プロセッサ設計教育の実践と評価

山根 章吾 † 鈴木 裕人 †† 小柳 滋 ††
† 立命館大学大学院理工学研究科 †† 立命館大学理工学部情報理工学部

1 はじめに

コンピュータアーキテクチャを専攻する学生の導入教育としてプロセッサ教育は重要である。また、その理解を深めるため、実際にプロセッサ設計を経験することが特に重要である。そこで、我々の研究室では研究室に配属された3回生全員を対象にして、短期間（一月）で、最低限の機能を備えているプロセッサの作成を行うカリキュラムを考え、それを実践した。本稿ではその内容と評価について述べる。

2 設計演習概要

本演習では3週間で16bitの単一命令を実行できるプロセッサをVerilog HDLで作成し、引き続いて1週間でパイプラインへの拡張を行う。基本目標としては全員がシミュレータ上でプロセッサを完成させることである。演習の進行方法は、週一度の全体に対する講義の後、各自でプロセッサの設計とシミュレータ上でのテストを行う。また、質問等は研究室で受け付け、進捗の確認は週に一度、用意したテストベンチの実行により行った。

2.1 プロセッサ概要

今回設計するプロセッサはFig 1のように5つのモジュールと、3つの記憶装置から成り立っている。基本的にはMIPSを参考にしており、IF Modeuleでは命令の読み込み、DE Moduleでは命令のデコードとレジスタフェッチ、ALU Moduleでは演算、MA Moduleではメモリとの入出力、WB Modeuleではレジスタの書き込みをそれぞれ制御する。また、記憶装置はハーバードアーキテクチャとし、レジスタは8個とする。

2.2 命令セット

命令には算術・論理演算命令、分岐命令、ロード・ストア命令の3種類を用意しており、授業の進度によっては拡張できるだけの余地を残してある。また、命令の種類を示すOPCODEは、先頭2bitで上記の3種類の内一つであることを示し、次の3bitで更に処理の指定を、最後の1bitでレジスタ形式を示す形にしている。レジスタオペランドはrs,rt,rdの3アドレス方式とし、Functは即値命令使用時の拡張部分である。

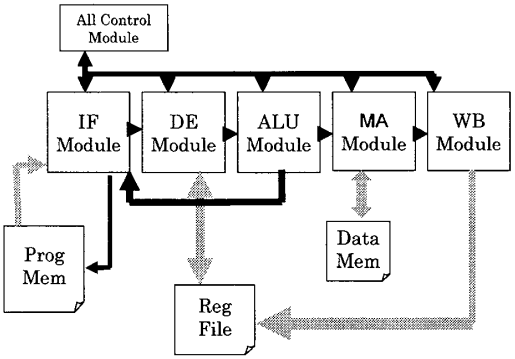

図1: 作成プロセッサのブロック図

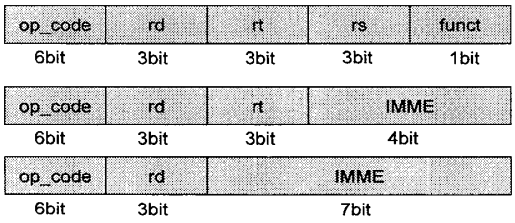

図2: 命令形式

2.3 カリキュラムの内容

第1週目：IF Module周りの作成

第1週では、作成する分量を全4回の内で最も少なくし、作成するプロセッサの全体的な理解、Verilog-HDLの復習、使用するソフトウェアの習熟に時間を割く。第1週では作成するモジュールの機能だけではなく、具体的な内部構成まで言及し作成に取り掛かるまでの負担を軽減している。

第2週目：DE、ALU Module周りの作成

命令セットの理解を中心に作成を行う。演算に必要なデータの取り出しを行うDE Moduleや、指定された演算を行うALU Moduleでは、命令セットの理解が不可欠であり、命令セットに深く関わる2つのモジュールを続けて作成することにより、理解を深めるのが狙いとなっている。また、ここでは第1週とは違い、最終的な入出力にのみ指定項目を絞り、内部構成については指示をせず、自ら考え作成する足がかりとする。

Practice and evaluation of processor design education

† Syogo Yamane
†† Yuto Suzuki
†† Shigeru Oyanagi
Ritsumeikan Graduate School of Science and Engineering (†)
Ritsumeikan University of Information Science and Engineering (††)

第3週目：MA、WB Module周りの作成

データの書き込みを行うことで命令同士につながりを持たせ本格的な処理を行えるようにする。MA Moduleではデータメモリへの書き込み、読み込みを行うモジュールで、どちらの動作も1クロックで行えるように作成する。また以前作成したレジスタファイルの拡張も行う。

第4週目：予備時間、ならびに拡張

進捗が芳しく無かった学生には第3回までの完成を、そうでない学生にはパイプラインプロセッサへの拡張に挑戦してもらった。パイプラインの作成をできるだけ簡易に行うため、パイプラインの機能を3段階に分けて行った。第1段階ではNOP信号を用いることで各モジュールの誤動作を防ぎ、分岐やハザードのない命令の実行を、第2段階では分岐により発生するハザードの解決を、第3段階ではデータハザードの解決を目標にしている。また、ここではパイプラインに対する理解を優先するため、各段階でどのような動作をするのかを例を挙げて解説し、難易度を下げている。

3 評価

	1週目	2週目	3週目	4週目
作成時間	2.73	4.68	4	4.56
デバッグ時間	1.77	1.75	4.4	9.06
完成人数	15	15	15	8
理解度	1.46	1.82	1.324	1.29

表1: 結果の一覧表

全員に毎週の作成時間、デバッグ時間、理解度、感想をアンケートの記入させ、これを集計した。表1に結果を示す。作成時間およびデバッグ時間の単位は時間、完成人数は課題が終了した人数、理解度は4段階評価で各週ごとに複数の項目に対して行った。

理解度の評価基準は以下のようにした。

1. 全体での解説のみで理解できた
2. 個別に教えてもらって理解できた
3. 理解はできなかったが作成できた
4. 作成できなかった

アンケートで特に多かった各週の課題を以下に示す。

第1週：文法忘れ、HW動作の不理解

第2週：デバッグが複雑、命令セットが分かり辛い

第3週：書き込み制御の難しさ

第4週：全体への修正の難しさ

・各週の作成時間について

Table 1に示すように第3週までの基本的なプロセッサ作成については全員が完了しており、また日に1時間前後の学習時間で行えているため、想定した短い期間での作成を行えていると考えられる。また、基本的なプロセッサ部分については全員が作成を完了しており、理解度についてもある程度は満たせていると考えられる。

しかし、第4週のパイプラインについては、作成が完了している生徒は全体の半分に留まり、また作成時間についても約2倍になっていることから、現状のままでは目的に沿わない結果になっている。この解決案としては、パイプライン作成を3段階に分けた中で比較的短時間でできる1,2段階を第4週で、3段階を第5週目に分けることが考えられる。

・理解度について

他の週に比べて第2週目の理解度が低く、アンケートのでも命令セットに対する理解の難しさが挙げられている。

解決案としては、命令セットの解説の際、資料の読み方だけではなく、いくつかの命令の具体的な動作も含めて解説するべきだったと考えている。

・デバッグ時間について

デバッグに関する問題としては、全体の作成時間のうち5割がデバッグ時間で占められており、特にパイプラインについては7割近くがデバッグ時間で占められていることから、大きなネックになっていることがわかる。また相談に来た学生を指導する場合でも、問題箇所の発見に手間取り短時間で指導ができない。

そこで解決案として、デバッグの際の注意すべき部分を第1週の解説に加えることや、担当者用に過去に存在したバグの凡例を渡すなどの手段が考えられる。

・アンケートについて

アンケートからわかるいくつかの問題があるが、今まで述べていない部分ではHDLへの理解度の問題が挙げられる。

これについては、最もよく使われる文法の解説や、ハードウェア特有の並列動作についてなどを解説に加えることでスムーズな進行を行えるものと考える。

4 おわりに

今回の実習では、最低限の機能を備えたプロセッサ作成を一月で全員が行うことができた。解説方法についてや、パイプライン化については課題が残ったものの、目的は果たせていると考える。今後の展望としては、今回見つかった問題を克服し来年に繋げる事と、プロセッサ研究を行う人のための課題として、フォワーディング、分岐予測を組み込むなど、更に踏み込んだカリキュラムを考えていく予定である。