Vol.2016-EMB-42 No.3
2016/8/25

SWORDS フレームワークにおける Stream 通信への対応および構成情報記述の JSON 化

谷　祐輔[†1]　高瀬　英希[†1]　高木　一義[†1]　高木　直史[†1]

概要：近年の組込みシステムでは，プロセッサと FPGA を組み合わせてヘテロジニアス構成を取るデバイスである，プログラマブル SoC が注目されている．我々は，Zynq のためのソフトウェア志向のシステム設計環境として，SoftWare-Oriented Design and Synthesis(SWORDS) フレームワークを開発中である．本稿では，現在の SWORDS フレームワークの開発の状況を述べる．まず，AXI-Stream プロトコルの通信に対応したことについて述べる．次に，HW タスク化する関数および通信インタフェースを指定するシステム構成情報の形式として，JSON を採用したことを述べる．

1. はじめに

近年の組込みシステムでは，プロセッサと FPGA を組み合わせてヘテロジニアス構成を取るデバイスであるプログラマブル SoC が注目されている．これによって，近年の組込みシステムにおいてますます高まっている，高性能化や複雑化の要求に応えることが期待できる．プログラマブル SoC の例として，Xilinx 社の Zynq-7000 All Programmable SoC（Zynq）が挙げられる．

Zynq はプロセッシングシステム（PS） とプログラマブルロジック（PL）で構成されている．PS はデュアルコアの Cortex-A9，キャッシュ，オンチップメモリ，外部メモリインターフェース，DMA コントローラおよび入出力ペリフェラルで構成されている．PL は FPGA を中心として，RAM，DSP，プログラマブル入出力ブロック，シリアルトランシーバおよび A-D コンバータで構成されている．Zynq には，3 種類の通信ポートが備わっている．GP-AXI（GP）は汎用ポートであり，PS と PL いずれもマスタとなれる．GP ポートは PS がマスタとなるものが 2 個，PL がマスタとなるものが 2 個用意されている．AXI-HP（HP）は，オンチップメモリまたは DDR メモリへのアクセスが可能である．AXI-ACP（ACP）は，L2 キャッシュを介したアクセスが可能である．マスタとスレーブを接続するため，Xilinx 社が提供する IP コアである AXI Interconnect を使用してポートと AXI モジュールを接続する．

Zynq のためのソフトウェア志向のシステム設計環境として，我々は SoftWare-Oriented Design and Synthesis（SWORDS）フレームワークを開発中である [1][2]．SWORDS の合成フローを図 1 に示す．SWORDS フレームワークを活用することで，プログラマブル SoC に関する深い開発知識を有していないシステム設計者でも，高品質なシステムの設計が可能となる．

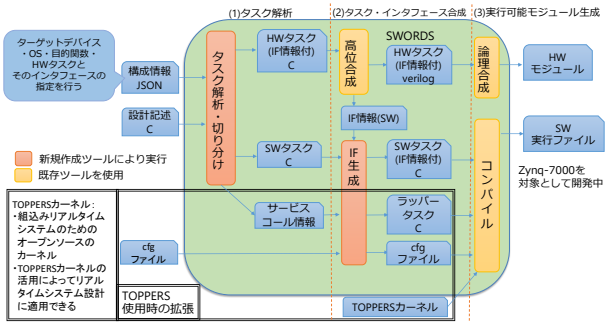

図 1　SWORDS の合成フロー

本稿では，現在の SWORDS フレームワークの開発の状況を述べる．まず，AXI-Lite，AXI プロトコルの通信に加え，AXI-Stream プロトコルの通信に対応したことについて述べる．次に，HW タスク化する関数および通信インタフェースを指定するシステム構成情報の形式として，新たに JSON を採用したことを述べる．

2. Stream 通信への対応

これまでの SWORDS では，AXI-Lite プロトコルおよび AXI プロトコルを使用した通信のみに対応していた [2]．本稿では，新たに AXI-Stream プロトコルを使用した通信に対応したことを報告する．AXI-Stream プロトコルを使用する

[†1] 現在，京都大学 大学院情報学研究科 通信情報システム専攻
Presently with Graduate School of Informatics, Kyoto University

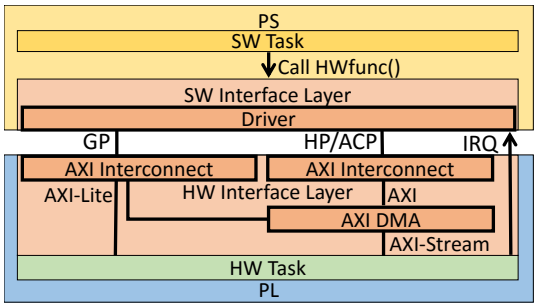

図 2　SWORDS における AXI-Stream を使用する通信方式の構造

通信方式の構造を図 2 に示す．

AXI プロトコルでは，HW タスクが memcpy 関数を実行することにより，ローカルメモリとの通信を行っている．対して，AXI-Stream プロトコルでは，主に AXI DMA が通信を担う．シーケンシャルなデータアクセスのみを行う HW タスクの場合，AXI-Stream プロトコルではローカルメモリへのバッファが不要となり，HW タスクの実行を高速化できる可能性があると考えられる．

HW タスクの実行について説明する．まず，SW タスクは SW インタフェースレイヤを呼び出す．次に，SW インタフェースレイヤは DMA の初期化と HW タスクがアイドル状態かどうかを確認する．アイドル状態が確認できた場合，SW インタフェースレイヤは HW タスクに開始信号を送り，DMA に入力引数の通信を指示する．その後，出力引数が取得可能であることを検知したのちに，DMA に出力引数の通信を指示する．HW タスクの終了を検知すれば，HW タスクの呼び出しは終了となる．

AXI-Stream プロトコル通信の性能を測定するため，評価実験を行った．設計対象は，unsigned long long 型の要素を持つ一辺 32 の二次元配列の組を入力とし，行列積を求める C プログラムを用いた．評価対象の設計は，SW のみで実行するもの，および，引数の通信に AXI プロトコルを使用するもの，AXI-Stream プロトコルを使用するものの 3 種類である．それぞれの HW の合成時の目標周波数は 100MHz とし，HW タスクの終了検知はポーリングで取得するものとした．3 種類の評価対象に対して，実行時間と FPGA の使用リソースを計測した．実行時間はそれぞれ 10 回計測した平均とした．表 1 および表 2 に，実行時間と使用リソースの結果をそれぞれ示す．

行列積の計算ではシーケンシャルでないデータアクセスを行うため，AXI プロトコルと同様にローカルメモリへのバッファが必要となり，AXI プロトコルを用いた通信よりも実行時間が長くなる．また，AXI DMA のためのリソースが必要となり，使用 FPGA リソース数が増加する．

表 1　実行時間

SW	2.953ms
AXI	0.201ms
AXI-Stream	0.300ms

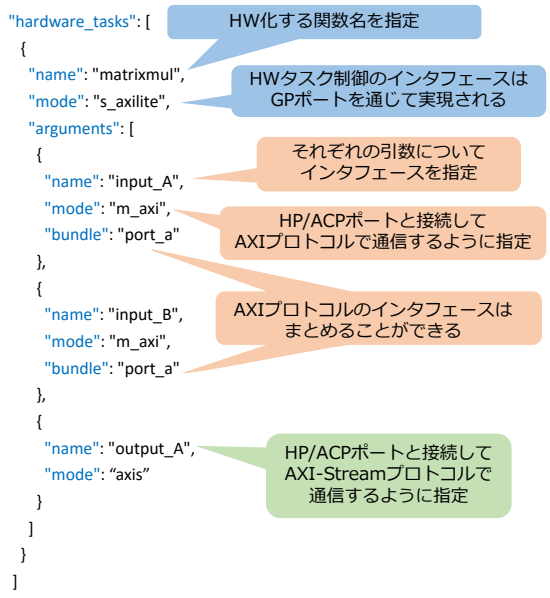

図 3　JSON 形式による HW タスクのインタフェースの指定

3. 構成情報の JSON 化

SWORDS では，設計者は C ソースコードに加えて構成情報ファイルを入力として記述する．構成情報ファイルでは，対象のプログラマブル SoC や HW タスク化する関数および通信インタフェースなどを指定する．従来，構成情報ファイルの形式を XML としていたものを，JSON とした．JSON を採用することにより，XML と比較して可読性・保守性が向上し，JSON Schema による型のチェックが可能になる．HW タスクのインタフェースの指定方法を図 3 に示す．

4. おわりに

本稿では，SWORDS フレームワークにおいて AXI-Stream プロトコルを使用する通信への対応を行い，構成情報記述の JSON 化を提案した．今後の方針として，HW タスクの通信方式，および最適化プラグマを自動で選択するための設計空間探索の検討を行うことが考えられる．

参考文献

[1] 谷祐輔，他：プログラマブル SoC のためのシステム設計環境における SW/HW インタフェース生成手法，電子情報通信学会技術研究報告，Vol. 115, No. 109，pp. 73–78 (2015).
[2] 谷祐輔，他：プログラマブル SoC のためのシステム設計環境における SW/HW 間通信方式の比較評価，電子情報通信学会技術研究報告，Vol. 175, No. 24，pp. 1–6 (2016).

表 2　使用 FPGA リソース

エレメント	全リソース	AXI	AXI-Stream
FF	106400	23656	26307
LUT	53200	33828	37194
Memory LUT	17400	6630	6636
BRAM	140	6	44.5
DSP48	220	220	220