では、この境界情報のうち終端を示す信号として EoP を扱う。EoP は、単純なカウンタだけでは終端を表しにくい場合でも、ストリーム終端を動的に表現できる。また、全データ受信後の追加処理や状態更新の契機としても利用できる。したがって、圧縮データストリームを Elastic CGRA へ導入する際には、データだけでなく、EoP のようなパケット境界情報もあわせて正しく伝搬させる必要がある。

一方で、Elastic CGRA ではデータ転送がハンドシェイクによって制御され、バックプレッシャによりデータの到着タイミングが変動する。したがって、データと EoP を独立に流すと、あるパケットに属するデータとその終端情報の対応関係が崩れ、後段での正しい受け渡しや復元を妨げる可能性がある。以上より、圧縮データストリームを Elastic CGRA 上で扱うためには、パケット境界情報である EoP を対応するデータと同期させる仕組みが必要となる。

3. Special Join の提案

3.1 目的・要求仕様

本研究では、EoP を各データトークンに付随する 1-bit の制御値として扱う。すなわち、データ列を D_0, D_1, \dots 、それぞれに対応する EoP 列を E_0, E_1, \dots としたとき、Special Join は下流へ $(D_0, E_0), (D_1, E_1), \dots$ の順でのみ出力しなければならない。したがって、 $(D_i, E_j) (i \neq j)$ のような不整合な組合せや、データ/EoP いずれかの欠落、重複、順序逆転は許されない。

Special Join の第一の目的は、可変遅延およびバックプレッシャの存在下でもデータと EoP の一対一対応を保持することである。Elastic CGRA では、データと EoP を別々に流すと、片方のトークンのみが先行あるいは滞留し、後段でパケット境界情報が誤って解釈される可能性がある。第二の目的は、一方の入力が先に有効になった場合でも、対応する他方が到着するまで整合を崩さないことである。本実装では、先着トークンを内部レジスタへ格納するのではなく、上流へバックプレッシャを返して送信側に値を保持させる。第三の目的は、同期済みのデータ/EoP 組を下流ストールから保護することである。本モジュールはデータの出力側と EoP の出力側の 2 出力を持ち、それぞれが独立にストールし得るため、一方が先に受理しても他方が受理するまで同一の組を保持できなければならない。第四の目的は、既存のエラスティックな設計を壊さずに CGRA 内へ組み込み可能であることである。

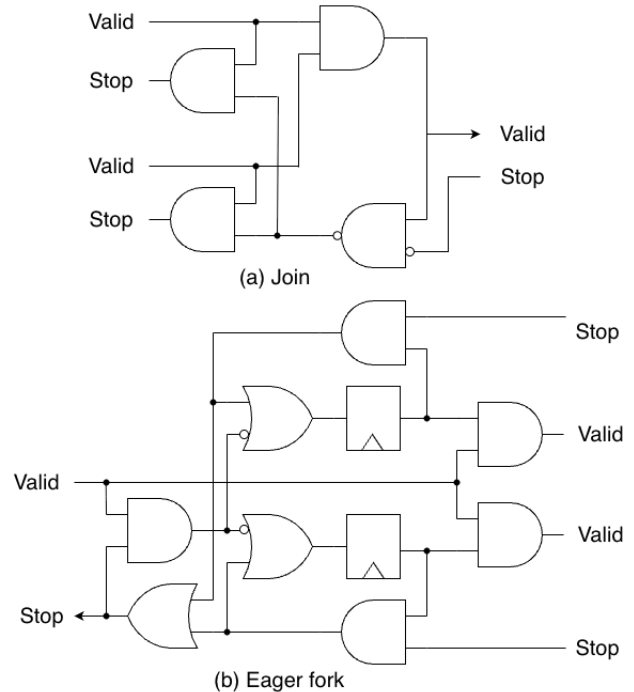


図5 Join と Fork の構造

以上より、Special Join に要求される性質は、(1) データ/EoP 対応の保存、(2) 先着入力に対するバックプレッシャによる保持、(3) 出力側ストール時の整合維持、(4) 既存プリミティブとの整合、の 4 点に整理できる。Special Join は、Join の「複数入力がそろって出力しない」という性質と、Eager Fork の「複数出力の受理状況を個別に記憶しながら整合を保って配布する」という性質を、EoP 同期のために組み合わせたプリミティブと位置付けられる [9]。

3.2 構造

Special Join は、データ入力チャンネルと EoP 入力チャンネルの 2 入力、およびデータ出力チャンネルと EoP 出力チャンネルの 2 出力から構成される。データ入力は 32 bit 幅のデータ値を選び、EoP 入力は 1 bit 幅の制御値を運ぶ。両入力および両出力は、それぞれ独立したチャンネルとしてハンドシェイク信号を備えるが、下流へは常に対応する 1 組のデータ/EoP として提示される。図 5 に、Special Join の構成要素である Join と Eager Fork の構造を示す。図 5(a) の Join は、複数入力がそろった時点で初めて出力を有効化する同期点であり、図 5(b) の Eager Fork は、1 つの入力トークンを複数の出力チャンネルへ配布し、片方の出力が先に受理した場合でも、他方への伝搬が完了するまで状態を保持できる [5]。SpecialJoin は、入力側で Join によりデータと EoP

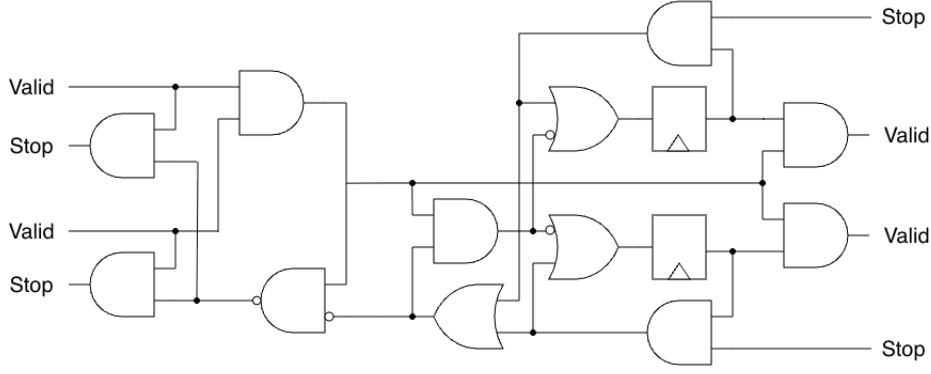


図 6 Special Join の構造

を同期させ、出力側で Eager Fork によりその同期済みトークンをデータ出力チャンネルと EoP 出力チャンネルへ整合性を保ったまま配布する構成である。

図 6 に提案する Special Join の構造を示す。本実装ではデータ値や EoP 値を保持する専用レジスタを内部に持たず、値の経路そのものは入力から出力へ直接接続される。

整合の保持は、valid/stop 制御と、出力側の受理状態を記憶する 2 つの状態ビット q_d, q_e によって実現される。入力データ側および EoP 側の有効化ビットをそれぞれ u_d および u_e とし、入力 valid をそれぞれ v_d および v_e とする。また、データ出力側および EoP 出力側の下流 stop をそれぞれ s_d^{down} および s_e^{down} とする。通常の利用形態では $u_d = u_e = 1$ である。このとき、入力側の Join 条件 J は

$$J = (u_d \vee u_e) \wedge (v_d \vee \neg u_d) \wedge (v_e \vee \neg u_e)$$

で表され、通常の 2 入力動作では

$$J = v_d \wedge v_e$$

となる。

各出力に対するストール条件は

$$t_d = u_d \wedge q_d \wedge s_d^{\text{down}}, \quad t_e = u_e \wedge q_e \wedge s_e^{\text{down}}$$

であり、2 出力全体として前進可能かどうかを表す共通条件 G は

$$G = \neg(t_d \vee t_e)$$

で定義できる。これにより、各出力の valid は

$$v_d^{\text{out}} = u_d \wedge J \wedge q_d, \quad v_e^{\text{out}} = u_e \wedge J \wedge q_e$$

と表される。すなわち、Special Join の論理は、図 5 の Join により入力同期を行い、その後、図 5 の Eager Fork に対応する状態ビット q_d, q_e により 2 出力の受理状態を管理する構成になっている。

3.3 内部 buffer/FIFO の挿入

上述のとおり、本実装の Special Join は、データおよび EoP の値そのものを内部で保持しない。未同期の入力、あるいは一部だけが受理済みの同期済み出力は、valid/stop 制御によって

上流または未受理側に保持させる。この方式は、Join と Eager Fork の論理を比較的素直に実装でき、内部状態も最小限に抑えられる。

ただし、内部に Buffer/FIFO を挿入する設計が不要であることを意味するわけではない。むしろ、上流がリトライ中にペイロードを安定保持できない場合、上流から独立した 1 段の蓄積を持たせたい場合、あるいは EoP 側だけに追加の余裕を持たせたい場合には、内部に Buffer/FIFO を設ける意義がある。とくに、Elastic Buffer はレイテンシ非依存なデータ伝搬とバッファ深さの調整に有効であり、Eager Fork や Join と同様にエラスティック設計の基本要素として位置付けられる [5]。

したがって、Special Join の本質は、Join による入力同期と Eager Fork による 2 出力整合にある。内部に Buffer/FIFO を持つかどうかは、この本質を変えるものではなく、上流インタフェースの保持能力、要求周波数、必要なバッファ深さ、ならびにデータ経路と EoP 経路に求められる余裕に応じた実装上の選択である。入力側に Buffer/FIFO を置く構成は入力保持を Special Join 内部で完結させやすく、出力側に Buffer/FIFO を置く構成は後段との切り分けやタイミング改善に有効である。よって、現在の実装は最小構成の一例であり、システム要件に応じて内部にバッファを施した拡張版 Special Join も妥当な設計選択となる。

4. CGRA-ME 上での実装

4.1 実装方針

本章では、3 章で述べた Special Join の論理そのものではなく、それを CGRA-ME の内部表現とマッパーの上でどのように扱ったかを述べる。Special Join は、CGRA-ME 上で新たなプリミティブとして実装した。実装にあたっては、既存の Elastic RIKEN CGRA のハンドシェイクを維持したまま、データチャンネルと EoP チャンネルを同時に扱えることを基本方針とした。具体的には、データ入力、EoP 入力、データ出力、および EoP 出力を持つ 2 入力 2 出力のプリミティブとして記述した。

また、CGRA-ME におけるアーキテクチャ記述では、Special Join を通常の演算器ではなく、データ整合性を担う制御プリミティブとして位置付けた。これにより、マッパーはデータ経路と EoP 経路を同一の Special Join インスタンスへ収束させ、そ

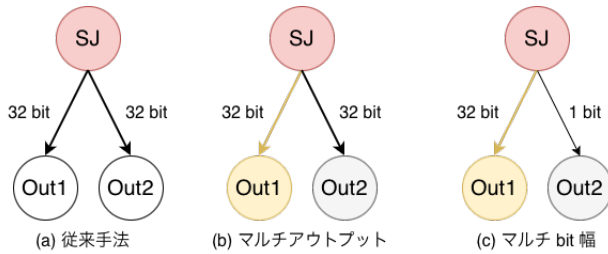


図7 マッパー上での表現

の後のデータ出力と EoP 出力を後段へ接続できる。すなわち、Special Join の導入は単なる Verilog RTL の追加にとどまらず、アーキテクチャモデル上でも専用資源として扱われる。

CGRA-ME の内部では、アプリケーションは演算グラフとして扱われ、各演算ノードの入出力に基づいてマッピングが進む。Special Join については、前述の通りデータ入力、EoP 入力、データ出力、EoP 出力を持つ 1 つの演算ノードとして表現した。ここで重要なのは、3 章で述べたように、SpecialJoin は論理的に 2 入力 2 出力のプリミティブであり、出力側にはデータ出力と EoP 出力という意味の異なる 2 つの出口が存在する点である。

このため、CGRA-ME 上では、Special Join を単なる「1 つの演算ノードから複数のファンアウトが出る」構造としてではなく、「異なる意味を持つ 2 つの出力ポートを備える演算ノード」として扱う必要がある。以下では、そのためのマルチアウトプットとマルチビット幅への対応について述べる。

4.2 複数出力への対応

図 7 に、CGRA-ME のマッパーにおける Special Join の表現を示す。図 7(a) の従来表現では、1 つの演算ノードから出る複数エッジは同一出力の複製として扱われる。これに対し、図 7(b) では、Special Join の data 出力と EoP 出力を意味の異なる 2 つの論理出力として扱っている。Special Join では、2 出力が単なるファンアウトではなく、後段で異なる役割を持つため、この区別を明示的に表現できなければならない。

そこで、本研究では各論理出力に独立の出力ポート識別子を与え、各出力エッジにその出力ポートを示すドライバー識別子を付与した。これにより、マッパーはデータ側のエッジと EoP 側のエッジを、単なる同一値の複製ではなく、異なる意味を持つ別個の出力として認識できる。実装上は、RIKEN CGRA を NoC 向けに拡張した先行研究で整備した CGRA-ME 側の拡張コードをベースに、Special Join の 2 出力へ適用した [10]。

したがって、本研究でのマルチアウトプット対応は、1 つの演算ノードが複数の論理出力ポートを持つことを明示的に扱う枠組みの上に構築されている。もしこの区別がなければ、データ出力と EoP 出力は同一の論理出力として扱われ、後段の接続先でどちらの出力を参照しているのかが曖昧になる。以上より、Special Join では出力ポート識別子に基づく複数出力対応が不可欠である。

4.3 複数ビット幅への対応

マルチアウトプットへの対応だけでは、Special Join の混在し

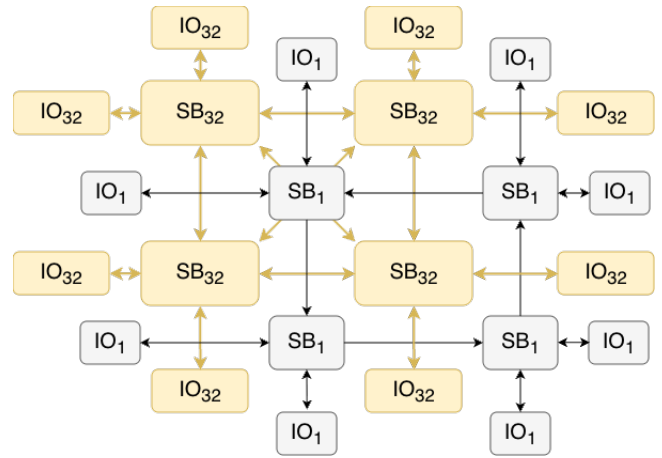


図8 1bit Network

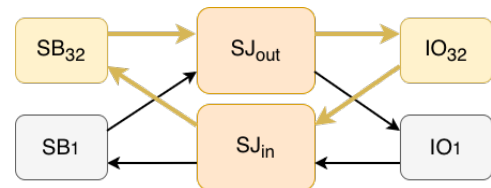


図9 Special Join の実装

ている bit 幅の出力を十分に扱えない。従来の演算グラフ表現では、1 つの演算ノードが事実上 1 種類の出力値しか持てず、複数の出力エッジは同一の出力値を共有していた [8]。この表現は、すべての出力が同一ビット幅である場合には問題にならない。しかし、Special Join ではデータ出力が 32 bit、EoP 出力が 1 bit であるため、同一の出力値を再利用するとビット幅衝突が生じる。すなわち、最初に生成された 32 bit の出力値が EoP 側にも再利用されると、1 bit であるべき EoP 出力まで 32 bit として解釈されてしまう。

そこで、1 つの Special Join ノードに対して、データ用出力値と EoP 用出力値という 2 種類の独立な出力値を持てるように演算グラフ表現を拡張した。図 7(c) にその表現を示す。前者は 32 bit の幅情報を持ち、後者は 1 bit の幅情報を持つ。さらに、マッパー後段の出力処理も、演算ノード単位ではなく出力値単位で扱うように変更した。これにより、データ出力は 32 bit、EoP 出力は 1 bit として独立に管理でき、(32 bit, 1 bit) のマルチ bit 幅な Special Join を正しく扱えるようになった。

この拡張によって、従来から扱えていた同一ビット幅のケースを損なうことなく、異なる bit 幅の 2 出力プリミティブを一般的に表現できるようになった。したがって、Special Join は単なる EoP 同期プリミティブにとどまらず、今後、複数の意味を持つ出力と異なるビット幅を同時に扱う他のプリミティブを追加する際の拡張点としても有用である。

4.4 1bit Network との接続

図 8 に、1bit Network を含むアーキテクチャ接続を示す。アーキテクチャ上では、データ入出力を通常のクロスバに、EoP 入出力を 1bit Network に接続した。これにより、CGRA 全体のデータバス幅を拡張することなく、EoP だけを 1 bit の制御情報

として独立に流しつつ、3章で述べた同期機構によって data と EoP を再結合できる。すなわち、Special Join は、32 bit のデータパスと 1 bit の制御経路を橋渡しする同期点として機能する。

この構成の利点は、EoP のために CGRA 全体のデータ経路を広げる必要がない点にある。データは従来どおりのクロスバを流れ、EoP は 1bit Network を流れるため、ハードウェア変更を局所化しやすい。また、CGRA-ME 上でも、データ側のエッジと EoP 側のエッジを異なるビット幅の出力値として扱うことで、ルーティングとビット幅整合を自然に記述できる。

具体的な Special Join の CGRA への組み込みを SpecialJoin を図 9 に示す。本実装では、入力側と出力側の双方で data と EoP の対応を維持するため、Special Join を 1 つではなく 2 つ用いる。すなわち、入力側の SJ_in は外部から CGRA へ流入するデータと EoP を同期させ、出力側の SJ_out は CGRA 内部から外部へ出力されるデータと EoP を同期させる。これにより、CGRA の内部に入る時点と外部へ出る時点の両方で、EoP とデータの整合を保証できる。

5. 機能検証

本研究では、図 8 に示した 32 bit クロスバと 1 bit Network を併用する構成を対象として、CGRA-ME から生成した Verilog RTL とテストベンチを用いて機能検証を行った。検証の主眼は、図 6 の同期機構が実際のアーキテクチャ上でも有効に機能し、(i)EoP とデータが常に一対一対応を保ったまま出力されること、(ii) 一方の入力が先着した場合でも他方が到着するまで整合が保持されること、(iii) 下流 stall 時にも同期済みの組が破壊されないこと、(iv) 複数出力およびマルチ bit 幅構成が CGRA-ME 上で正しく扱われることである。

検証では、代表的に 7 つのシナリオを確認した。すなわち、データと EoP の同時到着、データ先行、EoP 先行、同期済み後のデータ出力側のみ stall、EoP 出力側のみ stall、両出力同時 stall、ならびに 2 出力の区別と 32 bit データ/1 bit EoP のマルチ bit 幅構成の妥当性である。検証の結果、いずれのシナリオにおいても、Special Join は図 6 で意図したとおり Join 条件が成立するまで不整合な組合せを出力せず、また一方の出力のみが先に受理される場合でも、Eager Fork 相当の状態保持により同一の同期済み組を保ったまま後段へ伝搬できることを確認した。さらに、図 7 で拡張したマルチアウトプット、複数ビット幅の表現により、CGRA-ME 上でもデータ出力と EoP 出力が混同されることなく正しくマッピングおよび動作できることを確認した。

6. まとめと今後の展望

本稿では、Elastic CGRA において EoP 信号とデータを同期させるプリミティブ Special Join を提案した。Special Join は、データと EoP の一対一対応を保持し、入力の到着順や下流の stall によらず両者の整合性を維持できる。さらに、Elastic RIKEN CGRA を対象として CGRA-ME 上に実装し、複数出力および異なるビット幅を持つプリミティブとして扱えるようにした。機能検証の結果、各検証シナリオにおいて EoP とデータを対応

付けたまま正しく後段へ受け渡せることを確認した。

今後は、Special Join の面積、遅延、スループットへの影響を定量的に評価する。また、EoP だけでなく SoP を含む一般的なパケット境界情報への拡張を検討する。さらに、圧縮データストリームを用いる実アプリケーションに組み込み、SerDes や圧縮器・伸長器と連携したシステムレベルの有効性を評価する。これにより、Elastic CGRA 上でパケット境界情報を扱うための実用性と適用範囲を明らかにする。

謝 辞

本研究は、JST ASPIRE JPMJAP2341 の支援を受けたものである。

文 献

- [1] Yuanjie Huang, Paolo Ienne, Olivier Temam, Yunji Chen, and Chengyong Wu. Elastic cgras. In *Proceedings of the ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA '13)*, pp. 171–180. ACM, 2013.
- [2] Omar Ragheb, Rami Beidas, and Jason Anderson. Statically scheduled vs. elastic cgra architectures: Impact on mapping feasibility. In *2023 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2023.
- [3] Emanuele Del Sozzo, Xinyuan Wang, Boma Adhi, Carlos Cortes, Jason Anderson, and Kentaro Sano. Exploration of trade-offs between general-purpose and specialized processing elements in hpc-oriented cgra. In *2024 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2024.
- [4] Tomohiro Ueno, Kentaro Sano, and Satoru Yamamoto. Bandwidth compression of floating-point numerical data streams for fpga-based high-performance computing. *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, Vol. 10, No. 3, pp. 1–22, 2017.
- [5] Jordi Cortadella, Mike Kishinevsky, and Bill Grundmann. Self: Specification and design of a synchronous elastic architecture for dsm systems. Handouts of the International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems (TAU'2006), February 2006. Available at http://www.lsi.upc.edu/~jordicf/gavina/BIB/reports/self_tr.pdf.
- [6] Boma Adhi, Carlos Cortes, Yiyu Tan, Takuya Kojima, Artur Podobas, and Kentaro Sano. Exploration framework for synthesizable cgras targeting hpc: Initial design and evaluation. In *Proceedings of the IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2022.
- [7] Boma Adhi, Carlos Cortes, Yiyu Tan, Takuya Kojima, Artur Podobas, and Kentaro Sano. The cost of flexibility: Embedded versus discrete routers in cgras for hpc. In *Proceedings of the IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE, 2022.
- [8] S. Alexander Chin, Noriaki Sakamoto, Allan Rui, Jim Zhao, Jin Hee Kim, and Yuko Hara-Azumi. Cgra-me: A unified framework for cgra modelling and exploration. In *Proceedings of the 2017 IEEE 28th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pp. 1–8. IEEE, 2017.
- [9] G. Dimitrakopoulos, I. Seitanidis, A. Psarras, K. Tsiouris, P. M. Mattheakis, and J. Cortadella. Hardware primitives for the synthesis of multithreaded elastic systems. In *Proceedings of the 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1–6. IEEE, 2014.
- [10] Haoran Wei, Omkar Bhilare, Hamas Waqar, and Jason H. Anderson. Cad techniques for noc-connected multi-cgra systems. In *Proceedings of the 14th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies (HEART '24)*, pp. 109–118. ACM, 2024.