動的再構成可能プロセッサの FPGA 実装に関する研究

北　道　淳　司　　会津大学コンピュータ理工学部上級准教授

1 はじめに

近年、チップ内部の構成をシステムの動作中に変更することのできる動的再構成可能プロセッサが、IP フレックス、NEC エレクトロニクス、SONY など多くの半導体メーカ、大学の研究機関から提案・発表されている。動的再構成可能プロセッサは、必要に応じて動作を変更することが可能であるので、大規模なシステムを少ないハードウエアリソースにより実現する可能性があり、ネットワーク通信システムにおけるサーバシステムでのネットワークスイッチ機構において増大するプロトコルの種類に対応するスイッチング機能を実現したり、携帯端末においてより省電力（一般に省電力のデバイスは演算能力は劣る）のハードウエアリソースを用いても、操作する種類ごとにシステム構成を変更させ多種類の機能を実現するなど、今後ネットワークの基幹システムから端末システムにいたるまで、主要なハードウエア素子として注目されている。しかし、それぞれの動的再構成可能プロセッサは、演算機能の粒度、切り替えの機構などそれぞれ異なり、それぞれ得意とするアプリケーションが異なる。

本研究では、従来研究に引き続き、システムレベルからの段階的設計によって具体化された動的再構成可能プロセッサを実際の動的再構成デバイスにマッピングする方法を提案し、設計によって得られた動的再構成可能プロセッサの性能評価を行う。これにより、従来研究で行っているシステムレベルから設計された動的再構成可能プロセッサを実際のハードウエアまで実装する一連の設計手法となる。

2 Dynamic Module Library

システムレベルにおいて、モジュールおよびインターフェースの動的な生成・削除を含むシステムのモデリングを可能にする動的モジュールライブラリについて述べる。動的再構成可能プロセッサにおいて、モジュールおよびインターフェースは、プロセッサの動作中に生成・削除される。システムレベルにおいてこれらの振舞を自然に表現するためには、モジュール、ポートおよびチャネルが動的に生成・削除されなければならない。しかし、SystemC 2.1.v1 を用いたとしても、シミュレーションが開始された後、モジュール、ポートおよびチャネルを生成したり削除することはできない。我々は、動的に生成・削除可能なモジュール、ポートおよびチャネルとそれらの対する処理を含む動的モジュールライブラリを開発した。提案ライブラリは、モジュールの活性時間の記録などのプロファイリングの機能を実現している。提案ライブラリの詳細は論文[1]に述べられている。システムレベルでのモデリングにおいて提案ライブラリを用いると、通常の *sc_module* クラスに加えて、dynamic module クラスなどを用いることができる。これのクラスは、動的モジュール、ポートおよびチャネルに対して、生成・削除のための、*newmod()*あるいは *delmod()* のメソッドなどを提供する。動的モジュールでは、入出力ポートの宣言として用いられる *sc_in* の代わりに *dc_in* が、チャネルの宣言として用いられる *sc_fifo* の代わりに *dc_fifo* を用いて宣言する。これらのポートやチャネルは、シミュレーション後に、動的に生成・削除が可能であり、動的な接続・分断が可能である。例えばマルチコンテキスト型動的再構成可能アーキテクチャを用いたモデリングでは、動的モジュールにおける生成あるいは削除期間の動作は、外部システムからのコンテキストのローディング、コンテキストやレジスタの初期化、再構成を行うレジスタデータの退避に対応する。それらに加えて、動的再構成時のエラー処理についても表現する必要がある場合がある。 これらの処理のモデリングのために、提案ライブラリにおける動的モジュールは、あらかじめ用意された 3 つの状態(*creating*、*running*および *deleting*)を使用できる。各状態における処理は、動的モジュールの生成時、通常動作時、削除時の処理を定義することができ、設計者は各状態での処理をユーザプロセスとして宣言できる。生成および削除に必要な時間を、それらのモジュールの大きさやアーキテクチャの特徴に応じてそれぞれ *creating time* および *deleting time* というパラメータとして記述できる。動的モジュールの処理のオーバヘッドによってシミュレーション時間が増大する

可能性もあるが、動的モジュールにはいくつかの高速化手法が組み込まれており、必要に応じて必要なものを適用することができる。

３ 提案手法

３-１　システムレベルからの段階的設計

動的再構成可能プロセッサは、画像処理あるいはネットワーク処理など特定用途に用いられることが多く、システムレベルではプロセッサ構成を決定することが重要である。例えば下図(a)は動的再構成可能ユニット（Dynamically Reconfigurable Operating Unit: DROU）の入出力がプロセッサ内のレジスタファイルに接続されており、一方(b)では、外部のデータパスと接続されている。図(a)の構成は、演算データを通常の ALU と DROU を組み合わせて演算を行うような処理に適しており、一方図(b)の構成は外部のメモリとの間でストリーム状にデータを受け渡す処理に適している。大量のデータを処理するには図(b)の構成が有利であるが、DMA（Direct Memory Access）モジュールおよびそれらへの制御などが必要となる。システムレベルにおいては、動的再構成可能プロセッサは命令レベルでモデリングされ、高速なシミュレーションによりシステムの動作検証およびプロファイリングが行われる。

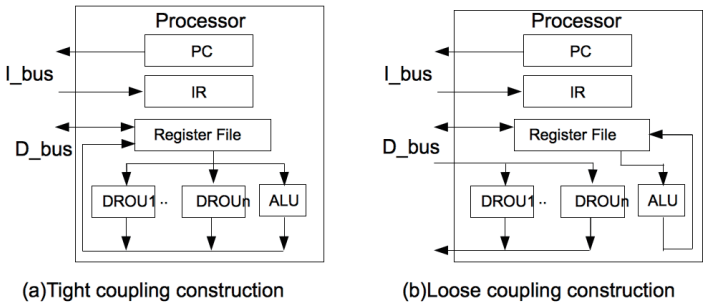

図１　システムレベルでの動的再構成可能プロセッサモデル例

従来研究において開発を行った命令レベルの動的再構成可能プロセッサのモデルおよび開発を行った動的再構成可能プロセッサを元に、レジスタ転送レベルでの設計内容を整理した。これらの内容は文献[2]にまとめられている。文献[2]では、上図(a)を具体化し、下図のような５段のパイプラインプロセッサとしてモデリングを行っている。

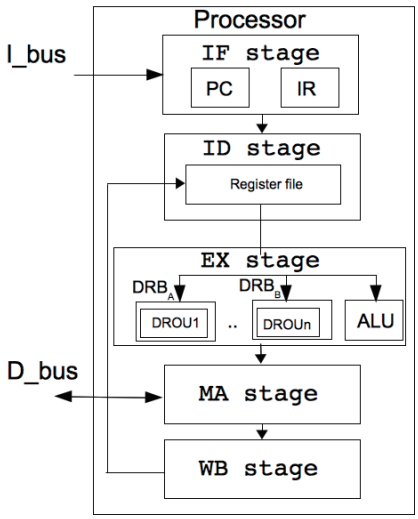

図２　サイクルアキュレートレベルでの動的再構成可能プロセッサのモデル例

上述のモデルは、２.で述べたDynamic Module Libraryを用いてモデル化されている。Dynamic Module Libraryを用いることによって、自然なモデリングおよび高速なシミュレーションによるシステム検証を行うことが可能である。

より具体的な設計レベルとしてサイクルアキュレートレベルあるいはRTレベルがある。このレベルでは、プロセッサ内部の構成がより具体的に決定される。パイプラインの段数や各ステージにおける動作が決定される。DROU もサイクルアキュレートレベルで設計され、実装すべき演算が具体的に設計される。このレベルのモデルについては文献[3]にまとめられている。

３-２　動的再構成機能の実デバイスへの実現方法

さらに具体的な動的再構成可能プロセッサを実装するデバイスを考慮するレベルを考える。動的再構成可能アーキテクチャの中から FPGA（例えば Xilinx 社の FPGA）を選択し、Dynamic Module Library でモデリングされた動的再構成部分を、採用したデバイスに実現するために、既存CADを用いて下位工程の設計が可能なように変更する。

Dynamic Module Library を用いてモデリングされた動的再構成可能プロセッサのモデルは下記のように分類される。

（１）固定的な命令セットを実行する静的なモジュール(例：プログラムカウンタ、レジスタファイル)

（２）動的再構成を制御、管理するモジュール

（３）Dynamic Module Library でモデリングされた動的再構成可能モジュール

（４）Dynamic Module Library でモデリングされた上述の（１)、（２）および（３）を接続するための動的に生成削除されるチャネル

これらの（１）および（２）は、システムの実行上常に静的に配置されるモジュールであり、これらは Dynamic Module Library を用いずにモデリングされた部分であるので、既存手法をもちいて下位工程の設計が可能である。

（３）は、さらに、(a)動的再構成において生成中に実行される動作（例えば生成中にモジュールへの制御が発生した場合にエラーの割り込みを発生させ、その制御を中断させるような動作）、(b)動的再構成モジュールが生成された後のどのモジュールの動作、(c)およびそのモジュールが削除中に実行される動作（生成時と同様に行われるエラー動作）がある。これらは Dynamic Module Library を用いて自然にモデリング可能である。このような分類に対して、(a)および(c)を、システムの実行時に静的に配置することにし、上述のように既存手法を用いることにした。しかし、この方法では、モデルによってはかなりのハードウエアリソースを消費する可能性があり、今後の実験において評価すべき項目である。

(b)については、動的に再構成する必要がある部分である。この部分に関しては、Dynamic Module Libraryを用いたモデルにおいては、プロセッサに動的再構成される演算モジュールが格納される外枠に相当するモジュールを用い、その中に動的に再構成される演算モジュールは１つと限定することにした。この外枠に相当するモジュールは１つのプロセッサに複数あり、演算モジュールは複数動的再構成可能である。このような制限は、動的再構成に関する命令セットにおいて、どの外枠にどのモジュールを生成させるという動的再構成に関係する命令を構成する各フィールを定義する際に、外枠に相当するモジュールの最大値、演算モジュールの種類の最大値を決定しているので、特に不自然な制約ではない。むしろ、動的再構成可能なハードウエアリソースに対して、領域を限定せずに大小様々なモジュールを動的に配置させるような実装方法も考えられるが、ハードウエアリソースを管理するために膨大なハードウエアリソースを必要とし、さらにパフォーマンスの低下を招くおそれがあると考えられる。

したがって、演算モジュールを格納する外枠に相当するモジュールに対して、そこに格納される最大の演算モジュールは、上位レベルのシミュレーションで確認することができるので、ハードウエア実装

において、最大の演算モジュールが格納可能な領域を FPGA 上に確保することにした。また動的再構成に必要となる時間も領域が確定されているので、見積りも容易となった。

（4）に関しては、採用する FPGA デバイスでは、これに対応する専用のバス領域を定義する。

4 まとめと今後の課題

本研究では、動的再構成可能システムレベルからの段階的設計によって具体化された動的再構成可能プロセッサを実際の動的再構成デバイスにマッピングする、従来研究で行っているシステムレベルから設計された動的再構成可能プロセッサを実際のハードウエアまで実装する一連の設計手法を提案した。提案手法では採用するデバイスとして Xilinx 社の FPGA デバイスを採用した。提案した設計手法を用いることにより効率的に Xilinx 社の FPGA アーキテクチャにマッピングすることが可能であることがわかった。

今後の課題として、他の FPGA デバイスあるいはカスタム LSI への実装を行い、本手法の有用性を検証することがあげられる。

【参考文献】

[1] Kenji Asano, Junji Kitamichi, Kenichi Kuroda: "Dynamic Module Library for System Level Modeling and Simulation of Dynamical Reconfigurable System," Journal of Computers, Vol.3, Issue 2, pp.55-62, Academy Publisher, Feb. 2008.

[2] Junji Kitamichi, "Proposal of Instruction Level Modeling for Dynamically Reconfigurable Processor using SystemC," 17th IFIP/IEEE International Conference On Very Large Scale Integration(VLSI-SOC 2009), Oct. 2009.

[3]Junji Kitamichi, Koji Ueda and Kenichi Kuroda:"A Modeling of a Dynamically Reconfigurable Processor using SystemC," 21st Int. Conf. on VLSI DESIGN 2008,pp.91-96, Jan. 2008.

＜発　表　資　料＞

題　　名	掲載誌・学会名等	発表年月
Proposal of Instruction Level Modeling for Dynamically Reconfigurable Processor using SystemC	17th IFIP/IEEE International Conference On Very Large Scale Integration	2009 年 10 月