動的再構成可能ハードウェアと汎用モジュールの開発
- Palette for computer architecture design -

1. 背景

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ります. 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す．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 つです．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ります．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します．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す.

2. 目的

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することにより，理解をより一層深めることがねらいです．とはいうものの，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います．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します．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します.

本プロジェクトは教育を対象としていますが，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ます．例えば，気軽にスーパーコンピュータは利用できませんし，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す．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です．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ます．本プロジェクト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ます．提案している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ますから，専門分野を問わず有効に活用することができます．一方，バイオインフォマティクスのように多種多様な解析手法を併用する場合でも同様の理由から非常に大きな効果が得られます．企業などでは，自社で設計した試作プログラムの動作確認やデバッグとして繰り返し利用できます．

また，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ます．遠隔地からのハードウェア操作が可能になりますから，例えば学部のキャンパスが離れていても環境は変わりませんので，同じ講義を受講することができます．

3. 開発の内容

本プロジェクトはユーザが自由な発想で論理回路をデザインすることができるシステムです．そのハードウェアは丁度，絵の具を混ぜて無限の色を生み出すための受け皿であるパレットの様です．本システムは画期的なコンピュータアーキテクチャの発明やそれらの相互作用による発展や進化の基盤の位置付けとして”Palette”と呼んでいます．

3.1 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であるデバイスが必要です．Complex Programmable Logic Device(CPLD) や IPFlex 社の DAPDNA など幾つかのプログラム可能なデバイスが存在しますが，本プロジェクトでは Field Programmable Gate Array(FPGA) を採用します．FPGA は論理回路をプログラムできることはもちろん，Micro Processing Unit(MPU) や Digital Signal Processor(DSP) なども実装されており，多様なアプリケーションに対して柔軟に対応することができます．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 だけです．

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します．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す．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ります．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ります．これは非常に手間のかかる作業になりますし，PROM へデータを

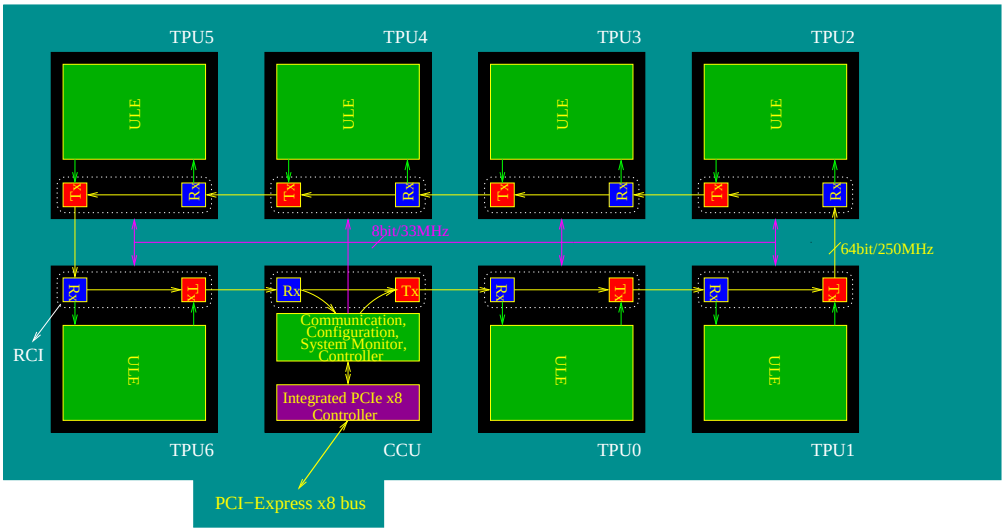

図 1: Palette ハードウェア構成図

書き込む機器も必要になります．そこで，本プロジェクトでは図 1 のような仕様を持つ計算機ハードウェアを開発します．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ります．CCU は PCI-Express バスと TPU をブリッジするような位置に設定します．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ようにします．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ます．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) を接続します．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します．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ります．TPU のコンフィギュレーション中はその FPGA がダウンしますが，PCI-Express バスに対しては CCU が応答しますので，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ます．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ります．FPGA の回路規模にもよりますが，1 秒以下で書き換えが完了します．ゆえに，間違ったプログラムを作成しても何度でも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ます．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ます．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ますから，備品の購入を最小限に抑えることができます．

3.2 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ます．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します．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ります．これらの階層の間には難易度の高い層が密集しており，短期間での学習や開発には不向きです．

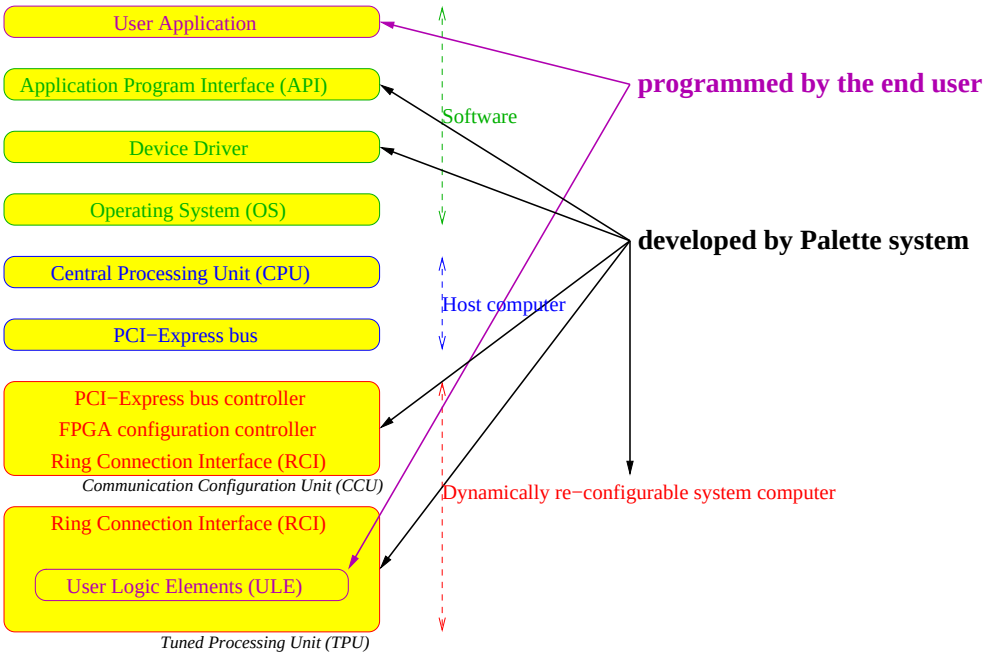

図 2: エンドユーザから論理回路までの階層構造

図 1 から伺えるように，本プロジェクトで提案しているシステムは複雑です．そこで，通信制御系統を汎用的に開発してしまいます．ハードウェア上で該当する箇所は図 1 上では CCU と Ring Connection Interface(RCI) です．CCU を開発しますから，デバイスドライバや Application Program Interface(API) の開発も含みます．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します．また，ULE から CCU に対しての通信には必ず RCI を通過しますので，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ありません．

TPU および CCU におけるモジュールは VHSIC Hardware Description Language(VHDL) を用いて開発します．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ます．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います．本プロジェクトでは CCU と

そのデバイスドライバを開発します．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ります．また，API も開発しますから，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ます．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 などのようなオープンソースコミュニティで公開します．より深くハードウェアの勉強がしたい学生の手助けにもなりますし，本プロジェクトにおけるバグの発見や機能の発展を望むことができます．

3.3 遠隔利用

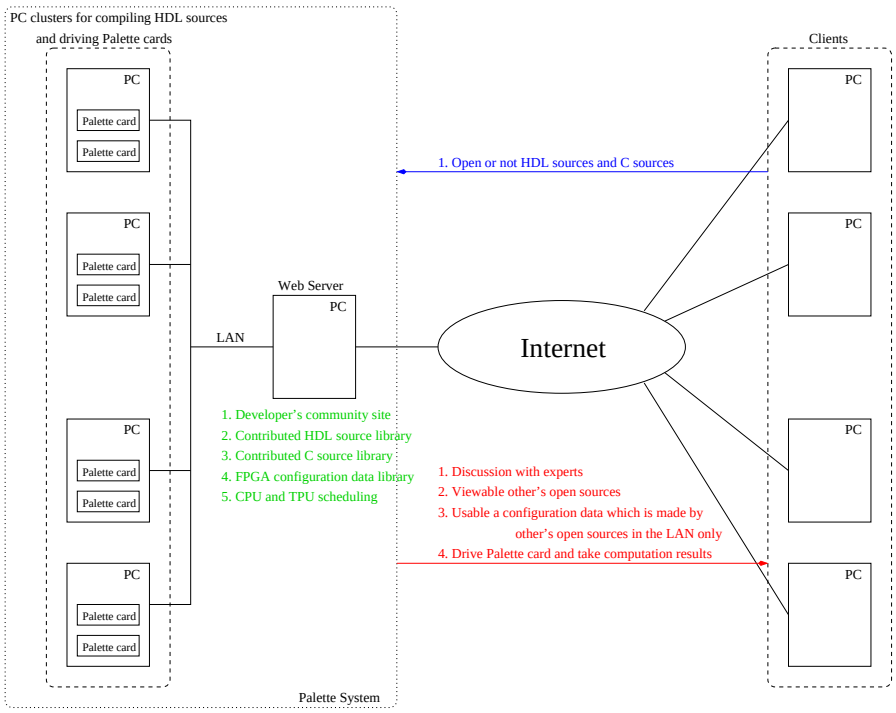

図 3: インターネットを通じた Palette ハードウェアの一般公開システム

インターネットを通じて計算機ハードウェアを提供するウェブサイトを立ち上げると面白いのではないかと考えています．現状の案では，図 3 のようなコミュニティサイトが出来ればと思っています．クライアントは HDL ソースコードと C/C++ソースコードをオープンソースかそうでないかを指定して，ウェブサーバにアップロードします．提案しているシステム内で EDA を使い，HDL ソースコードから TPU のコンフィギュレーションデータを作成します．これは非常に大きな CPU リ

ソースを必要としてしまいますが，TPU の配線などが適切に設定されているという保証を得るための措置です．そして TPU を用いて動的に実行され，計算結果のみをクライアントへ返します．つまり，クライアントはソースコードを送信することで EDA を使ったコンパイルが不要になりますし，計算機ハードウェアを購入せずに FPGA を利用することもできます．また，オープンソースとしての提供者は他のオープンソースのコンフィギュレーションデータを Local Area Network(LAN) 内に限って利用可能にします．コンフィギュレーションデータを再利用しますから，開発コストを大幅に削減することができます．このようにクライアントは時間的にも経済的にもコストを抑えられるメリットがありますから，必然的に多くの開発者が集うのではないかと考えております．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ます．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と思います．

3.4 動作環境

以下のホストを想定して開発を行っています．(2008 年 4 月 22 日現在)

- i686-pc-linux-gnu
- x86_64-pc-linux-gnu

以下のソフトを用いて開発を行っています．(2008 年 4 月 22 日現在)

- linux-2.6.24.4
- binutils-2.18
- gcc-4.2.3
- glibc-2.5
- make-3.81
- m4-1.4.11
- libtool-2.2.2
- automake-1.10.1

- autoconf-2.61
- gettext-0.17
- ptetex3-20080414
- X.org-1.4.0
- gtk+-2.12.9
- ise-10.1
- vpp-2.0.3

4. 従来の技術との相違

外部インターフェイスに PCI-Express バスを備えた動的再構成可能システムは存在しますが，PCI-Express バスから直接 FPGA へコンフィギュレーションデータを送信できる計算機は見当たりません．FPGA のコンフィギュレーション時間が短いことは大きな利点の一つです．

一般に C 言語などから論理回路を作成できるコンパイラは存在します．開発期間の短縮や開発難易度を大幅に低減させることができ，大変有用なツールです．本プロジェクトではユーザがプログラムする論理回路と，それを操作する C/C++言語の間を全て隠蔽するように各モジュールを設計しています．論理回路の設計では，まだまだ興味深い発想へ至る可能性があり，奇抜性や新規性のあるアーキテクチャを創造することができます．本プロジェクトは論理回路の学習が第一の目的でありますが，論理回路の作成を全てコンパイラに任せてしまうと論理回路の性能が完全にコンパイラ依存になることにも考慮し，ユーザの論理回路についてはなるべく触れないように設計しています．つまり，その部分には最大限の自由度を与え，初心者でも直ぐに理解して扱えるようなモジュール群を設計していることが大きな特徴です．例えば，ユーザ空間から論理回路のメモリ空間を直接見てプログラムすることができますから，非常に高度な論理回路の学習でも簡単に制御することができます．このようなモジュールは既に存在するかもしれませんが，動的再構成可能システムを採用している点，ハードウェアプログラムとソフトウェアプログラムを一括管理している点，何よりもオープンソースとして一般に公開する点で従来との相違があると考えています．本プロジェクトでは隠蔽するべき所は根こそぎ開発してしまいますが，その技術までを隠蔽しません．

本プロジェクトでは任意の論理回路を遠隔操作可能なシステムまで発展させます．また，一般に公開して利用できるようなシステム設計を行っていることも特徴の一つです．

5．期待される効果

高いレベルの論理回路学習がスムースに行えることが挙げられます．21 世紀の時代に，全加算器程度を学習して終わるのはつまらないと思います．もちろん，基本の学習は非常に大切ですが，そこから数歩先の技術をさらに体験させる学習は，非常に大きな興味を与える効果があります．計算機に対する理解度も深いものとなり，優れた人材の育成に貢献できると考えています．

本プロジェクトの計算機は専用の論理回路を実装していながらそれを汎用的に扱えますので，複数のアルゴリズムが必要な高速計算などの分野では大きな効果を期待することができます．また，繰り返し論理回路を変更できるという点から，企業などでは自社で開発したプログラムのデバックや複数の試作回路を実装してテストすることもできます．

今のところ運用費や運用場所を考えていませんが，TPU を一般に開放するシステムの構築を目指しています．これは離れた場所で同じ論理回路などの講義を受講することにも利用できますが，ハードウェアの開発には費用がかかるため手を出せない会社や研究室などにとって，導入初期段階においてある程度の開発の指標を示せる効果があります．開発に必要な言語，ツール，デバイスやハード化によって得られる効果までをも推し量ることができます．もし本システムの利用によって多くの人が計算機の開発を推進するようになるならば，それらによって波及する経済効果も大変大きなものになると思います．

ここで，本システムの利用者として想定される人をまとめてみます．将来の技術者のたまごである学生，学術機関や企業における研究者や技術者，計算機に全く触れたことのない人，つまり大変幅広い多くのユーザの利用が考えられます．本システムはインターネット上に公開しますから，そこにコミュニティ環境を用意すれば様々な意見が発言されます．より良いアーキテクチャの考案へ発展することを期待できますし，学生など勉強を目的とするユーザには大変有意義な情報がもたらされる効果もあります．そしてアップロードされた論理回路などのプログラムは全てライブラリ化されますので，論理回路の相互利用が可能になります．多数のユーザが各々の発想によって利用することで全く新しい使い方や組み合せが考案されたり，斬新なアーキテクチャの発想へ至るヒントになったりすることが期待できます．これらの効果は実際にシステムを稼働させてみないと全く予想がつきません．ただ，非常に興味深い展開になることは想像に難しくありません．

C 言語などの関数をリンクするように，Palette システムに蓄積させている FPGA のコンフィギュレーションデータをオンデマンドで要求し，TPU を使用しながら動作可能な実行ファイルを生成できるリンカの開発も行いたいと思っています．

6. 普及の見通し

2008 年 4 月 22 日現在では，本プロジェクトのソースコードを一般公開するに至っていません．本プロジェクトの開発者レベルにおけるデバッグを完了し，早急に一般公開します．これにより本システムのローカル利用においては普及の目処が立ちます．同時に，より多くのユーザからのフィードバックによってシステムの安定化を図ることができます．特に大学の講義などで学生の利用に耐えられることができれば，インターネットによる不特定多数のユーザ利用へシステムを拡張することが可能になると考えています．この段階でインターネットによる一般公開へ向けたシステムの開発に取りかかります．ここまで到達すると，本プロジェクトで想定している全てのユーザへの普及環境が整います．この後は本システムの利用状況から判断して，より良いシステムへ発展させていきたいと考えています．

7. 開発者名

杉江崇繁 (東京工科大学 コンピュータサイエンス学部 助教)

プロジェクトページ (予定)
http://sourceforge.net/projects/palette/