計算機アーキテクチャ　103－13
設　計　自　動　化　69－13
(1993. 12. 17)

教育用計算機QP-DLXの開発と開発環境

岩井原 瑞穂, 山家 陽, 中川 智水, 國貞 勝弘, 斎藤 靖彦, 永浦 渉, 池 兼次郎,
中村 秀一, 山田 哲也, 村上 和彰, 安浦 寛人

九州大学 大学院総合理工学研究科 情報システム学専攻

〒816 福岡県春日市春日公園 6-1

E-mail: {iwaihara, yamaga, tnakagaw, kunisada, saitoh, nagaura, ike,
nakamura, yamada, murakami, yasuura}@is.kyushu-u.ac.jp

あらまし　計算機工学教育用32ビットRISC型マイクロプロセッサQP-DLX(Kyushu University Education-Purpose DLX Microprocessor)の開発プロジェクトにおける，ハードウェア記述言語SFLによる設計の過程を報告する．QP-DLXの開発環境，設計結果，CADベンチマークとしての設計データの公開，QP-DLXを用いたマイクロプロセッサ設計演習の構想について述べる．

和文キーワード　教育用マイクロプロセッサ，論理合成，ハードウェア記述言語，命令パイプライン

Development and Design Environment of the Educational Computer System QP–DLX

Mizuho IWAIHARA, Akira YAMAGA, Tomomi NAKAGAWA, Katuhiro KUNISADA,
Yasuhiko SAITOH, Wataru NAGAURA, Kenjiro IKE, Shuichi NAKAMURA, Tetsuya YAMADA,
Kazuaki MURAKAMI and Hiroto YASUURA

Department of Information Systems
Interdisciplinary Graduate School of Engineering Sciences
Kyushu University
Kasuga-shi, Fukuoka 816 Japan

E-mail: {iwaihara, yamaga, tnakagaw, kunisada, saitoh, nagaura, ike,
nakamura, yamada, murakami, yasuura}@is.kyushu-u.ac.jp

Abstract　We report our design experience and design process of the developing project of QP–DLX (Kyushu University Education-Purpose DLX Microprocessor), which is a 32-bit RISC microprocessor especially suited for educating software/hardware courses. We describe design results on the hardware description language SFL, our design environments, usage of QP–DLX design data as a CAD benchmark set, and our plan for a microprocessor design course using QP–DLX.

英文 key words　Educational microprocessor, logic synthesis, hardware description language, instruction pipeline

1 はじめに

我が国の大学教育における集積回路設計教育について，さまざまな努力が各大学で行なわれている．しかし，その多くは講義のみによるもので，実験や実習を伴うものは極めて少ないのが現状である．しかも，ソフトウェアによる実現部分と集積回路技術によってハードウェアとして実現する部分を総合的に考える，システム設計側に立った集積回路設計という立場で組まれた実験カリキュラムはほとんど無いと思われる．現在の我が国の大学における予算や人員の制約から考えて，システム設計側の利用者に対するサービスとして集積回路を製造するような組織を大学内に持つことは極めて難しいと考えられる．できれば，安定した産業界のASIC技術を利用して，教育が行なえることが望ましい．

米国においては1980年代初頭から，システム設計の立場からの集積回路設計教育が進められており，MOSIS等の組織を利用して学生実験で集積回路製作が体験可能となっている．また，ヨーロッパにおいても，仏ES2社等の協力の下，仏IMAGを中心としたEUROCHIPプロジェクト等により，大学の教育研究目的で集積回路の作製が可能な状況が作られている．我が国でも，大学等の教育機関における集積回路設計・製作演習の必要性がようやく認識されており，教育機関のために集積回路を製造する組織の設立も具体化しつつある．

１０万素子以上の回路が比較的安定に製造でき，自動レイアウトや論理合成のツールが完備してきた現在，ソフトウェアとハードウェアを統合してシステム設計を進めることのできる人材の育成が重要となっている．

大学の教育においては，OSやコンパイラなどのソフトウェアと，計算機アーキテクチャ，論理回路設計，集積回路設計などのハードウェアの密接な技術的つながりへの理解を深めることのできる講義／実験に乏しいのが現状である．そのため，ソフトウェア／ハードウェアの両面に渡る技術分野を概観できる一貫教育カリキュラムとして，システム設計を通したソフトウェア／ハードウェア統合型実験／演習カリキュラムの開発が望まれる．

以上の状況に鑑み，我々は，コンパイラやOSなどソフトウェアから，計算機アーキテクチャ，論理回路設計，集積回路設計・製造に至るまでの一貫した教育の共通の教材として用いることができる**『計算機工学の一貫教育』**用マイクロプロセッサ開発プロジェクトを進めている[9][10][11].

本稿では，現在開発中のマイクロプロセッサQP–DLX (Kyushu University Education-Purpose DLX Microprocessor)について，ハードウェア記述言語SFLによる設計を終えたためその設計事例を報告し，教育機関におけるマイクロプロセッサ設計環境について考察する．

まず，2章で，QP–DLXプロジェクト全体の方針を述べる．そして，3章で，アーキテクチャとして採択した*DLX*についてその仕様を与え，教育用として追加した観測機能を紹介する．4章では，ハードウェア記述言語による設計過程および設計環境を述べ，5章ではQP–DLXに対して行なった設計検証について報告する．6章では，QP–DLXによるマイクロプロセッサ設計演習の構想を述べ，7章で結論を述べる．

2 QP–DLXプロジェクト

2.1 目的

現在我々は，教育用マイクロプロセッサQP–DLX(Kyushu University Education-Purpose DLX Microprocessor)の開発を行なっている．このプロジェクトの目的は，以下の3つからなる．

- ソフトウェアから，計算機アーキテクチャ，論理回路設計，集積回路設計・製造に至る計算機工学全般の講義／演習に用いらることのできる，『計算機工学一貫教育用マイクロプロセッサ』の開発．およびそれを用いたカリキュラムの開発．
- CAD研究用ベンチマークとして公開のできる，実用規模のマイクロプロセッサの設計データの蓄積．
- 大学などの教育・研究機関におけるマイクロプロセッサ開発環境の整備．

以下，本プロジェクトの背景および方針を述べる．

2.2 教育用マイクロプロセッサに対する要求

①計算機アーキテクチャの講義，②当該計算機アーキテクチャに基づいたマイクロプロセッサの設計・製作，そして，③当該マイクロプロセッサ上でのソフトウェア（OS，コンパイラ，アプリケーション，等）の作成，を一貫して１つのアーキテクチャおよびマイクロプロセッサを用いて教育することの効果は絶大である．ソフトウェアからハードウェアの講義／演習まで一貫して利用できる教育用マイクロプロセッサへの要求としては，以下のものがある．

- **計算機アーキテクチャ講義用教材**：本マイクロプロセッサを教材として用いて，現在の計算機アーキテクチャの技術水準（パイプライン処理，RISC，キャッシュ，仮想記憶，等）を反映した計算機アーキテクチャの講義が行えること．
- **ソフトウェア演習用教材**：本マイクロプロセッサをターゲット・マシンとしてプログラミング演習やシステム・プログラム（OS，コンパイラ，等）製作実習が行えること．
- **ハードウェア実験用教材**：本マイクロプロセッサの仕様を基に，論理回路設計実習，さらに，ハードウェア製作が可能なこと．

2.3 教育用マイクロプロセッサの現状

現在の教育現場におけるマイクロプロセッサ利用の現状について述べる．

- **市販マイクロプロセッサの利用**：たとえば，Z80やM68000といった初期の市販マイクロプロセッサを用いて，アセンブリ・プログラミング演習やマイクロコンピュータ・システム製作を行う．安価に実現できるが，アーキテクチャが単純ではないので，マイクロプロセッサそのものをハードウェア実験で設計・製作するのが難しい．計算機アーキテクチャの現在の技術水準から遅れている．
- **仮想マイクロプロセッサのソフトウェア・シミュレータ**：仮想マイクロプロセッサのシミュレータを作成し，専らソフトウェア演習に用いる．教育を目的として設計しているので，本質的でない仕様に煩わされない．無償でシミュレータの環境を整えられることもある．しかし，実体のない仮想マイクロプロセッサのため，ハードウェア実験の教材としては利用できない．
- **オリジナル実マイクロプロセッサ**：教育用に実際にマイクロプロセッサを製作する．これには，たとえば，京都大学や京都高度技術研究所が中心となって開発した「計算機工学・集積回路工学教育用マイクロプロセッサ」KUE-CHIP（1989年）[3]，KUE-CHIP2（1992年）[5]や，九州工業大学が開発した「再構成可能な論理LSIを用いた教育用マイクロプロセッサ」KITE（1992年）[7]がある．これらは，教育を目的として設計しているので，本質的でない仕様を含まない．極めて単純なアーキテクチャとなっているので，環境さえ整っていれば，実際にハードウェア実験で当該マイクロプロセッサそのものを設計・製作できる．しかし，語長やアドレス長の制限がきつく，かつ，割込みをサポートしていないので，ソフトウェア演習用教材としては使いにくい．計算機アーキテクチャの現在の技術水準（特に，パイプライン処理，キャッシュ，仮想記憶）が反映されていない．

以上，現在教育に用いられているマイクロプロセッサに対する問題点として，(1)現在の技術水準を反映していない，(2)ソフトウェア実験に必要な割込み処理などの機能を持ち，しかもハードウェア実験で設計可能な，設計データが公開されているマイクロプロセッサがない，という2点にまとめられる．

2.4 CADベンチマークに対する要求

我々はKUE–CHIP, KUE–CHIP2の設計で得られた設計データをベンチマーク・データとして公開している．またKUE–CHIPの高位記述はVHDL, Verilog HDL, UDL/IやSFLなどのハードウェア記述言語(HDL)に翻訳され，これらの言語の比較に用いられている[6]．これらの経験を通して，我々は実用的かつ統合

的なベンチマーク・セットの必要性を認識した．統合的なベンチマーク・セットは，実用的な回路規模を持ち，以下の全ての種類の設計データを含むべきであろう．

- 高位記述 (レジスタ転送レベル)
- 論理回路
- レイアウト・データ
- テスト・パターン
- その他

レイアウトの情報を用いた論理合成を行なうツールや，高位記述からの情報によりレイアウトのフロアプランを作成するツールなど，新らしい CAD アルゴリズムに対応するためには統合型ベンチマークが欠かせないが，現在それに見合うものは見当たらない．さらに，統合型ベンチマークとしては，以下の条件を満すべきである．

- 設計データはバグを含まず，一貫性が取れていること．
- 回路の用途が特殊すぎないこと．
- 各設計レベル間の関連づけがよくなされていること．

これらの条件のためには，実際に製造された回路であることが望ましい．さらに，ベンチマークとして公開できるためには，知的所有権問題も解決されなければならない．

2.5 QP-DLX プロジェクトの方針

[*DLX* アーキテクチャの採用] 単純，かつ，現在の技術水準（RISC，32 ビット語長，パイプライン処理，割込み，キャッシュ，仮想記憶，等）を反映した計算機アーキテクチャとする．このため命令セット・アーキテクチャおよび命令パイプライン構造は，計算機アーキテクチャの教科書として米国で広く利用されている文献 [8] の *DLX* に基づくことにした．

[教育のための機能] ソフトウェア演習に必要な割込みやブレイクポイント機能を単純な方式で実装する．また，マイクロプロセッサ内部の信号をソフトウェアの介入なしに観測する機能，および 1 クロックサイクルごとに命令の実行／停止を行なう機能からなる観測機能を備える．従来のアーキテクチャの講義では内部動作は原理的な説明にとどまりがちであったのに対し，観測機能により実際に命令がパイプラインを流れる様子や，パイプラインで生じるハザードが解決される様子の観察が可能になり，学生のアーキテクチャ動作原理の理解に役立つものと思われる．また論理設計においては，学生に大まかなプロセッサ仕様のみを与えておき，QP-DLX の観測を通して細部の制御論理を推測させる方法がとれる．

[QP-DLX の設計データの公開] QP-DLX の設計ドキュメントおよび設計データを統合型 CAD ベンチマークとして広く公開し，論理合成や設計検証，レイアウトのベンチマークとして利用してもらう．QP-DLX による統合型ベンチマークは，以下の点で優れている．

- *DLX* アーキテクチャはよく知られた RISC 型マイクロプロセッサ・アーキテクチャであり，現実的な回路例である．
- 実際的な回路規模を持ち，様々な種類の回路を含む．
- 仕様記述，ハードウェア記述言語からテストデータに至るまで，設計階層の全てのレベルのデータがそろっている．
- QP-DLX プロジェクトにより実際に動作する回路が製造される．

我々は QP-DLX プロジェクトで得られた以下の設計資産を積極的に公開していく予定である．

- 設計ドキュメント
- SFL ソース・コード (VHDL 等への翻訳も計画している)
- EDIF ネットリスト
- テスト・パターン

[研究・教育機関における LSI 設計開発体制の整備]
大学あるいは高等専門学校の演習／実験の教材はややもすると時代遅れの技術水準となりやすいため，時代に合せた教材の更新は不可欠である．近年，教育環境での高性能ワークステーションの導入が進み，高機能 CAD の導入も可能となってきている．またハードウェア記述言語や自動論理合成・自動レイアウト機能などの最新の技術を駆使することにより，QP-DLX 程度の LSI の設計が学生演習で可能になりつつある．実際にこれらの技術を学生が体験してみることにより，集積回路工学への理解が大きく進むであろう．

今回の QP-DLX の設計では，将来教育現場で教材として利用されることを考慮し，大学で構築できる設計環境のもとでの，最大規模のマイクロプロセッサの開発を意図している．そのため，大学における標準的なハードウェア環境と人員構成のもとで，教材として導入できる CAD システムを用いた設計を行なう．

QP-DLX の設計データをもとに，論理回路工学および集積回路工学の設計演習のカリキュラムを作成し，広く配布する．

3 QP-DLX のアーキテクチャ仕様

3.1 *DLX* アーキテクチャ

DLX は，計算機アーキテクチャの教科書として米国で広く利用されている J. L. Hennessy and D. A. Patterson: *Computer Architecture: A Quantitative Approach*[8] の中で教材として用いられているアーキテクチャである．*DLX* は，今日のアーキテクチャの主流である RISC アーキテクチャに基づいており，RISC 型商用マイクロプロセッサの多くを平均化したようなアーキテクチャとなっている．

QP-DLX は *DLX* のアーキテクチャを採用しており，これに観測機能などの新たな機能を追加している．また，回路規模等の制限により，浮動小数点関連命令および乗算・除算命令は実装していない．文献 [8] では，*DLX* の基本仕様を定義するにとどめており，オペコード割当てなどの細部の仕様や割込み仕様は与えられてないため，これらの仕様を独自に定めている．以下，QP-DLX の仕様の概略を述べる．

- 基本語長：32 ビット
- 汎用レジスタ：32 ビット汎用レジスタ R0–R31
- 制御系レジスタ：
  - プログラム・カウンタ (PC)
  - ステータス・レジスタ (SR)
    スーパバイザ・フラグ，旧スーパバイザ・フラグ，割込み許可フラグ，旧割込み許可フラグ，割込みベクトル (4 ビット)，命令ブレイク許可フラグ，データブレイク許可フラグからなる．
  - 割込みアドレス・レジスタ (IAR): 割込まれた命令のアドレスを格納．
  - 命令／データ・アドレス・ブレイクポイント・レジスタ (BPI，BPD)：アドレス・ブレイク機能 (追加仕様) のためのブレイクポイントを保持．
- 命令セット：データ転送，算術論理演算，分岐／ジャンプ，浮動小数点演算の 4 種類に分類される．
- 命令形式：すべての命令は固定長かつ単一長，命令形式は I 形式，J 形式，R 形式の 3 種類である．3 アドレス方式であり，演算はレジスタ–レジスタ間およびレジスタ–即値間のみ可能である (ロード／ストア・アーキテクチャ)．
- メモリ・アドレッシング：ビッグ・エンディアン方式．バイト，ハーフワード，ワード単位のアクセスが可能．アドレッシング・モードは，ベース相対および PC 相対（ジャンプ命令のみ）が可能．

3.2 命令パイプライン

文献 [8] に従い，次の 5 ステージから成る命令パイプライン構成を採る．図 1に QP–DLX のブロック図を示す．

① IF：命令フェッチ

② ID：命令デコード，レジスタ・フェッチ，および分岐先アドレス生成

③ EX：実行，および，実効アドレス生成

④ MEM：メモリ・アクセス

⑤ WB：書込み

各ステージは 1 クロックごとに動作を完了し，次のステージへデータを渡す．渡されるデータとして，命令 (IR:32bit)，命令アドレス (PC:32bit)，割込みステータス・ベクトル (ISV:4bit) などがある．命令アドレスは PC 相対番地計算のベース，および割込みアドレスを与える．ISV は各ステージで発生する内部割込みイベントを記録する．MEM ステージから出力される ISV の内容を Controller が調べることにより，内部割込みイベントが検出される．

DLX では，RAW（Read After Write）データ・ハザード，制御ハザード，および，構造ハザードが生じ得る．ただし，QP–DLX ではマルチサイクル命令を実装していないので，構造ハザードは生じ得ない．文献 [8] に従い，RAW データ・ハザードおよび制御ハザードに対して，以下のように対処する．

[フォワーディング] 後続 3 命令に対してフォワーディングを行なうため，図 1に示すように EX，MEM, WB の各ステージの出力 EX_C, C_LMD, WB_DATA を ID ステージに送る．

[分岐] 文献 [8] では分岐先アドレス BTA の生成を EX ステージで行ない，PC の置換を MEM ステージで行なう方法と，その改良版として，ID ステージで分岐先アドレス生成と PC 置換の両方を行なう方法が示してあるが，QP–DLX では後者を採用している．

3.3 観測機能

観測機能により，ソフトウェアの介在なしでのプロセッサ内部動作の観測が可能になる．観測機能は以下の 4 つからなる．

- 全てのユーザ可視のレジスタ，命令パイプラインの各ステージでラッチされている値，および信号線を命令／データ用のバスとは独立に設けられた観測用のバス OB を通して読み出す．全てのレジスタには書き込みも可能である．
- 外部からの Start ／ Stop スイッチ信号によりプロセッサを動作／停止させる (動作／停止機能)．
- 停止状態から Step 信号により 1 サイクルの命令パイプライン実行後，停止状態に戻る (シングル・ステップ機能)．
- 割込み発生時に通常の割込み処理を行なう代りに，停止状態に遷移する (割込み観測機能)．

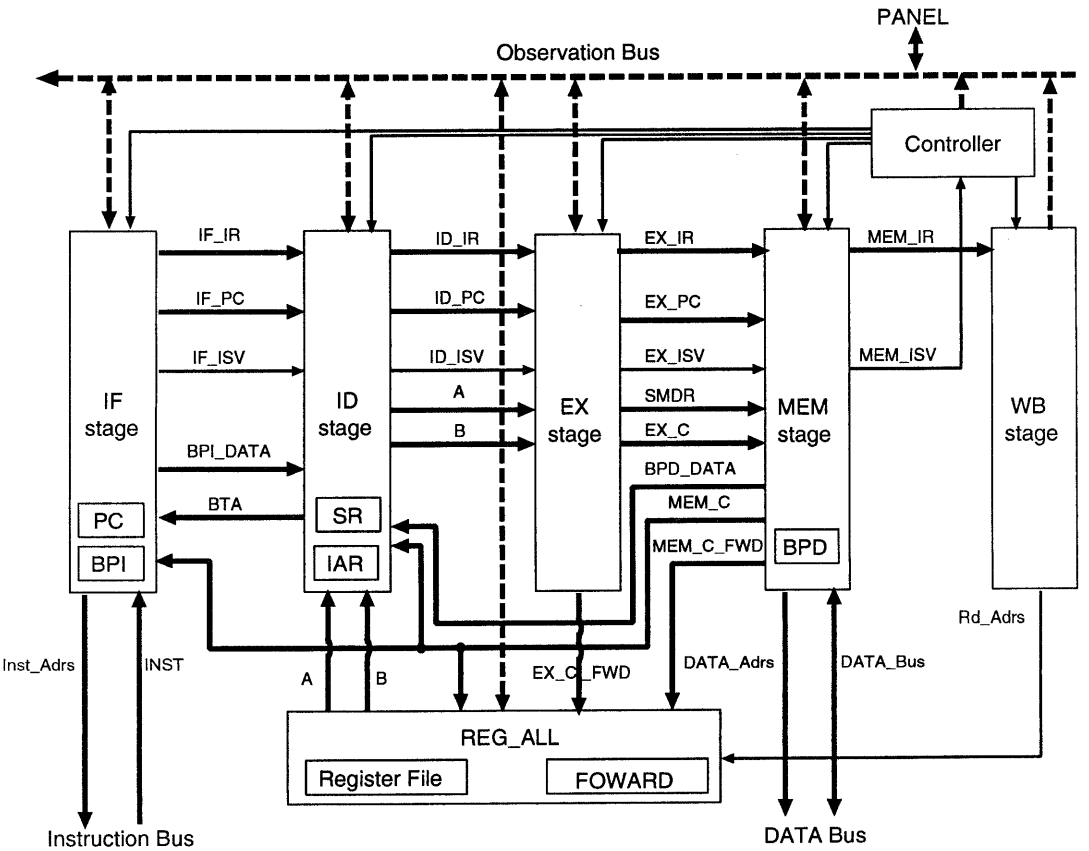

図 1: QP–DLX のブロック図

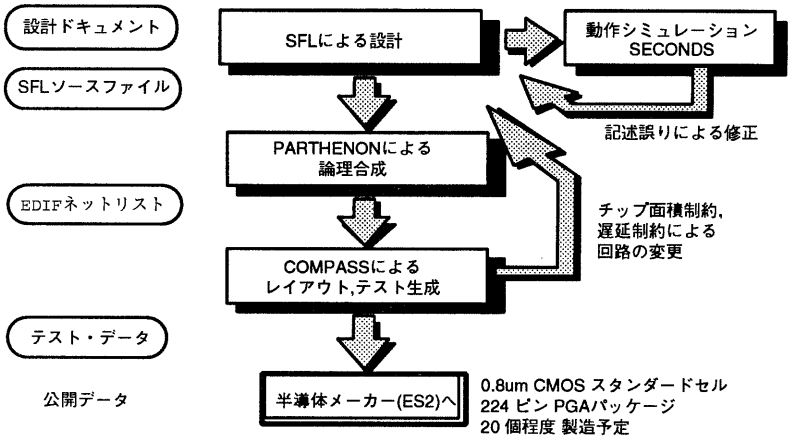

図 2: 開発環境および公開する設計データ

4 開発過程

4.1 CAD ツール

図 2に今回の設計における設計フロー，および用いた CAD ツール群，公開予定の設計データを示す．チップのファブリケータである ES2 社は，教育機関向け LSI 製造サービスを行なっており，KUE-CHIP, KUE-CHIP2 での製造も委託している．今回の設計のように比較的大規模な LSI を少量 (1 ウェファ程度) 製造するサービスのあるファブリケータは他に見当たらなっかったので，今回も ES2 社に委託することにした．レイアウト・ツールである COMPASS は，ES2 社との設計データ受渡しの実績があり，また ES2 社の 0.8 ミクロンの互換ライブラリを有することから使用することにした．次節で論理合成ツール PARTHENON について述べるが，COMPASS と PARTHENON は EDIF 形式のネットリストを介しての接続例があり，実績が豊富である．

4.2 PARTHENON

PARTHENON は NTT で開発された論理合成システムであり [2]，ハードウェア記述言語として SFL を用いている．SFL は単相クロックの同期回路を設計するために特化したハードウェア言語であり，非同期回路やアナログ回路まで設計できる言語に比べると言語構造が単純になっており，また C 言語に似た記法を用いているため，C によるプログラミングの経験者にとっては習得しやすい言語である．PARTHENON は国内の教育機関向けに無償貸与が行なわれているため，大学等への導入は容易である．

PARTHENON は SFL の機能レベルシミュレータ seconds，論理合成ツール sfl_exp, 実部品へのマッピングおよび最適化を行なう opt_map などのツール群からなる．

4.3 人員構成

本プロジェクトで QP-DLX チップ開発の人員構成を表 1に示す．実際に設計に携わったのは修士課程の学生 5 人，学部学生 1 人，教職員 2 人 (うち 1 人は途中まで) である．何人かの学生は以前に CAD ツールの使用経験があるが，本格的に LSI 設計を行なうのは全員が初めてである．SFL 設計担当のうち 1 人とレイアウト担当者は作業量が多いため設計に専念しているが，他の学生は自分の研究も並行して行なっている．

4.4 ツールの習熟

ハードウェア記述言語 SFL の習得については，個人差はあるが 1ヶ月程度要している．また身近に SFL の熟練者がいなかったため，試行錯誤的な言語理解が必要であった．設計作業を通して言語のセマンティクスを理解したり，各個人が独自の書法を身につけている．ハードウェア記述言語の普及のためには，初心者のための入門書はもとより，複雑な書法のプログラム例集の整備などによる設計文化の蓄積が必要であろう．

レイアウトに関しては扱うツールの規模が大きく，マニュアルの読解だけでは不充分であったため，担当者はデザイン・ハウスに QP-DLX の設計データを持ち込んで指導を受けている．

4.5 設計上の制約

1 つの LSI に搭載できる機能は，製作可能な LSI の規模により大きな制約を受ける．ES2 社は教育・研究用プロトタイプ製造サービスとして，1 ウェハに載る個数の LSI を製品テスト無しという条件で製作している．このサービスでは最大ピン数がピングリッドアレイで 224 ピンとなっている．チップ面積に特に制限はないが，主に経済的な理由で設計可能な面積が決まる．

担当	人数	備考	作業時間
SFL による設計	3	仕様決定，設計ドキュメント作成も含む	750
機能レベル検証	5	うち 2 人は SFL 設計者と重複	150
テスト生成	1	論理シミュレーション，ネットリスト変換も担当	165 (進行中)
レイアウト	1	ブロック配置プラン，各ブロックの自動レイアウト，パッドの配置	280 (進行中)
ソフトウェア・ツール作成	1	アセンブラ，コンパイラや機能検証のためのツールの整備	60

表 1: 人員構成

今回は0.8ミクロンルールで80mm^{2}以下という試算になり，搭載可能なゲート数は約4万ゲートとなった．この制約のもとに搭載する機能を取捨選択した．

4.6 ハードウェア記述言語による設計形態

HDLによる設計法として，以下の二形態がある．

① 演算回路やレジスタ・ファイルは過去の設計資産を活用し，HDLにはそれらの資産を機能回路と呼ばれるブラックボックスとして取り込む．HDLでは主に制御系の記述のみ行なう．

② 演算回路等までHDLでサブモジュールとして記述し，論理合成まで行なうフル・シンセシス．

一般に①の機能回路を駆使することにより，設計時間を大幅に短縮できるのであるが，今回の設計では，レイアウト済みの設計資産など，過去の蓄積のない状態から開始すること，および将来論理合成能力のベンチマークとして，設計データを公開することから，②のフル・シンセシスの立場を採った．ただしレジスタ・ファイルのみはゲート数が多いため，レイアウト・ツールのライブラリのものを用いている．

HDLによる設計ではグループによる協調設計，あるいは1人による単独設計の2方式が考えられる．上述①のように機能回路を多用できる状況ならば，HDLの記述量は少なく，QP-DLX程度のプロセッサならば1人の設計者で十分であろう．しかし，以下の理由で作業量が多いため，3人による協調設計を採った．

- 観測機能や割込み処理方式など処理方式の協議・決定．
- フル・シンセシスを行なうためのライブラリの整備．
- 教材用およびCADベンチマーク用としての，各モジュールごとの詳細な設計ドキュメントの作成．

各設計者で2–3のモジュールを担当し，各人の設計量が同程度になるようにした．

4.7 モジュール分割

機能的にまとまった回路をサブモジュールとして，モジュール階層を構築していくのはHDLでも共通の設計手法であるが，HDLにおけるモジュール分割には以下のような特徴がある．

① 生成される回路を考慮しなければ，比較的自由にモジュール分割を設定することができる．その場合，設計者に最も都合のよいモジュール分割が可能である．

② 一般にモジュール分割数を大きくすると，入出力端子宣言文が増加し，ソースコード全体の見通しが悪くなり，また設計変更・保守の手間が増大するため，不必要なモジュール分割は避けるべきである．

③ 協調設計において，記述とモジュール・テストが並行して行なわれる場合，モジュール単体でテストできるよう，各モジュールが完結した記述である必要がある．

④ HDLにおけるモジュール階層が，下位のレイアウト・ツールに引き継がれる．レイアウトにおいては，モジュール階層に基づいて階層レイアウトが行なわれるため，モジュール間の入出力がデータパスを定め，それが配線面積および配線遅延に大きな影響を及ぼす．従って，HDLにおけるモジュール階層は，レイアウトを考慮したものでなければならない．

⑤ ところが，①の機能重視によるモジュール分割と，④のレイアウト重視によるモジュール分割は一般に異なる．例えば，パイプラインを流れる命令，命令格納アドレス，レジスタ値等は，同期して各ステージを遷移するため，各ステージをモジュールにすれば，機能主体の記述しやすい分割になる．しかしレイアウト重視では，命令，命令格納アドレス，レジスタ値をそれぞれ別モジュールにした方が配線量が小さくなる．

PARTHENONでは，論理最適化ツールopt_mapにより，階層を取り払いフラットにすることや，親子の関係にあるモジュールを並列の関係にするなど，ある程度モジュール階層を操作することができる．しかし1つのモジュールを2つに分割にするなどのより複雑な操作ができないため，①の機能重視によるモジュール分割から④のレイアウト重視によるモジュール分割への移行を自由にはできない．そのため，HDL設計の段階からレイアウトを意識したモジュール分割が必要である．

4.8 設計の過程

QP-DLXの設計においては，教育用であることを念頭において，以下の方針に従った．

- 学生実験に必要な最小限の機能に限る．
- 1つの機能の実装方式はできるだけ単純なものにする．複雑な処理方式は教育用として相応しくない．
- 上記方針のためには，性能がある程度犠牲になってもかまわない．
- 性能向上可能な部分については，設計演習の課題として残しておく．

以下，設計を進めて行った過程で削減した機能，改良の可能な処理方式，設計に要した工数を述べる．

[演算ユニット] ソフトウェア実験では，割込み処理機能や特権モードが必要であり，ハードウェア実験では，観測機能が必要である．浮動小数点演算および整数除算のユニットは今回の設計制約上，搭載できなかった．さらに機能を削減する必要があったため，整数乗算ユニットかあるいは観測機能の一部を削減することを検討したが，観測機能の方が教材として重要であったため，整数乗算ユニットを搭載しなかった．将来の設計演習においては，シフト加算命令の追加による乗算の実現を課題とすることができる．

[割込み処理] 一般にパイプライン・プロセッサにおいて割込みが発生した場合，割込まれた命令から実行再開されることを保証する「正確な割込み」を実現するためには，かなりの複雑な機構を必要とする．QP-DLXでは，パイプライン後段のWBステージにおいて，流れてきた命令が前段において内部割込みを生じていないか，あるいは外部割込みがペンディングしていないかを調べている．この方法は単純であるが「正確な割込み」を保証する．この方法の欠点は，内部割込みが生じてからWBステージで認識されるまで，最大4クロックの遅延を生じることである．

[分岐] 文献[8]では分岐先アドレスBTAの生成をEXステージで行ない，PCの置換をMEMステージで行なう方法と，その改良版として，IDステージで分岐先アドレス生成とPC置換の両方を行なう方法(early branch)が示してある．当初，前者の方法で設計を進めていたが，*DLX*の命令セットは，early branchを容易に実現できるように設計されていること(例：単純な条件判断)，およびハードウェア量のオーバーヘッドとしては，IDステージでの分岐アドレス生成用加算器程度であることから，early branch方式に途中で設計変更した．

ただし，分岐遅延スロット(分岐命令の直後の命令)を条件成立／不成立にかかわらず実行するという遅延分岐については，割込み処理方式を複雑にする(例えば，分岐遅延スロットで割込みが生じた場合)．そのため，遅延分岐は実現しておらず，分岐遅延スロットの命令は分岐不成立時のみ実行される．

[設計工数] HDLによる設計，設計検証，ドキュメント作成，ミーティングを含めて，要した設計工数は，約1200時間・人である．また，CADツールの学習に始まり，HDLの設計および検証を終えるまでに約6ヶ月を要している．

4.9 論理合成結果

各モジュールの記述量，主な構成ユニット，およびPARTHENON v2.1による論理合成結果を表2に示す．ここで記述量とはSFLによるソース・プログラムの行数であり，コメント行も含んでいる．

モジュール	SFL の記述量 (行) コメントを含む	主な構成要素 PR: パイプライン・レジスタ (32 ビット) CR: 制御系レジスタ (32 ビット)	ゲート数	サブモジュール数 32 ビット演算器内の 階層は数えない
IF ステージ	464	PR × 2, CR × 2, インクリメンタ	2692	3
ID ステージ	1080	PR × 4, CR × 2, 加算器	4890	6
EX ステージ	1229	PR × 4, ALU, バレルシフタ	5696	5
MEM ステージ	597	PR × 2, CR × 1	4173	5
WB ステージ	263		242	2
レジスタファイル	432	32bit × 31	15541	3
制御部	250		366	0
QP-DLX	4863		40202	31

表 2: 各モジュールの設計結果

PARTHENON では，sfl_exp で論理合成を行ない，opt_map において不要な信号線の削除等による最適化，およびセル・ライブラリの遅延や負荷容量などの情報をもとに，仮想的な論理素子を実際のセルにマッピングする作業が行なわれる．その際，与えられたクロック周波数などの制約条件に対し，遅延時間を満さないパスに対しては，セルの置き換えにより高速化が行なわれる．

今回の設計では，最大クロック周波数が 15MHZ となった．

論理合成に要した時間は約 30 分，opt_map による最適化は対話的作業であるが，4 時間ほど要している．

5 設計検証

マイクロプロセッサの設計に誤りがないかを検証する設計検証は，設計工程でも重要な部分を占める．PARTHENON には，機能レベルシミュレータ seconds があり，そのソースコードレベルのシミュレーションの結果により検証することができる．ソースコードレベルでは，以下の検証を行なった．

- 人手によるテスト命令列の作成および確認 (数千ステップ程度)
  各命令，内部割込みおよび観測機能について，ソースファイルを目視し，誤りを発生しやすい箇所について，重点的にテストする．

- 命令セット・シミュレータとの実行結果比較
  DLX の命令セットを解釈・実行するソフトウェア・シミュレータにおいて，適当なテスト・プログラムを実行させ，その実行結果を seconds による SFL のシミュレーション結果と比較する．テスト・プログラムについては，以下の 2 種類を用いた．
  - 各命令 (10 ほどのグループにまとめている) が各ステージに入る組合わせ，すべてのフォワーディングが起るような組み合わせ等をカバーする命令列 (数万ステップ)
  - 適当な C プログラムをコンパイルし，実行させる (C プログラムで 1000 行程度)

命令セット・シミュレータについては，カリフォルニア大学で開発された *DLX* のシミュレータ dlxsim を 用いている．このソフトウェアは無償で公開されており，ftp により容易に入手できる．このソフトウェアには，GCC をベースにした C コンパイラ，アセンブラも含まれている．今回の検証のためには，dlxsim と seconds のシミュレーション結果を比較するツールを作成した．またアセンブラについては，QP-DLX のオペコードの割当てのために変更を行なっている．C コンパイラについては，アーキテクチャの微妙な違い等で修正する必要があった (例えば，分岐命令の方式やロード・インターロック等を行なう／行なわない)．機能レベルシミュレータの実行速度が 1 秒間で数 10 ステップ程度であった．そのため，dlxsim との比較では，この実行速度により検証可能なステップが制限された．

割込み処理および観測機能の検証については，ソフトウェア・シミュレータを作成していない．再開可能な割込み (外部割込み，ページフォールト等) については，通常のパイプライン動作中に外部から割込み信号を与え，割込み処理を行なわせたものと，割込み信号を与えなかったものの実行結果が一致するかを比較することにより，検証した．

観測パスを通した読み出し，あるいはメモリ・ウェイトによるストールについても，これらの要因がない場合との実行結果比較が一致するかによる検証を行なった．

機能レベルシミュレータ seconds では，1 つの内部端子あるいはレジスタに対し，同時に複数の書込み要求が発生した場合，エラーとして検出される．しかしこのエラーは論理合成の段階では検出されなかったため，機能レベルシミュレータの段階で十分な複数書込みエラーの検証を行なう必要があった．

論理合成結果の検証については，レイアウト・ツール上の論理シミュレータで行なった．その際の検証データは，seconds で用いた命令列のうちの一部を用いている．また，論理回路図をプロットし，論理合成結果を目視で検証している．

6 マイクロプロセッサ設計演習における QP-DLX の利用

今回の設計で得られた QP-DLX の設計データおよび設計ドキュメントをもとに，マイクロプロセッサ設計演習／実験の教材を開発する予定である．QP-DLX による演習は，『計算機工学一貫教育』を目指した，ソフトウェア／アーキテクチャ／論理設計の演習が統合された，システム設計的な内容とする．

マイクロプロセッサ設計演習としては，以下の課題に分けられる．

- ハードウェア記述言語による設計と論理合成
- 命令セットに合せたコンパイラの開発
- レイアウトおよびテスト生成
- 性能評価

ソフトウェア／ハードウェアにわたる技術分野を視野に入れた演習の形態として，以下の例が考えられる．

- 高級言語によるベンチマーク・プログラムを与え，それを実行するシステムを設計する．
- 設計制約として，レイアウト面積やピン制約，クロック周波数の下限等を与える．
- 上記ベンチマークを効率良く実行するために，与えられた命令セットの再設計，あるいは命令の追加を行なう．
- ハードウェア記述言語によりプロセッサの設計を行なう．
- 演算器等の改良によるクロック周波数の向上を図る．
- 増加するハードウェア量に対し，レイアウト面積の制約を満すようにレイアウトの改良を行なう．
- クロック・サイクル数による性能評価を HDL の機能レベルシミュレータで行ない，回路の遅延による時間評価をレイアウト・ツールで行なう．

一般に，演習のカリキュラムを作成する場合は，

- 限られた演習の時間内で行なえる内容とする．
- 単調な作業となる部分は極力減らし，教育効果／時間比を高める．

などの配慮が必要である．QP-DLX 程度の 32 ビット RISC マイクロプロセッサを設計演習の対象とするときには多大な工程を要するために，教材に対する十分な配慮をしなければ時間的制約を満たせないであろう．以下，マイクロプロセッサ設計演習において負荷を減らすための手段を述べる．

- CAD ツールの習得に時間がかかるため，演習に必要なコマンドを選定した簡易マニュアルを作成する．
- HDL は初めて使う学生がほとんどであり，言語のセマンティクスが理解しづらい点が多い．そのため，マイクロプロセッサの記述の模範例を示し，言語に固有な書き方等を示す必要がある．
- 使用できる演算ユニット等を指定して，あらかじめ機能回路として用意しておき，HDL での設計の手間を減らす．
- 論理合成ツールからレイアウト・ツールへのネットリストの変換ユーティリティを作成して，変換作業を自動化する．
- レイアウトでは，1 回のレイアウト時間が長いため，レイアウトの試行錯誤できる回数が限られる．また，ブロック分け，フロアプラン，端子・パッドの配置などの指定に時間が取られるため，あらかじめ階層レイアウトを行なったテンプレートを用意しておき，学生にはそれらの変更によるレイアウトを行なわせる．

時間的制約が厳しい場合には，QP-DLX のうち，一部分のみを設計するという形式が考えられる．その方法として，(1)QP-DLX のいくつかの機能を削減し，その機能の追加を課題とする方法，および (2)QP-DLX にない機能の追加を課題とする方法の 2 者が考えられる．以下，その両者の例を述べる．

① QP-DLX から以下の機能を削減し，それを設計する．

- シフト命令や論理演算命令など，EX ステージの機能ユニットを用いる命令
- フォワーディング回路 (フォワーディング回路を省略した場合，Read After Write ハザードが生じたときのパイプライン・インターロックを行なう機構の追加が必要)
- early branch (ID ステージで分岐処理)
- メモリ・ウェイト
- 割込み処理機構

② QP-DLX にない機能を追加する．

- 乗算等のマルチサイクル命令，およびそれらためのパイプライン・ストールの実装
- ロード・インターロックの実装
- 条件分岐命令の拡張 (2 つの値の比較結果を条件とできるようにする)
- 機能ユニットの追加 (シフト加算等)
- 遅延分岐
- 遅延の小さい割込み処理方式
- マルチプロセッサ構成のための通信機構

我々は，以上の点を考慮して QP-DLX を用いた演習カリキュラムの作成を行なう予定である．

7 あとがき

以上，コンパイラなどのソフトウェアから計算機アーキテクチャ，論理回路設計，集積回路設計に至る**計算機工学一貫教育**用マイクロプロセッサ QP-DLX について，ハードウェア記述言語による設計の過程，設計環境，QP-DLX を用いた計算機工学演習の形態について述べた．今後，マイクロプロセッサのレイアウト設計完了，およびボードの製作，設計データの整備・公開を順次行なってゆく予定である．

QP-DLX の SFL ソース・コードは，すでに形式的検証の対象として，京都大学，奈良先端科学技術大学，カーネギーメロン大学の研究グループで用いられている．また，海外からも設計データに関する問い合わせが多数来ているため，設計ドキュメントの英語への翻訳も順次進める予定である．

本プロジェクトをより良いものとするため，皆様からのご意見，ご指導が頂ければ幸いである．また，集積回路設計・製造の教育に関しては，大学だけの力ではその実現が非常に困難である．関係省庁，企業等の皆様にぜひご協力頂きたいと心から願っている次第である．

謝辞

本研究の推進に当たっては，(株) テクノアライアンスおよび ES2 社，LSI システムズ (株)，(株) ソリトンシステムズおよび Compass Design Automation 社，NTT データ通信 (株) および日本電信電話 (株)NTT コミュニケーション科学研究所，横河ヒューレットパッカード (株) および (株)YHP システム技術研究所の各社に CAD ツールの提供，設計コンサルタントなどでご協力いただいております．この場を借りて御礼申し上げます．

日頃ご討論頂く九州大学 大学院総合理工学研究科 安浦研究室の諸氏に感謝致します．

なお，本研究は一部，文部省科学研究費補助金 試験研究 (B)「計算機工学・集積回路工学教育研究用マイクロプロセッサの開発」(04555079) による．

参考文献

[1] 江刺正喜, “大学での LSI 製作と教育,” 電子情報通信学会誌, vol.68, no.1, pp.50–52, 1985 年 1 月.

[2] “PARTHENON Reference Manual,” NTT データ通信株式会社, 1989/1990.

[3] 神原, 安浦, “教育用マイクロコンピュータ KUE-CHIP の開発とその応用 — 集積回路技術を利用した情報工学実験 —,” 情報処理, vol.33, no.2, pp.118–127, 1992 年 2 月.

[4] 庄野克房, “大学における LSI 設計教育 — 2 ビットマイクロコンピュータ —,” 電子情報通信学会誌, vol.75, no.5, pp.530–533, 1992 年 5 月.

[5] 越智, 澤田, 岡田, 上嶋, 神原, 濱口, 安浦, “計算機工学・集積回路工学教育用マイクロプロセッサ KUE-CHIP2,” 信学技報, CPSY92–46, ICD92–86, 1992 年 10 月.

[6] 神原弘之, 安浦寛人, Kukkal, P., Kobayashi, H., 野地保, 小栗清, “ハードウェア記述言語の比較,” 情報処理, vol.33, no.11, pp.1269–1283, 1992 年 11 月.

[7] 末吉, 田中, 柴村, “再構成可能な論理 LSI を用いた教育用マイクロプロセッサ: KITE,” 信学技報, CPSY92–47, ICD92–87, 1992 年 10 月.

[8] Hennessy, J. L. and Patterson, D. A., *Computer Architecture: A Quantitative Approach*, Morgan Kaufmann Publishers, Inc., 1990;
富田, 村上, 新實 (訳), ヘネシー&パターソン コンピュータ・アーキテクチャ— 設計・実現・評価の定量的アプローチ —, 日経 BP 社, 1992.

[9] 諸富, 村上, 安浦, “計算機工学一貫教育用ＤＬＸ風マイクロプロセッサの開発構想,” 電子情報通信学会技術研究報告, VLD92–83, 1993 年 1 月.

[10] 諸富, “計算機工学一貫教育用マイクロプロセッサのプロトタイプ設計,” 九州大学工学部情報工学科卒業論文, 1993 年 3 月.

[11] Iwaihara, M., Nakamura, S., Murakami, K. and Yasuura, H., “A DLX Microprocessor for Education and CAD Benchmark,” Proceedings of the Synthesis and Simulation Meeting and International Interchange (SASIMI), pp. 74–83, Oct. 1993.