K-041

動的再構成可能計算機を用いた論理回路デザインシステム
Computer Architecture Design System with Dynamically Reconfigurable Computer

杉江 崇繁 *
Takashige Sugie

1. まえがき

ハードウェアの教育の 1 つとして論理回路を設計させることがよくある．Large Scale Integration(LSI) の発達によって人の手だけによる開発が非常に困難になり，Electronic Design Automation(EDA) などのコンピュータを用いた開発ツールに依存してしまうことが常である．一般にハードウェアシミュレータを使用することが可能であり，実際にハードウェアを開発する際には無くてはならないツールの 1 つである．ところが，教育の観点から見ると講義時間，設備や開発難易度などの問題から，シミュレータ上で正解を得ることが目的となってしまう場合がある．つまり学生は自分が作成したプログラムを実際にハードウェアへ実装させることがないため全ての工程が仮想状態であり，自分がいったい何を作ったのか，混乱したり理解に苦しんだりする．これでは応用が利かないばかりか，実機で動作をさせることすら困難である．

学生が自分でプログラムした論理回路をハードウェアに実装して操作することにより，シミュレータでは解りづらい演算速度などを体感することや Operating System(OS) からハードウェアの制御方法を学習させる．そして理解をより一層深めることが本研究のねらいである．しかしながら，現実問題としてハードウェアの制御は学生にとって非常にレベルの高い課題となってしまう．そこで，ハードウェアとの通信制御に関する部分を汎用的に開発してしまうことで，学生が安全かつ容易に操作できるようにすることを目的とする．同時に，PCI-Express を外部インターフェイスに持ち，動的再構成可能システムを採用した計算機ハードウェアの開発も目的とする．

2. 動的再構成可能ハードウェア

任意の論理回路を実装させるにはプログラム可能なデバイスが必要である．幾つかのプログラム可能なデバイスが存在するが，本研究では Field Programmable Gate Array(FPGA) を採用した．FPGA は論理回路をプログラムできる上，Micro Processing Unit(MPU) や Digital Signal Processor(DSP) なども実装されており，多様な講義内容に対して柔軟に対応することができる．また，PCI-Express バスをインターフェイスに取ることができるのは FPGA だけである．

FPGA は自身の論理回路を構成するためにコンフィギュレーションデータを必要とする．このデータは学生がプログラムした Hardware Description Language(HDL) ソースコードから生成されるものである．一般に FPGA を機能させるには，電源投入時に FPGA が自動的に専用の Programmable Read Only Memory(PROM) から読み込む形になる．PCI-Express バスなどを介してホストコンピュータへ接続されている場合，その OS が新しい論理回路を認識するためには再起動をする必要がある．これは非常に手間のかかる作業になり，PROM へデータを書き込む機器も必要になる．そこで，本研究では図 1 のような仕様を持つ計算機ハードウェアを開発した．大きな特徴は FPGA を Communication Configuration Unit(CCU) と Tuned Processing Unit(TPU) に分けるところにある．CCU は PCI-Express バスと TPU をブリッジするような位置に設定する．CCU は電源投入時に自身の回路を PROM から読み込んで構成する．すると，PCI-Express バス側にあるホストコンピュータからは常時 PCI-Express デバイスとして機能しているように見える．一方，CCU から TPU に対してコンフィギュレーション用バス (図中の桃色線) とデータ通信用バス (図中の黄色線) を接続する．TPU は CCU からコンフィギュレーションデータを受け取ることで自身の回路を構成する．つまり，TPU はホストコンピュータから PCI-Express バスを介して，任意のコンフィギュレーションデータを受け取ることが可能になる．TPU のコンフィギュレーション中はその FPGA がダウンするが，PCI-Express バスに対しては CCU が応答するので，ホストコンピュータをシャットダウンさせることなく TPU の論理回路を変更することができる．このシステムは任意の時間に任意の論理回路を構成することが可能であり，そのオーバヘッドが小さいという利点がある．FPGA の回路規模にもよるが，1 秒以下で書き換えが完了する．ゆえに，間違ったプログラムを作成しても，何度でも繰り返し再構成することができるため学生は論理回路の学習に打ち込める．教育者側の立場から見ると，本計算機ハードウェアを Linux などのサーバに搭載しておくことにより，マルチユーザ環境で使用することができる．誰かがデバイスを操作していてリソースビジーでない限りデバイスを共有して利用できるため，備品の購入を最小限に抑えることも可能である．

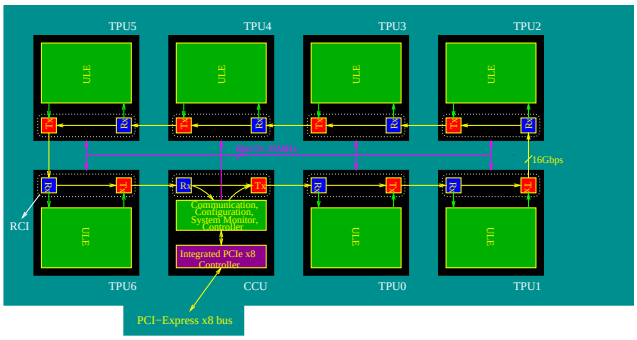

図 1: Palette ハードウェア構成図

*東京工科大学, Tokyo University of Technology

3. 汎用モジュール

図 2 はエンドユーザからハードウェアに到達するまでの階層構造を示している．エンドユーザが発行した命令は様々なプログラムや装置を経由して論理回路へ到達する．論理回路の学習では最も低レベルの階層と最も高レベルの階層を扱うことになる．これらの階層の間には開発難易度の高い層が密集しており，短期間での学習には不向きである．

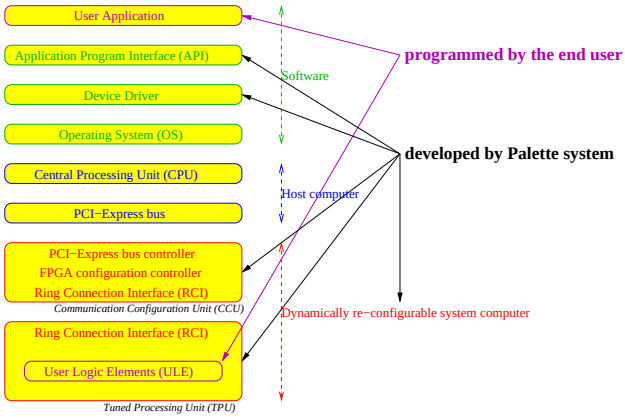

図 2: エンドユーザから論理回路までの階層構造

図 1，図 2 から伺えるように，本研究で提案しているシステムは複雑である．そこで，通信制御系統を汎用的に開発してしまう．ハードウェア上で該当する箇所は，図 1 における CCU と Ring Connection Interface(RCI) である．ホストコンピュータに接続される CCU を開発するため，デバイスドライバや Application Program Interface(API) の開発も含む．RCI に接続されている User Logic Elements(ULE) へは，メモリ空間を用いて汎用的なアクセス方法を提供する．また，ULE から CCU に対しての通信には必ず RCI を通過するので，学生が作成した論理回路に不備があってもハードウェアを損傷することはない．

TPU および CCU におけるモジュールは VHSIC Hardware Description Language(VHDL) を用いて開発した．TPU ではメモリ空間を用いた汎用インターフェイスを RCI に実装しており，ULE におけるデータ通信の難易度を最小限に抑えている．CCU は PCI-Express バスを用いたホストコンピュータとの通信制御と TPU のコンフィギュレーション制御を行う．本研究では CCU とそのデバイスドライバを開発する．したがって，学生は低レベルながらもシステムコールという汎用インターフェイスの仕様に従ってコーディングすれば，自分がプログラムした論理回路へアクセスすることが可能になる．また，API も開発するのでシステムコールや例外処理などの面倒な処理からも解放される上，ULE 空間を直視することも可能になる．これら，TPU と CCU における VHDL ソースコードおよびデバイスドライバと API の C/C++ソースコードは sourceforge.net[1] で一般公開する予定である．より深くハードウェアの勉強がしたい学生の手助けになり，本研究におけるバグの発見や機能の発展を望むこもできる．

表 1: 開発環境

ホストタイプ	i686-pc-linux-gnu
	x86_64-pc-linux-gnu
ソフトウェア	linux-2.6.24.7
	binutils-2.18
	gcc-4.2.4
	glibc-2.7
	make-3.81
	m4-1.4.11
	libtool-2.2.4
	automake-1.10.1
	autoconf-2.61
	gettext-0.17
	ptetex3-20080616
	X.org-1.4.0
	gtk+-2.12.10
	ise-10.1.02
	vpp-2.0.3

4. まとめ

本研究は教育を対象としているが，計算量に問題を抱えている研究者にも活用することができる．例えば，気軽にスーパーコンピュータを利用することは困難であるし，ハードウェアの知識が少ないため自分で計算機を開発することもできないといった場合である．一般の数値計算ではそれ自身の論理は比較的単純であることが多い．つまり，計算機の開発を困難にしているのは，それを制御するための複雑な通信プロトコルや OS の仕組であると言える．本研究ではその処理部分を汎用的に開発して一般に公開するため，開発の難易度の軽減や開発期間の短縮などが実現できる．本研究の計算機ハードウェアを用いれば任意の論理回路を動的に設定できるため，専門分野を問わず有効に活用することが可能である．特に多種多様な解析手法を併用する場合には非常に大きな効果が得られる．

インターネットを介してユーザが開発した論理回路を実機へ実装し，動作させることのできるシステムの構築も視野に入れている．これは遠隔地からのハードウェア操作が可能になることを意味する．例えば大学のキャンパスが離れていても環境は変わらないので，同じ講義を受講することができる．また，同時に議論できるコミュニティ環境をウェブサーバで提供すれば，バグの早期発見，アルゴリズムの熟考や新発見なども期待できる．学生にとっても研究者や企業の開発者の意見を聞くことができるため，教科書には載っていない実用的な技を教わることができるなど，非常に大きな刺激を受けることになる．

参考文献

[1] Palette for computer architecture design, http://sourceforge.net/projects/palette/.