TOPPERS 活用アイデア・アプリケーション開発コンテスト

部門　：　活用アイデア部門　　アプリケーション開発部門

作品のタイトル　：　SWORDS プラットフォームの TOPPERS 対応

作成者　：　京都大学高木研究室　（代表：畑山拓也・学生）

対象者　：　組込みシステム開発者

使用する開発成果物　：　TOPPERS/FMP　カーネル

目的・狙い

近年，CPU と FPGA を搭載したデバイスであるプログラマブル SoC が注目されている．プログラマブル SoC 上のシステムにおいて，リアルタイム OS を利用してソフトウェア（SW）タスクとハードウェア（HW）タスクを管理することでリアルタイム性の確保が容易となる．しかし，プログラマブル SoC 上でのシステム設計には SW 設計と HW 設計の両方の知識が必要となり困難である．そこでソフトウェア設計の知識のみで設計を行なうことが出来る SWORDS プラットフォームを TOPPERS カーネルに対応することでシステム設計を容易にすることができる．

アイデア/アプリケーションの概要

入力として, C 言語で記述されたシステム設計と XML 形式で記述された構成情報を与えることで，システム設計を SW タスクと HW タスクに切り分けて，プログラマブル SoC 上で実行可能なモジュールを生成する．タスクは関数単位とし，HW 化するタスクを構成情報ファイル内に記述する．SW 上に HW タスクのためのラッパーを作成することで，TOPPERS のサービスコールに対応する．

1　背景

近年の組込みシステムでは，プロセッサと FPGA を 1 チップ上に混載し，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．従来のヘテロジニアスシステムと比較すると，プロセッサと FPGA が別チップになったシステムに対してはプロセッサ/FPGA 間の通信速度が早いという利点が挙げられる．また，FPGA 上でソフトコアとハードウェアを利用するヘテロジニアスシステムに対しては，プロセッサの性能が高いという利点が挙げられる．プログラマブル SoC を用いたシステムでは，負荷が大きい処理は FPGA 上のハードウェア（HW) で実行し，柔軟性の求められる処理はプロセッサ上のソフトウェア（SW) で実行することにより，システムの高性能化と低消費エネルギー化が実現できる．さらに，この HW タスクと SW タスクの両方を含むシステム全体の管理をを RTOS によって行なうことで，リアルタイム性の確保や資源の管理が容易となる．

しかし，プロセッサと FPGA のそれぞれの利点を十分に活用するためには，SW 開発の知識と HW 開発の知識の双方が必要となる．さらに，SW と HW 間でタスク呼び出しなどの通信を行なうためには，専用の通信インタフェースについての知識が必要となる．このため，SW と HW の協調設計を実現することは難易度が高いものとなっている．

我々は，プログラマブル SoC のための SW 志向のシステム設計環境である，SWORDS(SoftWare ORiented Design and Synthesis) プラットフォームを開発中である．SWORDS プラットフォームにより，設計者は HW の知識を必要とせずにプログラマブル SoC を活用するシステムを設計することが可能となる．

本稿では，SWORDS プラットフォームを TOPPERS に対応するアイデアを検討する．SWORDS プラットフォームを TOPPERS に対応することで，TOPPERS を利用する既存のシステムの一部を HW 化することが出来るようになり，高性能化や低消費エネルギー化の恩恵が見込める．現在，SWORDS プラットフォームはプログラマブル SoC として Zynq を対象としており，Zedboard 対応のパッケージが存在することから TOPPERS/FMP カーネルに対応する．

2　予備評価

まずはアイディアである SWORDS プラットフォームの TOPPERS 対応の有効性を確認するため，Zynq を搭載した評価ボードである Zedboard を用いて予備評価を行なった．設計対象として，高位合成用ベンチマークである CHStone（http://www.ertl.jp/chstone/）より SHA を用いた．SHA のメイン処理を，与えられた入力コマンドに応じて SW タスクもしくは HW タスクで実行するシステムを SWORDS を利用して設計した．このシステムを用いて，SW タスクと HW タスクそれぞれの実行時間を測定した（図 1)．表 1 に SW タスクと HW タスクそれぞれの実行時間を 10 回測定した時の平均値を示す．表 1 より，タスクを HW 化することにより実行時間を短縮できることが確認できた．

したがって，SWORDS プラットフォームを利用して，SW タスクを HW タスク化することでシステムの高性能化が行えることが確認できた．しかし，現在 HW タスク化は手作業で行なっており，この作業を自動化することが本アイデアの目的である．

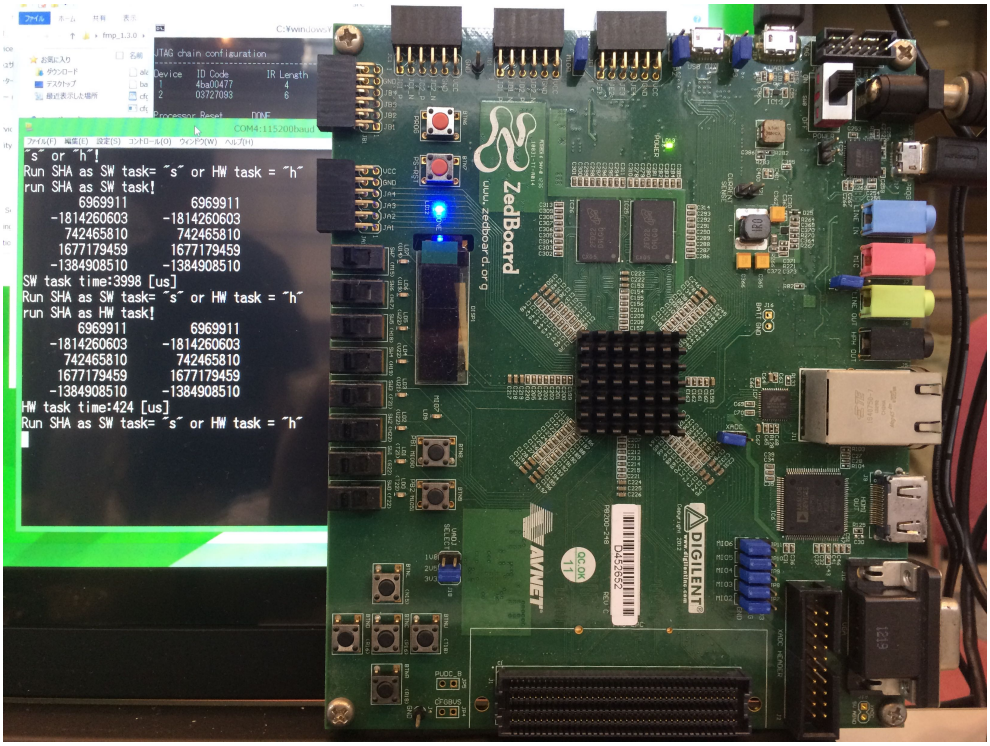

図 1: 予備評価イメージ

表 1: 実行時間

SW タスク	3.996 [ms]
HW タスク	0.423 [ms]

3 SWORD プラットフォーム

図 2 に SWORDS プラットフォームを示す．SWORDS では，XML 形式で記述された構成情報ファイルと C 言語で記述されたシステム設計ファイルを入力とし，HW モジュールと SW 実行ファイルを出力とする．構成情報の XML の構造を図 3 に示す．図に示すように，構成情報ファイルには，デバイスの情報や SW と HW それぞれで実行するタスクといった情報を記述する．つまり，この XML を書き換えるだけでシステムの SW/HW 構成を変更することが可能となる．また，ここではタスクは関数単位で指定する．現在 SWORDS ではプログラマブル SoC として Zynq を対象としている．

SWORDS プラットフォームのワークフローを説明する．

- タスク解析

構成情報に基づき入力の C ソースファイルを解析し，SW タスクと HW タスクに分割する．さらに，SW と HW 間のインタフェース生成のための情報を生成する．

- タスク・インタフェース合成

HW タスクとインタフェースの合成を行う．HW タスクの生成には，SW から HDL を合成する技

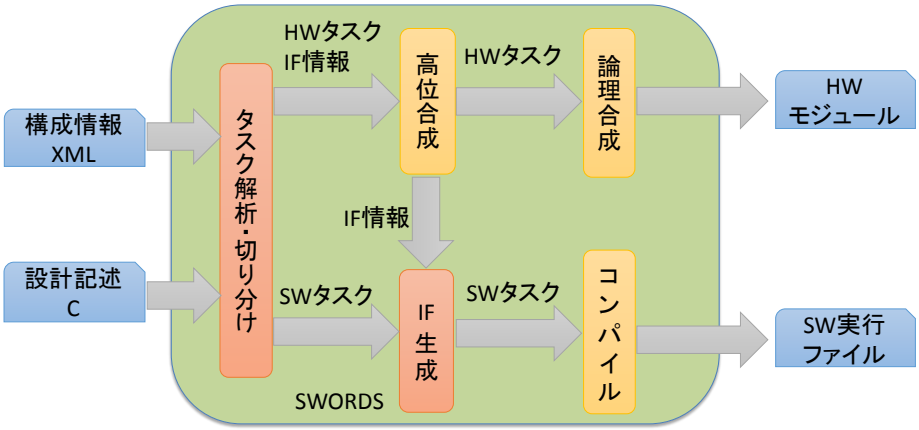

図 2: SWORDS プラットフォーム

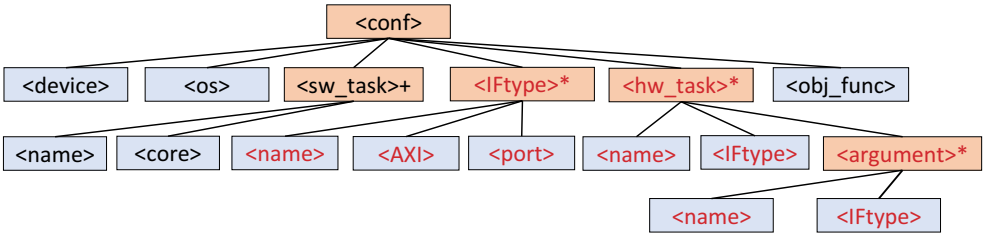

図 3: 構成情報

術である高位合成を利用する．タスク解析時に生成された SW タスクにはインタフェースに関する情報を追加する．HW に関するインタフェース情報は HDL 記述に含まれる．

- 実行可能モジュール生成

ターゲットとするプログラマブル SoC 上で実行可能なモジュールを生成する．

4 TOPPERS 対応

SWORDS を TOPPERS に対応するためには，TOPPERS のサービスコールが本来実現する動作に応じて HW タスクの動作を制御することが必要になる．このために，本アイデアでは，プロセッサ上の SW として HW タスクのためのラッパーを生成する手法を提案する．SW のラッパーは，SWORDS プラットフォームによって自動生成されることを検討している．ラッパーを用意することにより，TOPPERS カーネルは FPGA 上の HW タスクを通常のタスクと等価に扱える．さらに，セマフォ等のカーネル資源の操作も同様に扱えるようになると考えられる．

SWORDS プラットフォームによりラッパーを生成する利点は，以下の通りである．

- TOPPRES を利用した既存のシステムのソースコードをそのまま活用できる

- カーネルのソース自体に手を加える必要がない

SWORDS を用いて TOPPERS/FMP を含むシステムを Zynq 上に実装するアイデア図を図 4 に示す．図に示すように，HW タスクの実行は SW 上に実装されたラッパーを介して実行される．これにより，TOPPERS/FMP から見ると HW タスクを SW タスクとして扱うことが出来るため，act_tsk 等のサービスコールは SW 上のラッパーを対象として実行でき，cfg ファイルを含むソースファイルの大幅な修正は不要となる．

しかし，現在は SW タスクを HW タスク化する作業の自動化が出来ておらず，セマフォやメールボックス等の OS 資源の活用もできていない．また，複数の HW タスク間で通信を行う場合，提案手法では一度 SW を経由するためオーバーヘッドが大きくなるという問題がある．この問題に対処するために，HW タスク間での通信を行なうことも検討中である．また，セマフォなどの OS 資源について，SW 上だけでなく HW 上に生成して利用することも検討中である．このためには，TOPPERS カーネル自体に手を加える必要があると考えられる．

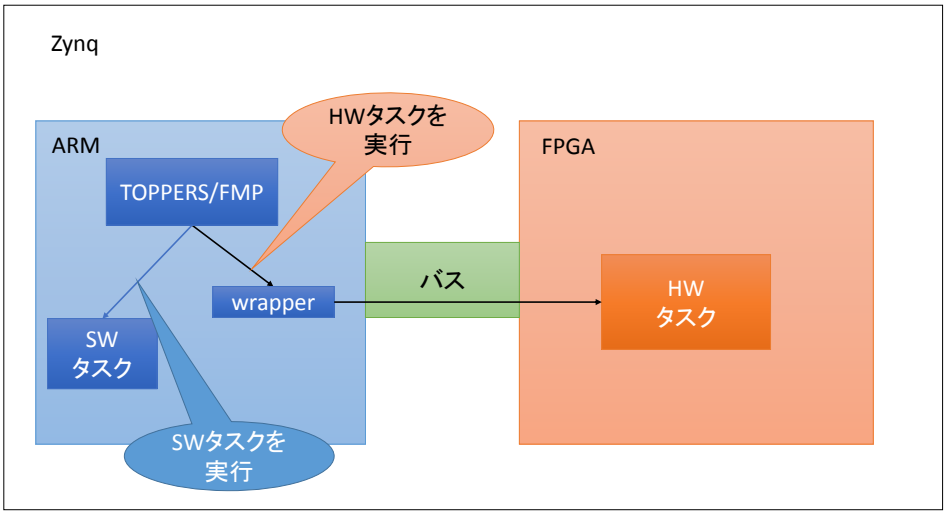

図 4: TOPPERS/FMP を含むシステム