卒 業 論 文

教育用マイクロプロセッサの設計と FPGA ボード上での検証

氏　　名　　：船附　誠弘

学籍番号　　：2210010187-8

指導教員　　：山崎　勝弘　教授

提 出 日　　：2005 年 2 月 21 日

立命館大学 理工学部 情報学科

内容梗概

本論文では、現在のLSI設計の主流であるハードウェア記述言語Verilog-HDLによるマイクロプロセッサの設計、FPGAへの実装を行った。設計対象のアーキテクチャには、京都大学、京都高度技術研究所(ASTEM)、そして本学で共同開発された教育用マルチサイクルプロセッサKUE-CHIP2(Kyoto University Education CHIP)の命令セットを用いている。

本研究では、設計したプロセッサを「SpartanⅡ学習用ボード」のFPGAに実装して動作を確認し、情報学科の情報学実験Ⅲでも用いられているKUE-CHIP2教育用ボ-ドとFPGAボードとの接続を可能にするための回路の設計を行った。これにより教育用ボードからFPGAボードへの操作、観測ができ、設計したマイクロプロセッサと教育用ボード内蔵のKUE-CHIP2の動作の比較が可能になった。

本研究では情報学実験Ⅲの内容を組み合わせることによって、アセンブリプログラミング、プロセッサアーキテクチャの動作、ハードウェア記述言語によるトップダウン設計とCADツールの使用法を総合的に理解することが目的である。

目次

図目次

表目次

1 はじめに

現在、LSIは日常に存在するあらゆる家電製品やパソコン、自動車、通信機器にまで使用されており、もはやLSIなしに私たちの生活は成り立たないといっても決して過言ではない。そして、エレクトロニクス技術の進歩は驚愕すべき速度であり、現在もなお成長を続けている。それは回路規模、集積度においても例外ではなく、世界初のコンピュータで重さが30トンもあるENIACがわずか100グラム程度の現在の携帯電話の計算能力に遠く及ばないという事実だけでも伺うことができる。

LSIに求められる機能は高度化し、実装規模が増大していくにもかかわらず、競争原理から新製品の開発期間はむしろ短期化を迫られ、従来の論理ゲートから設計を進めていくボトムアップ設計では限界が生じてきた。そこで近年ではハードウェア記述言語によるトップダウン設計が主流となっている。ハードウェア記述言語とEDA(Electric Design Automation)ツールの登場により、抽象度の高いレベルの回路の記述ができ、記述された言語から自動的に回路が生成されるようになった。これにより大幅な設計期間の短縮、コストの削減が実現されている。シミュレーションも機能レベルのシミュレーションからゲートレベルのシミュレーションまで上位レベルからの検証が可能なため、比較的初期の段階で設計の誤りを発見、修正ができ、設計の負担を軽減している。

そして1985年には、電気的にプログラミングすることによってその場で自由なロジックが構成でき、しかも、消去、再書き込みが可能と非常に柔軟性の高いプログラマブルデバイスのFPGA(Field Programmable Gate Array)がXilinx社[19]から発表された。ASIC(Application Specific Integrated Circuit)の開発のプロトタイプとしても利用され、なお一層の開発効率の向上に貢献している。

一方、組み込み機器分野を中心とする日本の産業界では、ハードウェアとソフトウェアの両方の知識を有する人材を様々な企業が求めており、大学教育においても、ハード、ソフト両方を理解できる人材の育成が必要とされている。本学情報学科における情報学実験の講義では、コンピュータの動作理解を目的にKUE-CHIP2教育用ボードが使用されているが、アセンブリプログラムのみにとどまっている[3]。ハード、ソフト両方の理解に着目すると、必ずしも十分な講義内容とは言えない。アセンブリ言語とプロセッサの動作は非常に緊密な関係を持ち、アセンブリ言語を示すビット列がプロセッサ中でどのような動作を引き起こしているのかを知るには、自らプロセッサを設計することによって、より理解を促すこ

とができるのではないかと考えられる。

以上のようなLSI設計の背景、本学での実験教育の現状を踏まえ、本研究では、ハードウェア記述言語Verilog-HDLによるマイクロプロセッサを設計し、FPGAボードに実装して動作の検証を行なう。そして、情報学実験のアセンブリプログラミングにマイクロプロセッサの設計を組み込み、ハードウェアとソフトウェアのインターフェースの更なる理解を促す学習形態の実現を目的としている。対象のアーキテクチャは京都大学、京都高度技術研究所(ASTEM)、そして本学で共同開発された教育用マルチサイクルプロセッサKUE-CHIP2(Kyoto University Education CHIP)の命令セットを用いた[1][2]。また、設計したFPGAボードとASTEMで販売されているKUE-CHIP2教育用ボードとの接続を行なった。これにより教育用ボードの豊富なスイッチによる操作や、内部信号を動的に選択して観測できるようになり、教育用ボードのKUE-CHIP2とFPGAの動作の比較をしやすいようにした。

本学では、2004年より、電子情報デザイン学科が新たに理工学部に新設され、ハード、ソフト両方の理解が非常に重要なテーマの一つとされている。情報学実験にプロセッサ設計を組み合わせることによって、アセンブリプログラミングだけでなく、プロセッサアーキテクチャの理解、ハードウェア記述言語とCADツールの習得、トップダウン設計の経験を学習から得ることができるため、新たな教育方法の1つとしての有用性があるものと考えられる。本研究では接続によるFPGAの実装を行なったが、教育現場への現実的な利用を考慮し、1つの教育用FPGAボードを設計する事を今後の課題としている。

本論文の構成として、第2章ではハードウェア記述言語とKUE-CHIP2の概要について述べ、第3章ではVerilog-HDLによるKUE-CHIP2の設計とシミュレーションツールによる論理検証について述べる。第4章ではFPGAボードへの実装と検証及びKUE-CHIP2教育用ボードとの接続について述べる。

2. ハードウェア記述言語によるシステム設計

2.1 ハードウェア記述言語

ハードウェア設計において、長い間設計者は回路図を紙面に書いたり、あるいは回路図エディタで論理素子などを入力することによって設計を行ってきた。しかし、回路の大規模化、複雑化に伴い、従来の方法では設計が困難になってきていた。このような背景の中、現在ではハードウェア記述言語(HDL:Hardware Description Language)と呼ばれる抽象度の高い言語で記述し、その記述からツールによって自動的に回路を生成する設計方法が一般的となってきている。HDLはC言語などのソフトウェア記述の高級言語のような条件分岐や繰り返しが記述でき、抽象度が高く、仕様の変更や再利用も容易であるという特徴を持つ。また、時間の概念があり、遅延の記述が可能なことや、並列動作に対応している点でC/C++などのソフトウェア言語と異なる性質を持っている。図１にハードウェア記述言語の1つであるVerilog-HDLの条件分岐if構文を用いた半加算器の記述例と図2にツールによって自動生成された回路図を示す。

```
module HalfAdder(s,c,a,b);
     output s, c;
     input a, b;
     reg s, c;

     always@(a or b)
       begin
         if((a & b) == 1)
           begin
             s = 0; c = 1;
           end
         else if((a | b) == 1)
           begin
             s = 1; c = 0;
           end
         else if((a | b) == 0)
           begin
             s = 0; c = 0;
           end
         else
           begin
             s = 1'bx; c = 1'bx;
           end
       end
endmodule
```

入出力の宣言

動作の記述

図1：Verilog-HDLによる半加算器の記述例

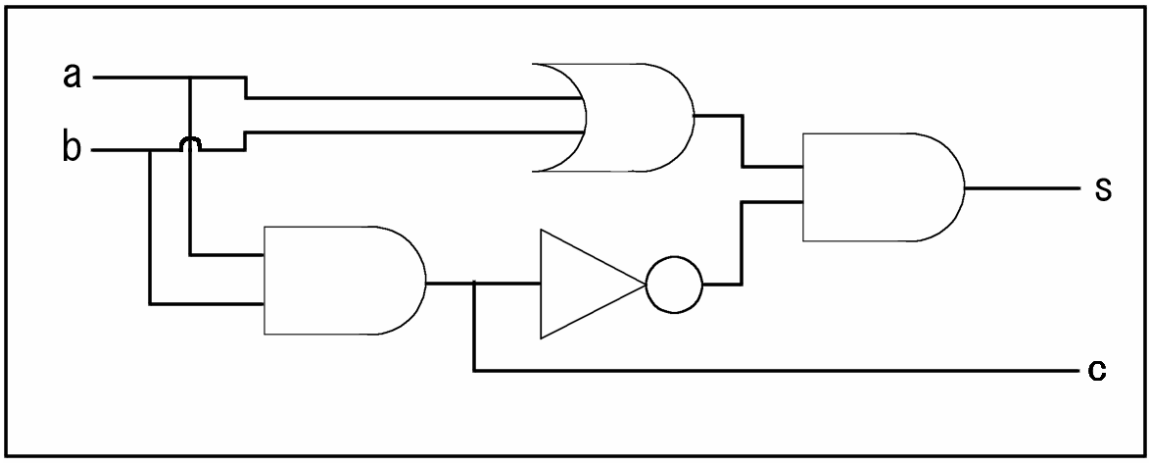

図2：自動生成ツールによって生成された図1の回路図

現在様々なハードウェア記述言語が存在しているが、Verilog-HDLとVHDLが最も代表的であり、産業界で広く用いられている。以下に各言語の特徴を示す。

- Verilog-HDL

Verilog-HDLは米国ゲートウェイ社(現ケイデンス社)で論理シミュレータ用に開発されたHDLである。1995年にIEEE-1364に認定され、標準化された。ライブラリが豊富であり、動作記述や構造記述などの異なるレベルの記述を混在させて設計することができる。また、テストパターンの記述の効率性が高いことも特徴に挙げられる。

- VHDL

VHDL（VHSIC Hardware Description Language）はアメリカ合衆国国防総省のプロジェクトのもとで開発されたHDLである。IEEE Std. 1076-1987 および IEEE Std. 1164-1993 規格として正式に承認され、標準化された。言語仕様が豊富であり、様々な構文が存在する一方、データタイプに厳格であり、言語としては比較的制約が厳しいのが特徴である。

2.2 システム設計

ハードウェア記述言語の登場によって、設計の手法が大きく変化し、従来のボトムアップ設計からトップダウン設計へと主流が移っている。ボトムアップ設計は以前から長い間用いられてきた手法である。システムを構成するモジュール単位で回路図を人手で作成し、モジュール同士を結合しながらシミュレーションを行い、システムを設計する。一方、トップダウン設計は現在主流となりつつある手法で、ハードウェア記述言語による抽象度の高いプログラミング言語で仕様を記述し、論理合成ツールによって自動的に回路を構成する設計手法である。

以下にボトムアップ設計手法とトップダウン設計手法の特徴をそれぞれ示す。[17]

- ボトムアップ設計手法
  - 対象のデバイスやプロセスを開発初期段階で決めておかなければならない
  - 各モジュールが完成しないと、全体の検証が始められない
  - デバッグが困難で、設計が進むにつれ、修正に対するコストが大きくなる
  - 設計の最終段階まで製品の完全な動作検証ができない

- トップダウン設計手法
  - 最終段階までデバイス選択の余地があり、最新のテクノロジが使用できる
  - 開発の初期段階で致命的な欠陥を発見できる
  - 設計の遅いブロックに全体の設計期間が引っ張られない
  - 論理合成により時間が節約でき、設計者は使用の検討や検証に専念できる
  - 設計データを簡単に再利用できる

両者を比較した場合、一般に設計期間や開発コストについては効率が高いという点でトップダウン設計のほうが優位であるといえる。一方ハードウェアの面積効率や動作速度の点について、トップダウン設計は論理合成ツールの能力に依存するため、プログラマの記述によっては非常に資源の使用効率が悪い回路を生成することもあり、従来のボトムアップ設計の方が最適化されているケースも少なくない。しかし、論理合成ツールの能力は徐々に上がっており、回路の複雑化、規模の増大という現在の背景も伴い、今後さらにトップダウンアプローチによる設計が中心になっていくものと思われる。

2.3 FPGA

1985年、米国xilinx社がプログラミングによって様々な論理回路を実現できるLSI、FPGA(Field Progarammable Gate Array)を発表した。現在のトップダウン設計アプローチの土台として、FPGAは多大な影響を持っている。図3にFPGAの基本構造を示す。

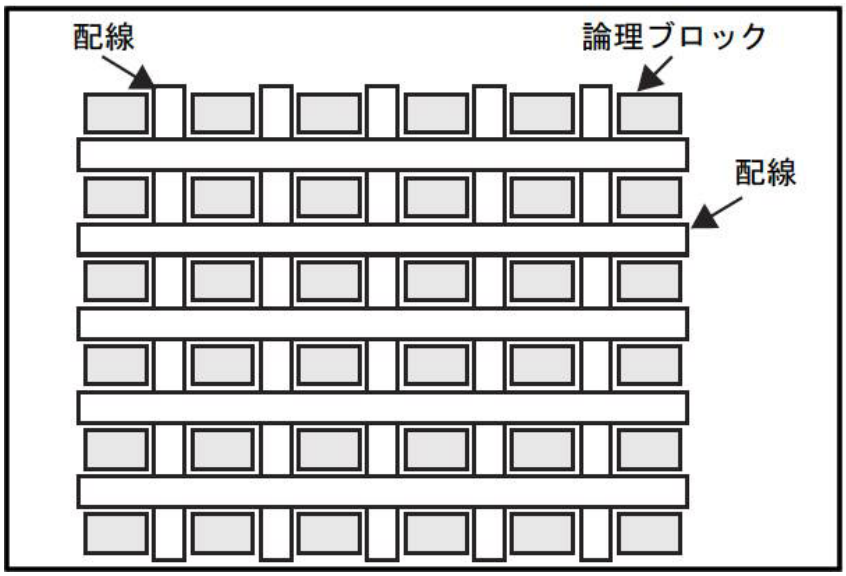

図3：FPGAの基本構造

FPGAは多数の論理ブロックと縦横に伸びた配線領域で構成されており、ブロック同士を配線で繋げることによって回路を構成する。プログラム素子は一般的にSRAMで構成されており、回路の電源を切ったり、新たに回路データをダウンロードすることにより、回路構造を何度も書き換えすることができる。従来のLSIでは書き換えができないため、動作の検証を行なうたびに回路を新しくLSIに焼き付ける必要があり、多大なコストと時間を要していたが、FPGAを検証用のプロトタイプとして用いることによってコストと時間の削減が可能となった。比較的最近まではASICの試作用途や少量生産に限って用いられることが多かったが、回路規模の増大にもかかわらずFPGAチップの単価が下がっていることや、製品出荷後でも重大な欠陥が見つかった場合の修正が可能で、リコールなどのリスクが軽減されるという強みがあり、FPGAの市場は年々増加している。

3. 教育用マイクロプロセッサKUE-CHIP2のアーキテクチャ

3.1 KUE-CHIP2の概要

本研究で設計の対象としたKUE－CHIP２(Kyoto University Education Chip2)は京都高度技術研究所（ASTEM)、京都大学、立命館大学で開発された教育用8ビットマイクロプロセッサであり、非常にシンプルなアーキテクチャ及び命令セットを持つ。また、ASTEMが製造、販売しているのKUE-CHIP2教育用ボード(図4)は国内の大学の計算機教育に数多く採用されており、情報学科の「情報学実験III」においても動作の観測やアセンブリプログラミングに用いられている。特徴としては各命令を３つから最大５つのクロックフェーズに分けて実行するマルチサイクルプロセッサであり、オペランドを指定するアドレッシングモードが4種類と、豊富であることが挙げられる。KUE-CHIP2教育用ボードでは、1クロックフェーズごとの実行を行なうシングルフェーズモードSP、1命令実行のシングルインストラクションモードSI、停止命令まで実行するスタートストップモードSSの3つの実行モードがあり、内部状態の観測性に優れている。

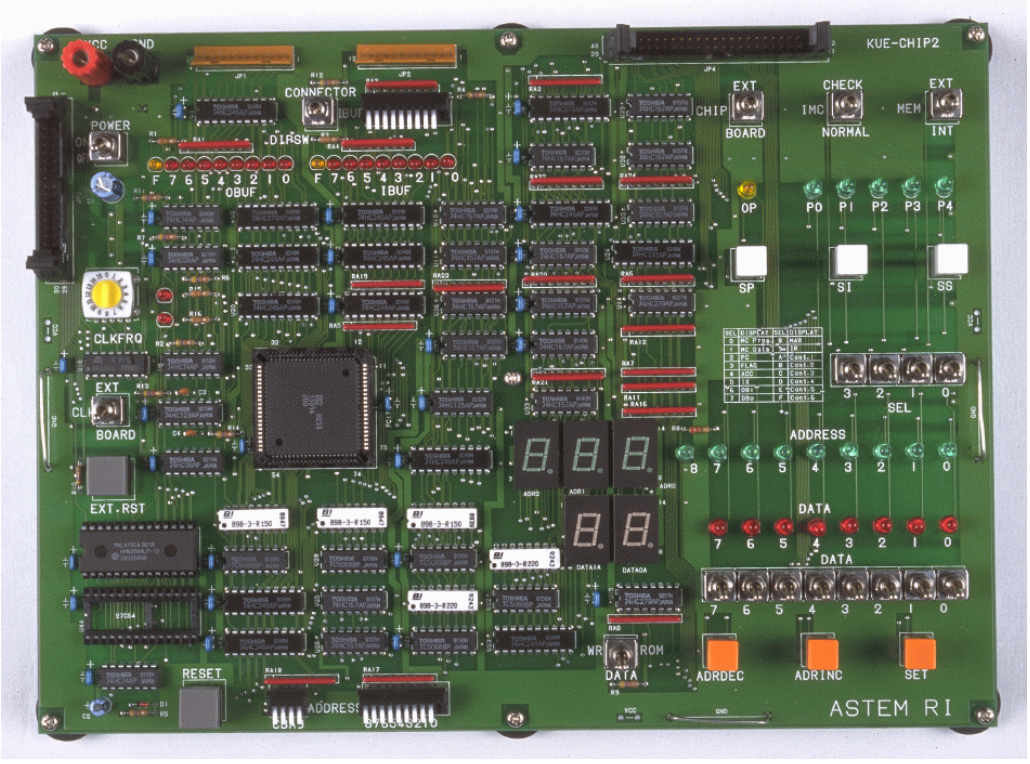

図4：KUE-CHIP2教育用ボード

3.2 KUE-CHIP2の構造

図5に今回設計したKUE-CHIP2のブロック図を示す。

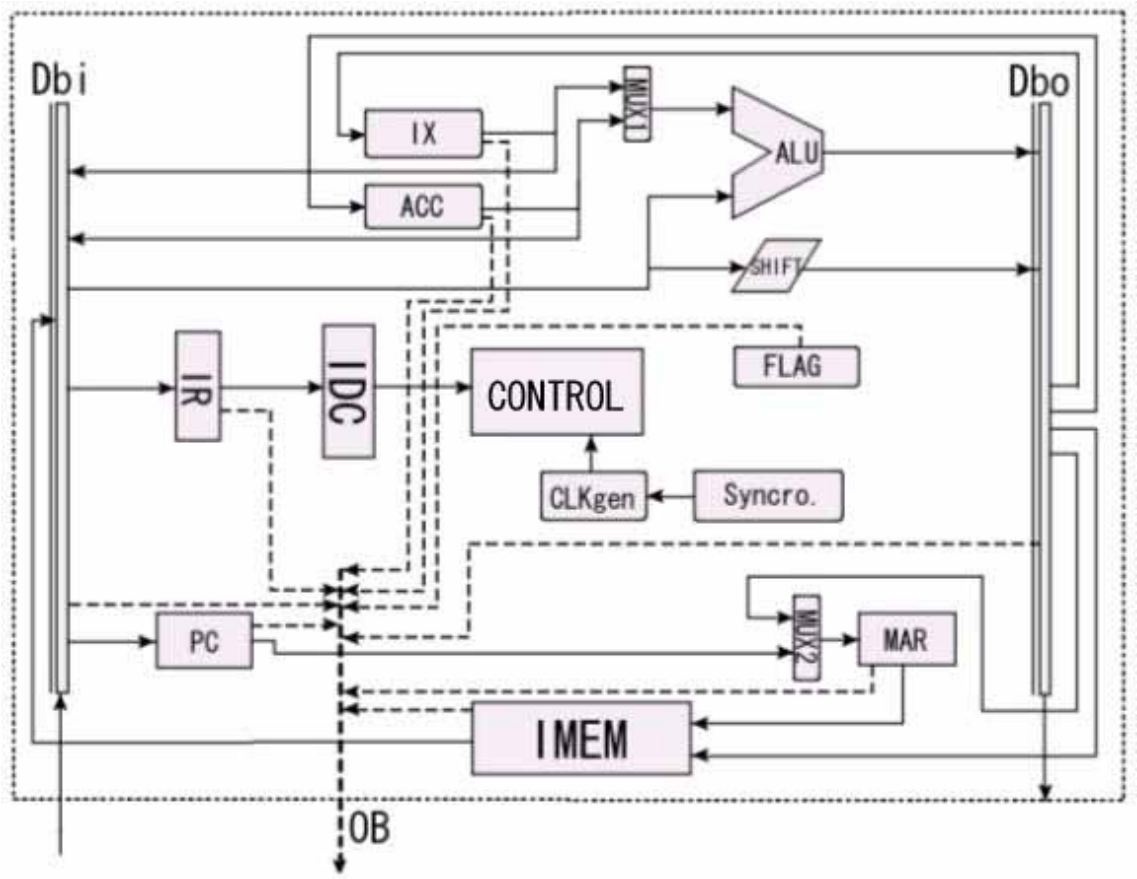

図5：KUE-CHIP2のブロック図

構成している各部の役割は以下のとおりである。

- ACC(アキュムレータ)：演算結果を保持し、オペランドに用いるレジスタ。
- IX(インデックスレジスタ)：演算結果を保持し、オペランドに用いるレジスタ。また、修飾アドレス指定のときのアドレス修飾に利用される。
- PC(プログラムカウンタ)：次に実行するメモリのアドレスを保持するレジスタ。
- MAR(メモリアドレスレジスタ)：メモリアクセスのためのアドレスを保持するレジスタ。読み書きを行なう対象のアドレスを保持する。
- IR(命令レジスタ)：実行する命令を保持するレジスタ。
- IDC(命令デコーダ)：命令レジスタの内容を解読する回路。
- ALU(演算器)：算術論理演算を行なう回路。アドレスの計算にも用いられる。
- SHIFT(シフト演算器)：シフト、ローテート演算を行なう回路。
- FLAG(フラグレジスタ)：ALU及びSHIFTの演算結果から、桁上げ(CF)、オーバーフロ

ー(VF)、負数の検出(NF)、ゼロ判定(ZF)の4ビットの情報を受け取り、保持するレジスタ。分岐判定に用いられる。

- CONTROL(制御回路)：命令デコーダ、及びクロックジェネレータのフェーズ信号を元に各部への制御信号を送る。
- CLKgen(クロックジェネレータ)：外部から与えられるクロックからフェーズ信号(P0からP4)を生成する回路。
- Syncro(シンクロナイザ)：スイッチなどのクロックと非同期の信号をクロックに同期するように調整する順序回路。
- Dbi(入力データバス)：メモリやACC、IXなどからの複数の入力信号から必要な入力を選択し、出力する伝送路(バス)。
- Dbo(出力データバス)：ALUあるいはSHIFTからの入力を選択し、メモリ、演算用レジスタ、またはKUE-CHIP2外部へ出力する伝送路(バス)。
- MUX(マルチプレクサ)：MUX1はACCとIXの選択を行ない、MUX2はMARへの入力に対してPCあるいはDboを選択する回路である。
- OB(観測用バス)：観測用に各種レジスタ、メモリ、バスの内容を選択して出力する。

なお、本研究ではメモリはFPGA内部で生成されるBlockRAMを内部メモリとし、FPGA外部のメモリ空間は利用しないものとする。また、今回は検証する8つのサンプルプログラムとデータのサイズの合計が各々256バイトに収まるため、通常512バイト(プログラム領域、データ領域各256バイト)から、プログラムとデータを256バイト内におさめたメモリ領域で実装した。

3.3 命令セット

KUE－CHIP２の命令セットを表１に示す。[1][2][3]

表１：KUE-CHIP2の命令セット

略記号	命令コード(1語目)	B'(2語目)	命令機能の概略	
NOP	0 0 0 0 0 - - -	×	No OPeration	
HLT	0 0 0 0 1 - - -	×	HaLT	停止
	0 1 0 1 - - - -	×		未使用(HLT)
OUT	0 0 0 1 0 - - -	×	OUTput	$(ACC) \rightarrow OBUF$
IN	0 0 0 1 1 - - -	×	INput	$(IBUF) \rightarrow ACC$
RCF	0 0 1 0 0 - - -	×	Reset CF	$0 \rightarrow CF$
SCF	0 0 1 0 1 - - -	×	Set CF	$1 \rightarrow CF$
Bcc	0 0 1 1 cc	◎	Branch cc	条件が成立すれば $B' \rightarrow PC$
Ssm	0 1 0 0 A 0 sm	×	Shift sm	$(A) \rightarrow$ shift , rotate $\rightarrow A$
Rsm	0 1 0 0 A 1 sm	×	Rotate sm	はみ出したビット $\rightarrow CF$
LD	0 1 1 0 A B	○	LoaD	$(B) \rightarrow A$
ST	0 1 1 1 A B	◎	STore	$(A) \rightarrow B$
SBC	1 0 0 0 A B	○	SuB with Carry	$(A)-(B)-CF \rightarrow A$
ADC	1 0 0 1 A B	○	ADd with Carry	$(A)+(B)+CF \rightarrow A$
SUB	1 0 1 0 A B	○	SUBtract	$(A)-(B) \rightarrow A$
ADD	1 0 1 1 A B	○	ADD	$(A)+(B) \rightarrow A$
EOR	1 1 0 0 A B	○	Exclusive OR	$(A) \oplus (B) \rightarrow A$
OR	1 1 0 1 A B	○	OR	$(A) \vee (B) \rightarrow A$
AND	1 1 1 0 A B	○	AND	$(A) \wedge (B) \rightarrow A$
CMP	1 1 1 1 A B	○	CoMPare	$(A)-(B)$

cc : Condition Code

A	0 0 0 0	Always	常に成立
VF	1 0 0 0	on oVerflow	桁あふれ $VF=1$
NZ	0 0 0 1	on Not Zero	$\neq 0$ $ZF=0$
Z	1 0 0 1	on Zero	$=0$ $ZF=1$
ZP	0 0 1 0	on Zero or Positive	≥ 0 $NF=0$
N	1 0 1 0	on Negative	<0 $NF=1$
P	0 0 1 1	on Positive	>0 $(NF \vee ZF)=0$
ZN	1 0 1 1	on Zero or Negative	≤ 0 $(NF \vee ZF)=1$
NI	0 1 0 0	on No Input	$IBUF\_FLG\_IN=0$
NO	1 1 0 0	on No Output	$OBUF\_FLG\_IN=1$
NC	0 1 0 1	on Not Carry	$CF=0$
C	1 1 0 1	on Carry	$CF=1$
GE	0 1 1 0	on Greater than or Equal	≥ 0 $(VF \oplus NF)=0$
LT	1 1 1 0	on Less Than	<0 $(VF \oplus NF)=1$
GT	0 1 1 1	on Greater Than	>0 $((VF \oplus NF) \vee ZF)=0$
LE	1 1 1 1	on Less than or Equal	≤ 0 $((VF \oplus NF) \vee ZF)=1$

sm : Shift Mode

RA	0 0	Right Arithmetically
LA	0 1	Left Arithmetically
RL	1 0	Right Logically
LL	1 1	Left Logically

A = 0 : ACC
A = 1 : IX

B'(2語目)
×：不用
○：不用 or 必要
◎：必要

B = 000 : ACC
B = 001 : IX
B = 01- : d (即値アドレス)
B = 100 : [d] (絶対アドレス)
B = 101 : (d) (絶対アドレス)
B = 110 : [IX+d] (インデックス修飾アドレス)
B = 111 : (IX+d) (インデックス修飾アドレス)

KUE－CHIP２の命令語は1語(1バイト)あるいは2語(2バイト)で構成される。1語目の上位4ビットあるいは5ビットによって命令の種類が識別される。シフト/巡回シフト命令、ロード/ストア命令、算術論理演算命令においては1語目上位5ビット目(表1中のAに相当)が第1

オペランドに、ロード/ストア命令、算術論理演算命令の1語目下位3ビット(表1中のBに相当)が第2オペランドと、最大2オペランド命令で構成される。2語長命令の2語目はアドレスあるいはデータであり、データの負数は二の補数表示で表される。

KUE－CHIP２には第２オペランドの指定の方法に４つのアドレスモードが存在する。

(1)　レジスタ指定モード

KUE－CHIP２にはアキュムレータ(ACC),インデックスレジスタ(IX)の2つの演算用レジスタがあり、これらのレジスタの値をオペランドとして指定する方式である。図6にADD命令によるレジスタ指定モードの演算の様子を示す。

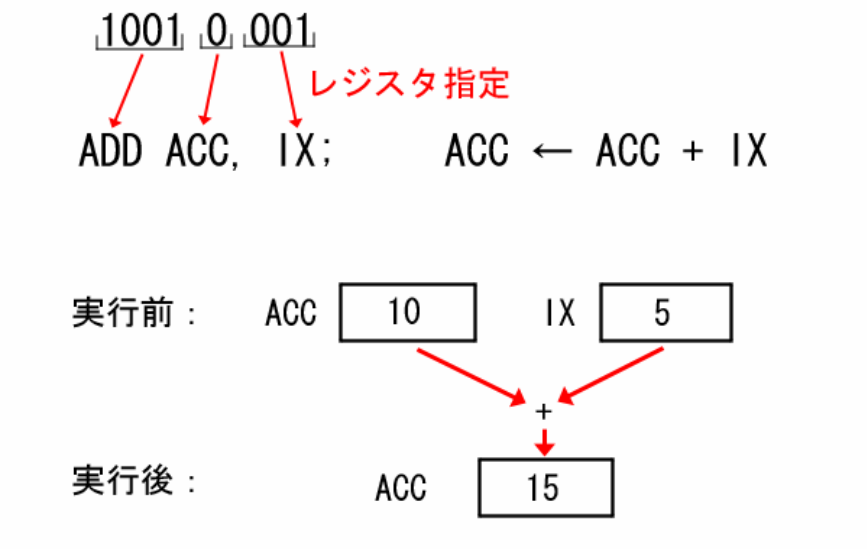

図6：レジスタ指定モードによるADD命令

(2)　即値アドレスモード

この方式では、算術論理演算命令ならば2語目の値が加減乗除や論理演算の対象のデータになり、ロード命令では2語目の値がそのままレジスタに書き込まれるオペランド指定方式である。図7に即値アドレスモードによるロード命令の実行の様子を示す。

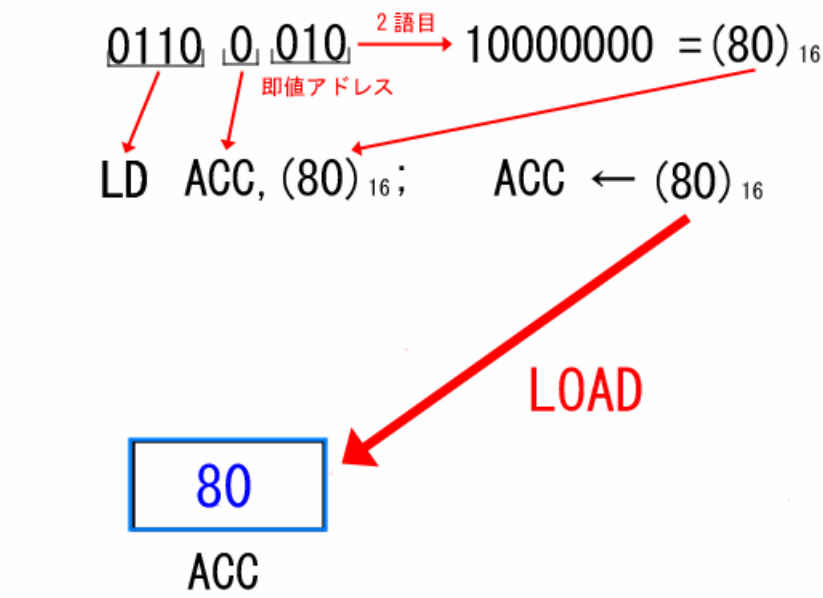

図7：即値アドレスモードによるLOAD命令の実行

(3)　絶対アドレスモード

　命令の2語目が有効アドレスを指すという方式であり、直接アドレス指定方式と呼ばれる。図8に絶対アドレスモードによるストア命令の実行の様子を示す。

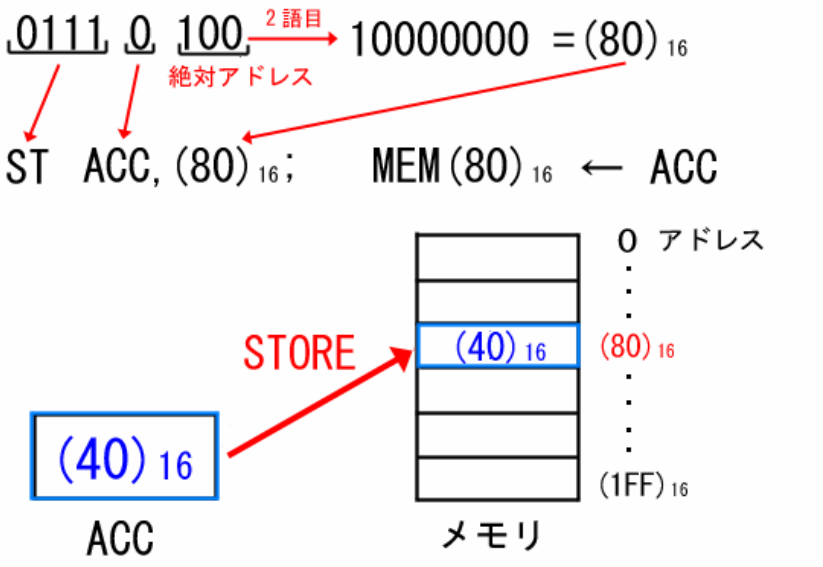

図8：絶対アドレスモードによるSTORE命令の実行

(4)　インデックス修飾アドレスモード

命令の2語目が示すアドレス定数とインデックスレジスタ(IX)の和を有効アドレスとする方式であり、このモードを用いることによってインデックスレジスタの値を要素番号とする配列の実現が可能である。図9にインデックス修飾アドレスモードによるロード命令の実行の様子を示す。

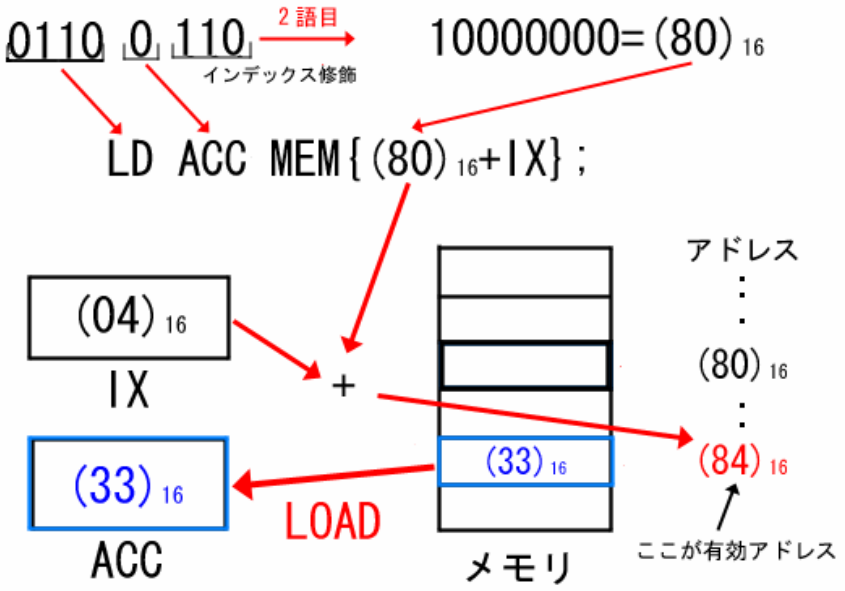

図9：インデックス修飾アドレスモードによるLOAD命令の実行

4　ハードウェア記述言語によるKUE-CHIP2の論理設計と検証

4.1　命令実行フェーズと動作

KUE-CHIP2は最大5段階の命令実行フェーズを持つマルチサイクルプロセッサであり、各命令のフェーズ動作に応じた制御系を状態遷移化することにより実現される。表2にKUE-CHIP2の各命令の実行フェーズ表を示す。[1][2][3]

表2：命令実行フェーズ表

phase / instruction		P0	P1	P2	P3	P4
HLT		(PC)→MAR PC++	(MEM)→IR	HALT		
NOP				NO OPERATION		
OUT				(ACC)→OBUF	0→OBUF_WE	
IN				(IBUF)→ACC 0→IBUF_FLG_RE	0→IBUF_FLG_CLR	
RCF				0→CF		
SCF				1→CF		
Bcc				(PC)→MAR　PC++	STATUS CHECK (MEM)→PC(if condition satisfied)	
Ssm/Rsm				TCF SET SHIFT A	NF, ZF, VF, CF SET	
LD	REG			(B)→A		
	IMM			(PC)→MAR PC++	(MEM)→A	
	MAR				(MEM)→MAR	(MEM)→A
	IX				(IX) (MEM) ALU→MAR	
ST	MAR			(PC)→MAR PC++	(MEM)→MAR	(A)→(MEM)
	IX				(IX) (MEM) ALU→MAR	
ALU	REG			(A) (B) ALU→A FLAGSET [(CF)]		
	IMM			(PC)→MAR PC++	(A) (B) ALU→A FLAGSET [(CF)]	
	MAR				(MEM)→MAR	(A) (B)→ALU→A FLAGSET [(CF)]
	IX				(IX) (MEM) ALU→MAR	

今回設計したKUE-CHIP2は、全て表2の命令実行フェーズ表に基づいてVerilog-HDLによって製作を行なった。構成されるレジスタへの書き込みは次の実行フェーズの始まる直前に行なわれる。

フェーズP0では、命令をフェッチするために、次に実行する命令の格納されたプログラムカウンタPCの内容をメモリアドレスレジスタMARに送り、その後PCの値を1増加させる。P0では、全ての命令で共通してこの動作が行なわれる。図10にフェーズP0におけるデータパスの様子を示す。

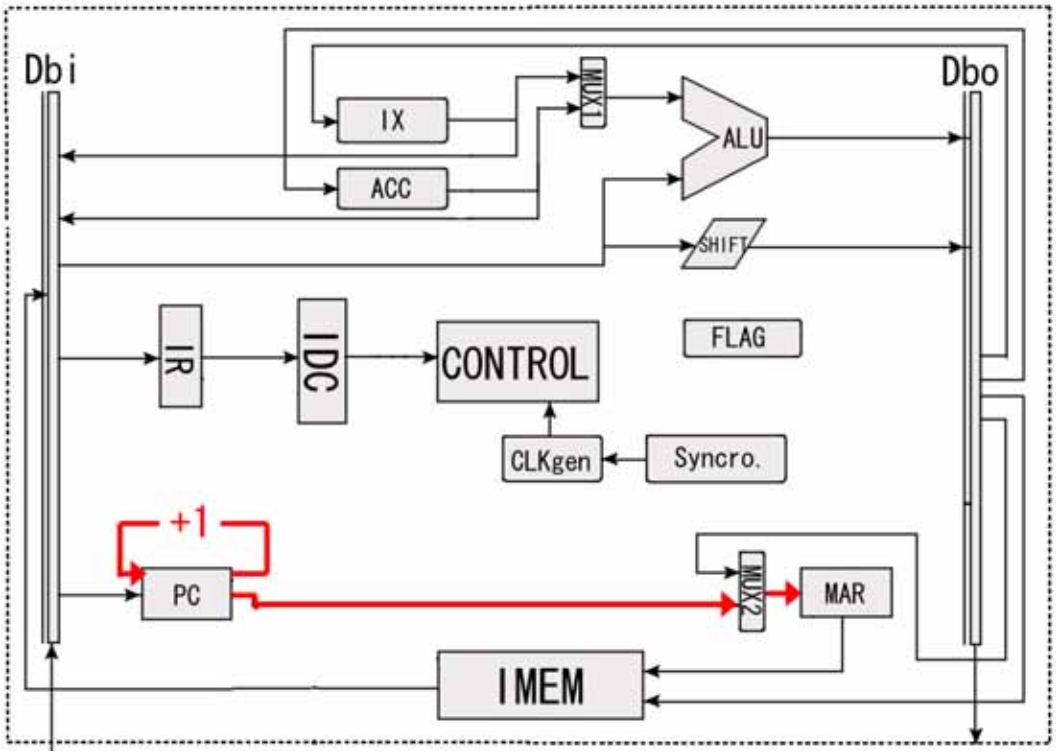

図10：フェーズP0におけるデータパス

フェーズP1においても、すべての命令で共通した動作が行なわれる。図11にフェーズP1におけるデータパスの様子を示す。

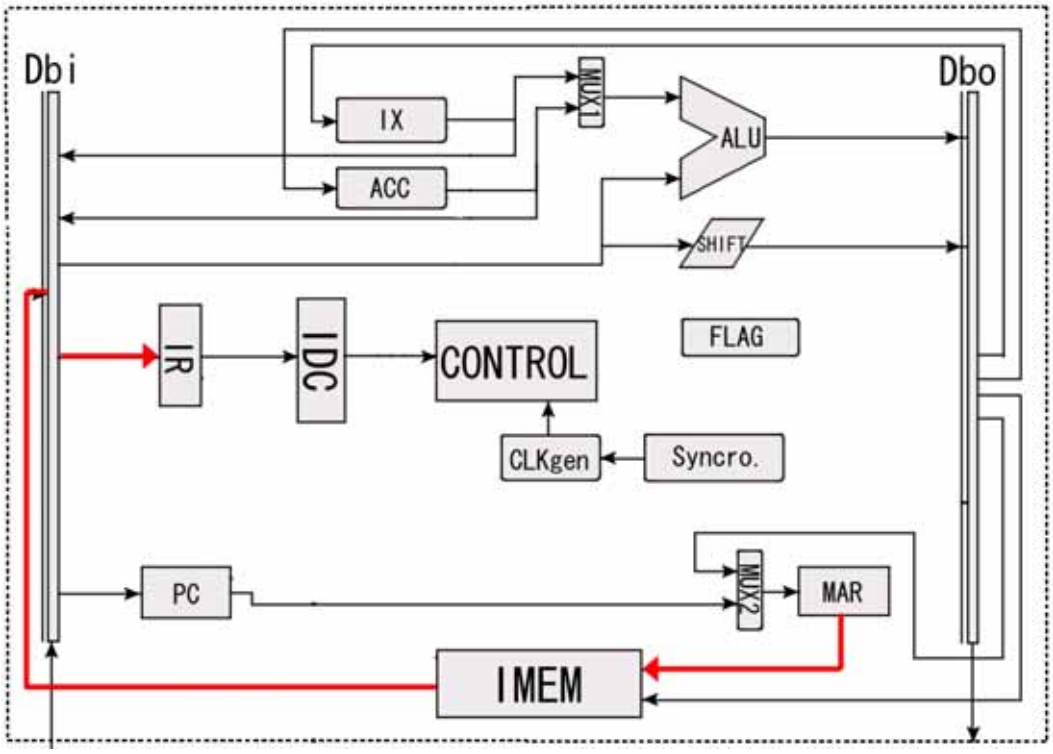

図11：フェーズP１におけるデータパス

フェーズP1では、メモリIMEMが読み出し状態になり、メモリアドレスレジスタの内容をアドレスとするメモリのデータを出力し、入力データバスDBiを通って命令レジスタIRに書き込まれる。

フェーズP2からP4では、表2のように命令の種類やアドレスモードによって動作が異なる。ここではアドレスモードごとの動作の様子を例を用いて説明する。

(1) レジスタ指定モード

第3章図6のレジスタ指定モードによるADD命令のデータパスを図12に示す

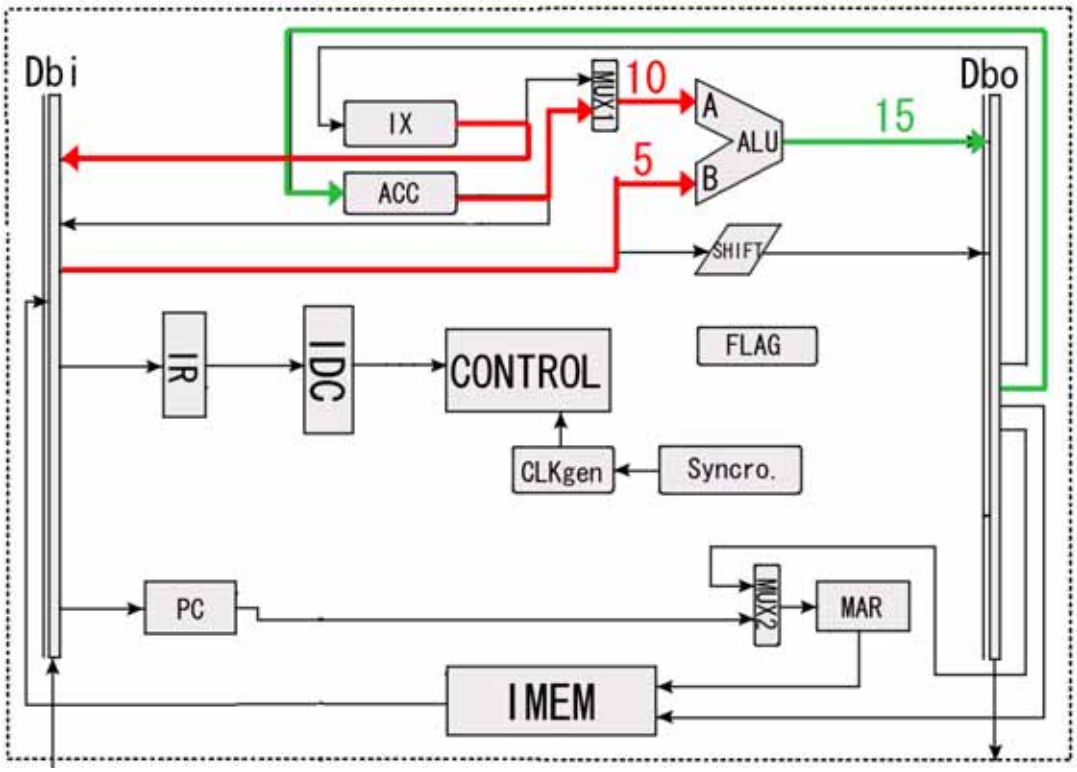

図12：レジスタ指定モードによるADD命令のデータパス

アセンブリ言語ではADD ACC,IX;と表記され、ACC(10)とIX(5)の和(15)を再びACCに格納する。ALUの入力AはACCであるのでマルチプレクサMUX1をACCに設定し、IXの内容がDBiを通って入力Bに入る。ALUによって加算されたデータはDBoを通り、再びACCに書き込まれる。以上の動作はすべてフェーズP2で行なわれ、再びフェーズP0に遷移する。

(2) 即値アドレスモード

第３章図７の即値アドレスモードによるLOAD命令のデータパスを図13に示す。

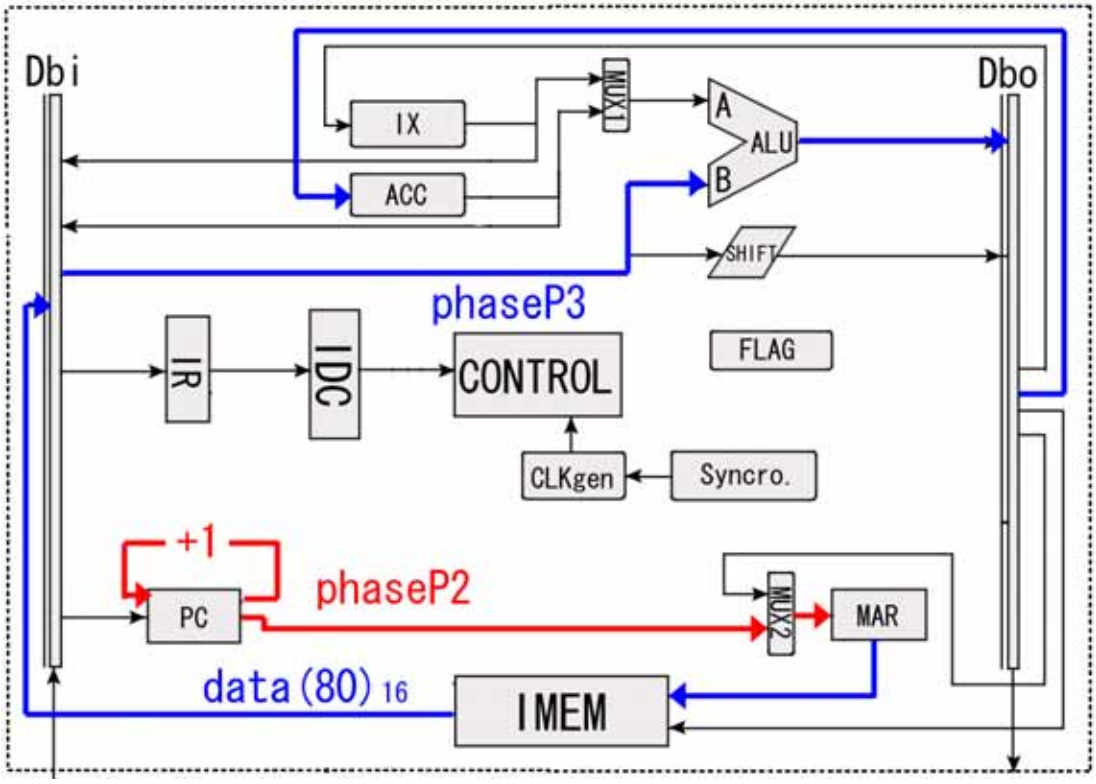

図13：即値アドレスモードによるLOAD命令のデータパス

アセンブリ言語ではLOAD ACC,80;と表記され、命令の2語目の値である(80)をそのままACCに格納する。フェーズP2では、PCに格納されている命令の2語目のアドレスをMARに送信し、PCの値を1増加させる。フェーズP3では、MARが示すアドレスのIMEMのデータ出力をDbi,ALU,Dboを経由してACCに書き込まれる。フェーズP3におけるALU出力は入力Bをそのまま出力するように設定している。

(3) 絶対アドレスモード

第３章図８の絶対アドレスモードによるSTORE命令のデータパスを図14に示す。アセンブリ言語ではST ACC, (80);と表記され、命令の2語目の値である(80)を有効アドレスとするメモリの番地にACCのデータ(40)を書き込む。フェーズP2では、PCに格納されている命令の2語目のアドレスをMARに送信し、PCの値を1増加させる。フェーズP3ではMARが示すアドレスのIMEMのデータ出力(80)をDbi,ALU,Dboを経由して再びMARに書き込まれる。フェーズP4では、書き込み対象のACC出力(40)をDbi,ALU,Dboを経由してIMEMに送信し、MAR出力(80)をアドレスとするIMEMの領域に(40)が書き込まれる。フェーズP3、P4におけるALU出力は入力Bをそのまま出力するように設定している。

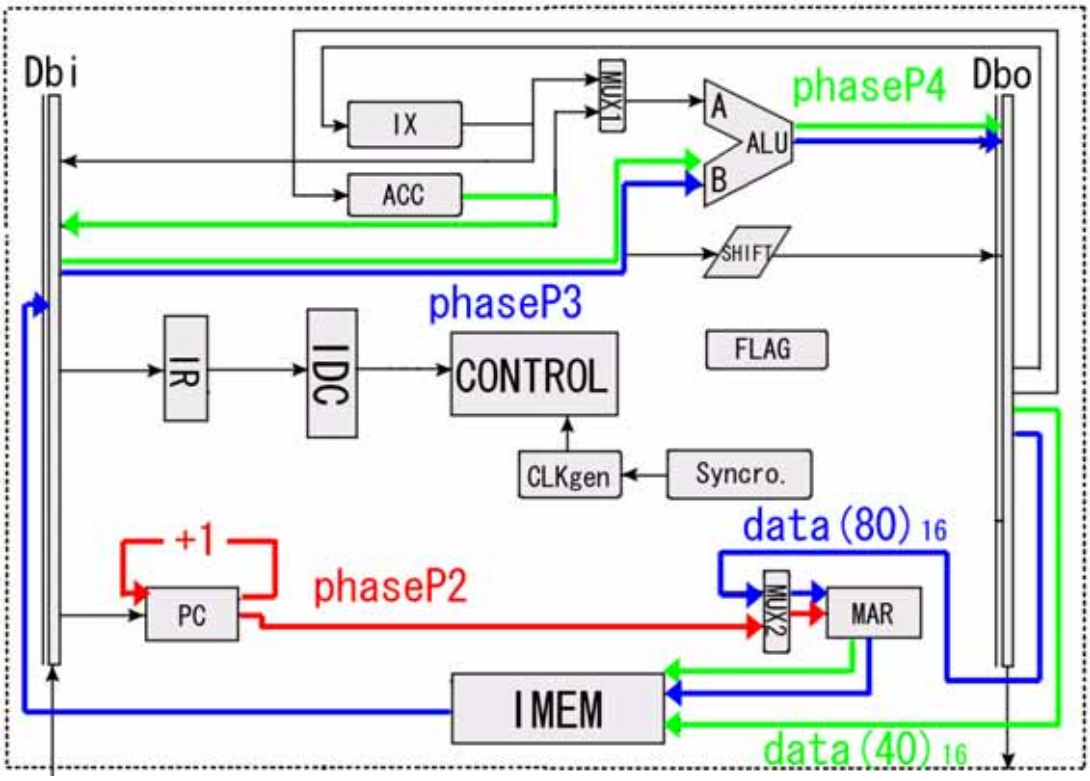

図14：絶対アドレスモードによるSTORE命令のデータパス

(4) インデックスアドレスモード

第３章図9のインデックスアドレスモードによるLOAD命令のデータパスを図15に示すアセンブリ言語ではLOAD ACC, (80+IX);と表記され、命令の2語目の値である(80)とIXの内容(4)の和(84)を有効アドレスとするメモリの番地のデータ(33)をACCに書き込む。フェーズP2では、PCに格納されている命令の2語目のアドレスをMARに送信し、PCの値を1増加させる。フェーズP3ではMARが示すアドレスのIMEMのデータ出力(80)とIX(4)の和をALUが出力し、再びMARに書き込まれる。フェーズP4では、MARの内容(84)をアドレスとするIMEMの出力(33)をDbi,ALU,Dboを経由してACCに送信し書き込まれる。

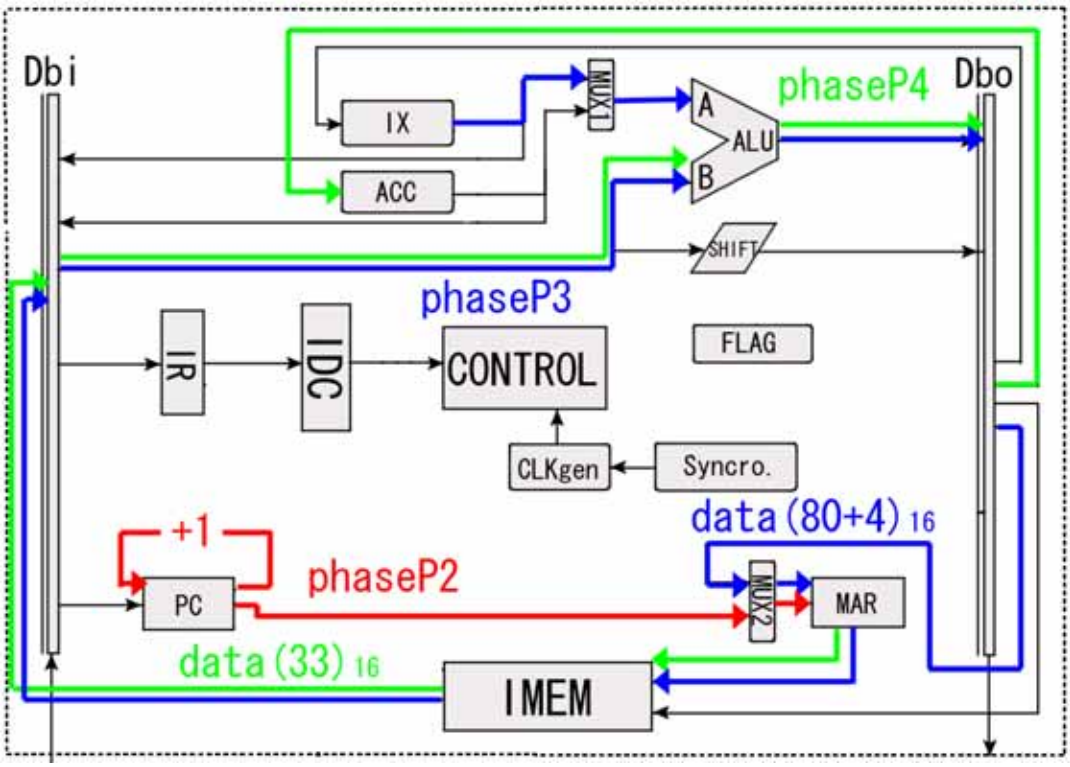

図15：インデックスアドレスモードによるLOAD命令のデータパス

4.2　モジュールの設計と単体検証

4.2.1　モジュールの設計

4.1で述べた命令実行フェーズの動作を満たすためのハードウェア記述言語によるモジュールの設計を行った。

(1)　クロックジェネレータCLKgen

同期された実行の信号やコントロール部から送られる制御信号からフェーズ信号を生成するユニットである。図16にクロックジェネレータの外部仕様の図を示す。

クロック入力CLKや各実行モード入力、制御回路からのフェーズリセットや停止信号などの入力信号をもとに、命令実行フェーズと現在の実行モードの情報を制御回路に出力し、出力opは外部のLED点灯にのみ使用する。

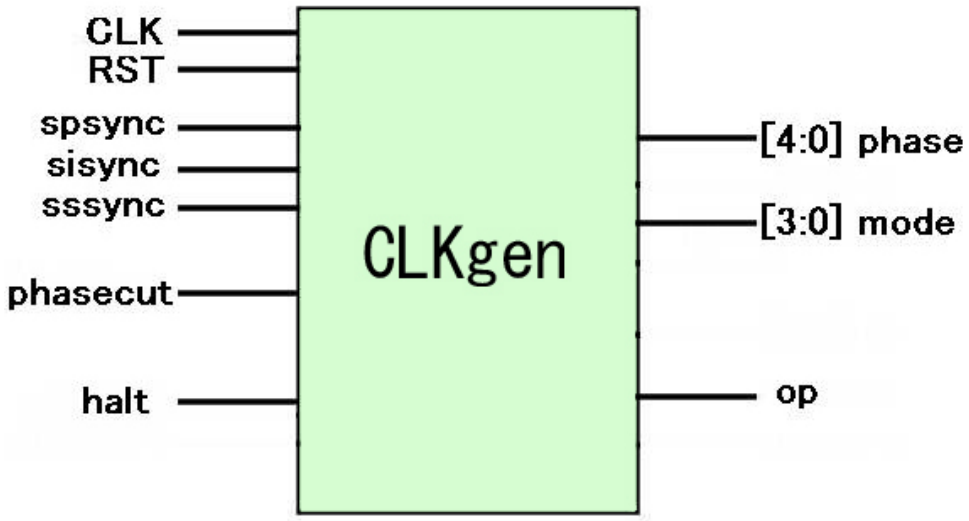

図16：クロックジェネレータの外部仕様

クロックジェネレータではSP、SI、SSといった各種実行に対応するためにSTANDBY、SP、SI、SSの4つのmodeが内部でクロックサイクルごとに状態遷移している。図17に内部状態modeの状態遷移の様子を示す。

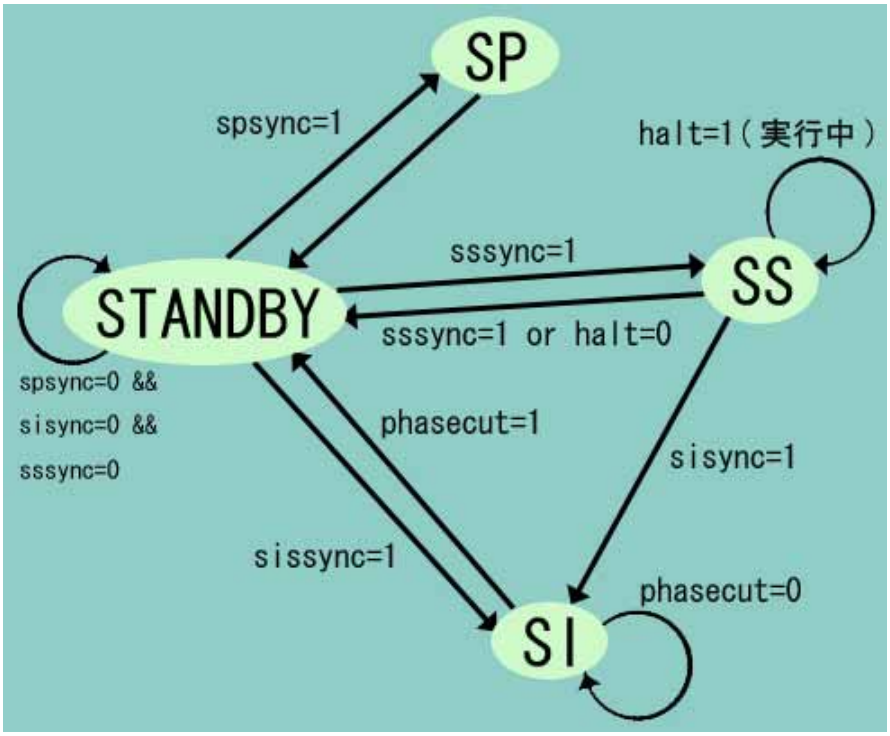

図17：内部状態modeの状態遷移

STANDBYはプロセッサの停止状態を示し、実行入力とクロックが同期して各実行状態に遷移する。実行状態にいる間はクロックサイクルの度に命令実行フェーズが次の段階に移行してゆき、phasecutやhalt信号などの入力に応じて図17のように状態が変化してゆく。

Verilog-HDLによって記述したクロックジェネレータのソースの一部を図18に示す。modeとphaseの状態をalways文の中のcase文やif文によって状態遷移を実現している。次に移行するモードとフェーズをそれぞれnextmode、nextphaseにノンブロッキング代入を行い、クロックの立ち上がりでmode、phaseに書き込まれる。

```
module CLKgen(phase,mode,…spsync,sisync)
.................
parameter STANDBY=4'b0001,                          // 内部状態 mode のパラメータ宣言
                .............
parameter p0=5'b00001,                              // 内部状態 phase のパラメータ宣言
           .........
always@(posedge CLK or negedge RST)
   ............
   mode<=nextmode;                                  // クロックサイクルごとの mode,phase の移行
   phase<=nextphase;
   ............
always@(mode or spsync or … or phasecut or halt)    // 内部状態 mode の遷移
   begin
     case(mode)
     STANBY:if(sssync)
            nextmode<=SS;
            else if(sisync)
            ............
     SP:begin
          nextmode<=STANBY;
        ............
   end
always@(mode or phase or phasecut or halt)          // 内部状態 phase の遷移
  begin
    case(phase)
      p0:begin
           case(mode)
             STANBY:nextphase<=p0;
                 SP:nextphase<=p1;
                 SI:nextphase<=p1;
                 SS:nextphase<=p1;
     ............
  end
endmodule
```

図18：クロックジェネレータのHDL記述例

(2)　制御回路CONTROL

命令デコーダから送られた命令の種類やクロックジェネレータからのフェーズ、実行モードなどをもとに各モジュールに制御信号を送るプロセッサの中枢にあたるモジュールである。図19に制御回路CONTROLの外部仕様の図を示す。

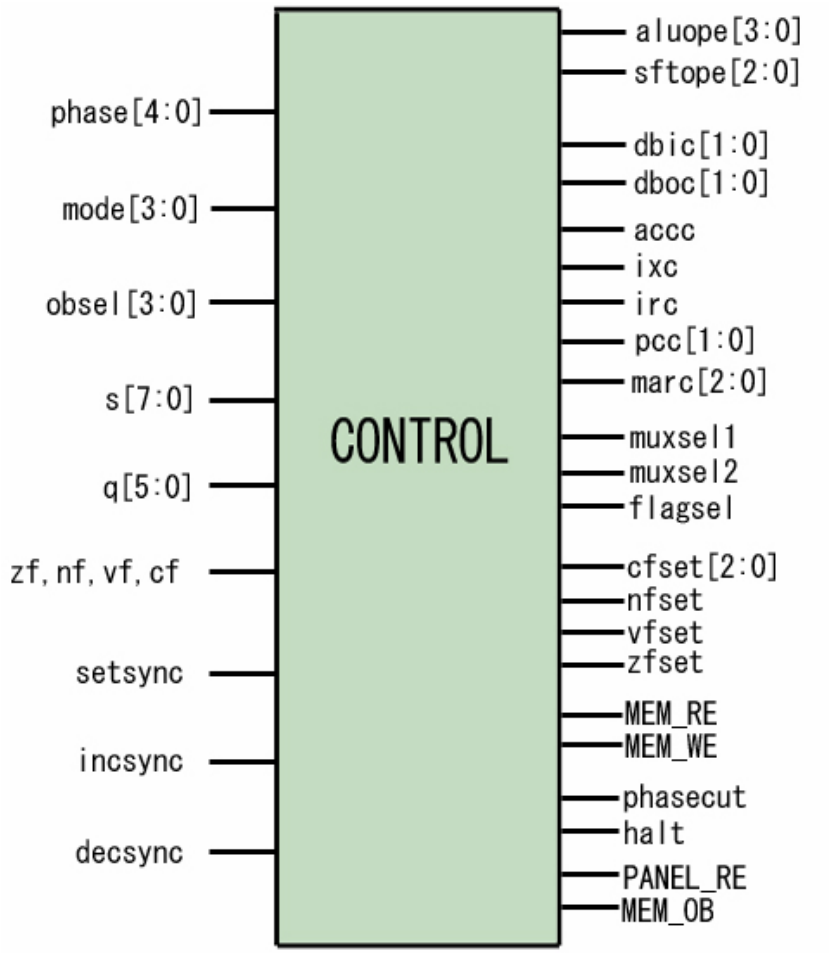

図19：制御回路の外部仕様

クロックジェネレータから送られるmode、phase、命令デコーダから送られるqの組み合わせによって複数の状態が存在し、命令レジスタの内容sによってオペランドの演算用レジスタの指定や分岐命令、演算命令、シフト命令の種類を指定するのに用いられる。フラグ入力は分岐条件として使用する。これらの入力信号をもとに、ALUやSHIFTの演算の種類を指定したり、フラグを含む各種レジスタの書き込み指定、データバスの制御、マルチプレクサへの選択信号の送信、メモリの読み書き指定、クロックジェネレータへのフェーズリセットや終了信号の送信などが行なわれる。また、プロセッサ外部との入出力の制御も行なわれる。Verilog-HDLによって記述した制御回路のソースの一部を図20に示す。

```
module CONTROL(phase,mode,s,q,irc,ixc,accc,aluope,zfset,……);
parameter NOP   =6'h00,            // 命令の種類、フェーズ、実行モードなどを
                ……                 // パラメータ宣言
parameter BZP = 4'b0010,
                ……
always@(phase or mode or q or …… or set or inc or dec)
   begin
     if(mode == STANDBY)begin
         pcc<=2'b00;
         ……
         accc<=0;
         ixc<=0;end
     else if(mode == SP || SI || SS)begin
         case(phase)
          p0:begin/////////phase0        // 共通部分の制御
               pcc<=2'b01;
               ……
               accc<=0;
               ixc<=0;end
          p1:begin/////////phase1
               …… end
          p2:begin////////phase2
               case(q)
                 NOP:begin              //P2 以降は命令の種類、オペランド
                       ……               // 等によって場合分けされる
               endcase
        endcase
  end
……
assign irc = (phase == p1)? 1'b1:          // 状態遷移の必要がない信号は
             1'b0;                           継続的代入文で記述
……
endmodule
```

図20：制御回路のHDL記述例

クロックジェネレータと同様に、always文による状態遷移によって制御信号が生成されるが、HDL行数が膨大になるので今回の設計では、動作に問題が無い制御に関しては継続的代入文assignを用いて制御を簡潔にしている。

4.2.2　テストベンチによるモジュールの動作検証

モジュールの設計後、仕様どおりの動作を行うかどうかシミュレーションツールによる検証を行なう。その際、テスト対象のモジュールに対してどのようなタイミングで入力信号を与えるかを記述し、出力を観測するテストベンチが必要である。図21に命令レジスタに対するテストベンチの記述例を示す。

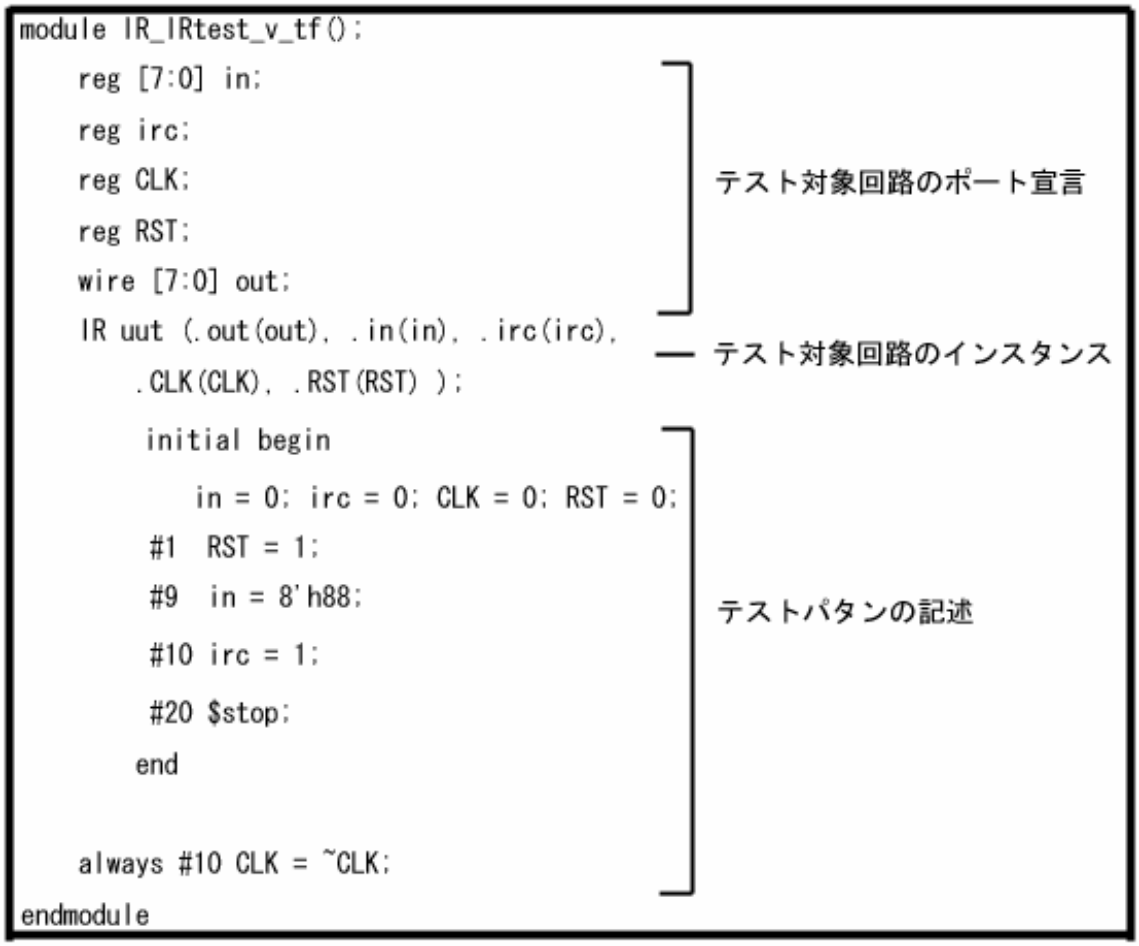

図21：テストベンチの記述例

テストベンチの記述後、ModelTechnology社のModelSimというシミュレーションツールにより、波形による動作の確認を行なう。ModelSimを使用すると図22のような波形図が現れる。

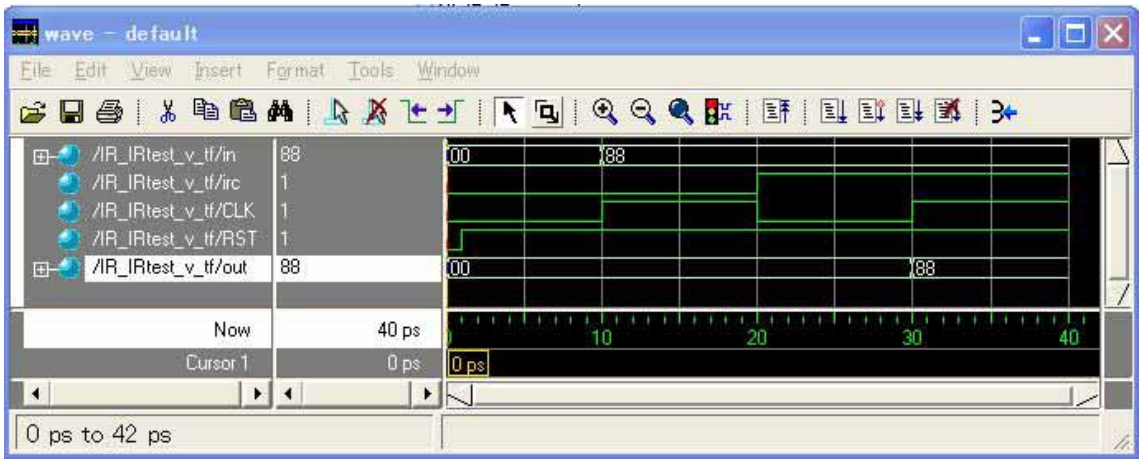

図22：ModelSimによるIRのシミュレーション

図23のタイミングチャートでは制御信号入力ircが1でかつCLKの立ち上がりでデータ入力88がレジスタに記憶されている様子がわかる。シミュレーションを重ねながらモジュールをデバッグすることにより、モジュールの論理設計が行われる。

4.3　シミュレーションによる全体検証

KUE-CHIP2のHDL記述後、ModelSimによってKUE-CHIP2全体のシミュレーションを行い、KUE-CHIP2の全体動作検証及びデバッグを行なう。検証用サンプルプログラムに、「Nまでの和」「最大値探索」「多倍長の加算」「ユークリッド互除法」「バブルソート」「符号なし1バイト乗算」「符号なし1バイト除算」「CRC計算」を用いた[2][3]。表3に各プログラムの行数を示す。

表3：検証用プログラムの行数

サンプルプログラム	プログラム行数
1 から N までの和	7
最大値探索	10
多倍長の加算	8
ユークリッド互除法	11
バブルソート	23
符号無し 1 バイト乗算	26
符号無し 1 バイト除算	39
CRC 計算	35

(単位：行)

これらのプログラムが正しく動作するかシミュレーション中のレジスタ、データバス、制御線のタイミングを確認しながら検証を行なう。図23にModelSimによる1からNまでの和のシミュレーションの様子を示す。

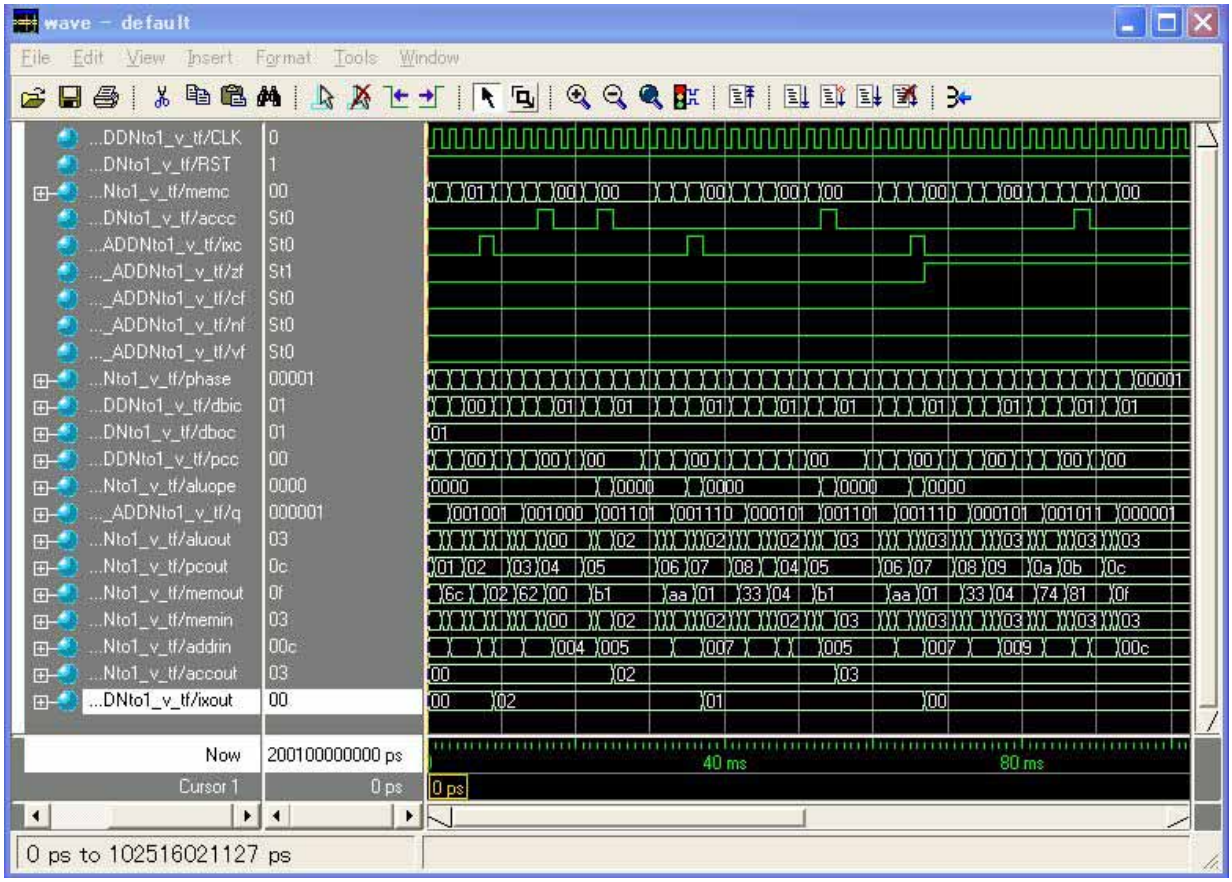

図23：1からNまでの和のシミュレーション

図23は1からNまでの和のプログラムでN=2とした場合のシミュレーションである。図の下2つの要素ixout、accoutはそれぞれ演算用レジスタIX、ACCの内容を示しており、IXの内容をACCに加算し、IXをデクリメントしながら、0になるまで加算を繰り返す。ACCには最終的に解である3が格納されていることが確認できる。値を確認しながら、不正な動作が確認されれば、他の制御線などのタイミングを見ながら、原因を推測し、HDLソースを調整するなどしてデバッグを繰り返す。そして、正常な動作の確認が出来れば、論理合成、配置配線とステップを進めて行き、デバイスへの実装となる。

5　KUE-CHIP2のFPGAボードへの実装と検証

5.1　KUE-CHIP2のFPGAボードへの実装

EDKツールによって設計されたKUE-CHIP2の実装を行なった。KUE-CHIP2教育用ボードを販売している京都高度技術研究所(ASTEM)では、教育用ボードとの接続に対応するXilinx社のプログラマブルデバイスXC5210を搭載しているKUECHIP2 FPGA Boardを用意しているが、デバイスのサポートや、開発ツールのヴァージョン等の理由により本研究での使用が困難と判断し、代替するFPGAボードを使用した。図24に今回使用した有限会社ヒューマンデータのSpartanⅡ学習用ボードXSP-006を示す。[21]

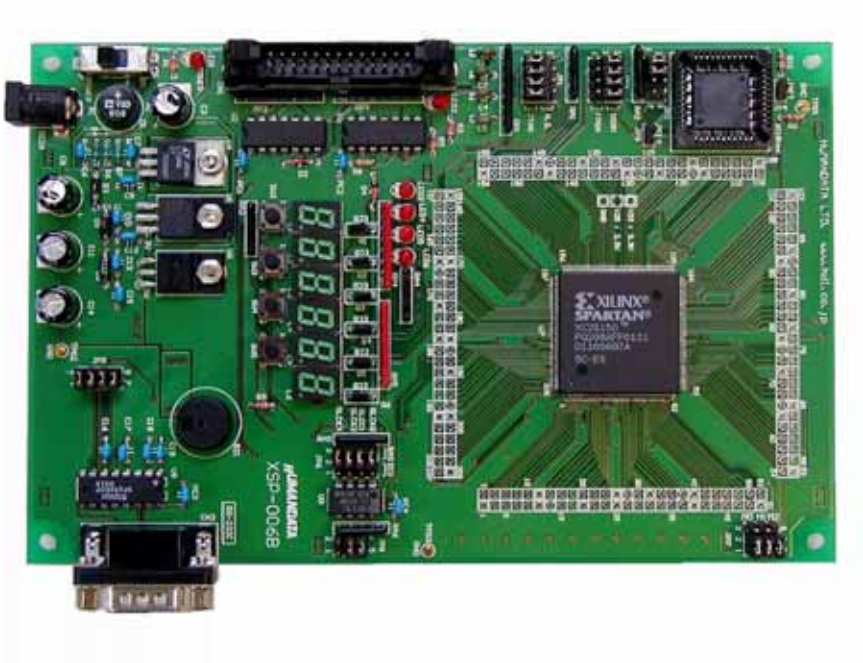

図24：SpartanⅡ学習用ボードXSP-006

XSP-006はXilinx社のFPGA SpartanⅡの他に7セグメント表示器6個、汎用LED4個、押しボタンスイッチ４個等が付属しており、これらの装置を用いてFPGAの操作や観測がボード単体で可能である。KUE-CHIP2教育用ボードでは動的な観測対象の切り替えやメモリへの書き込み用途に多数の表示機器、スイッチがあるため、FPGAは教育用ボードへの接続を前提とした入出力設定を行なった。

今回使用したSpartanⅡにはクロック専用の入出力ピンGCLKがあるが、XSP-006では19.432MHzのクロック生成器がGCLKを使用している。そこで、今研究では、FPGA入力のクロック周波数をクロック分周回路で静的に（コンフィギュレーション以前に分周回路のHDL記述を変更する）変更することにした。

5.2　FPGAボードと教育用ボードの接続

設計したFPGAボードをKUE-CHIP2教育用ボードで操作、観測を行なうため、接続を行なった。図25にFPGAボードとKUE-CHIP2教育用ボードの接続の様子を示す。

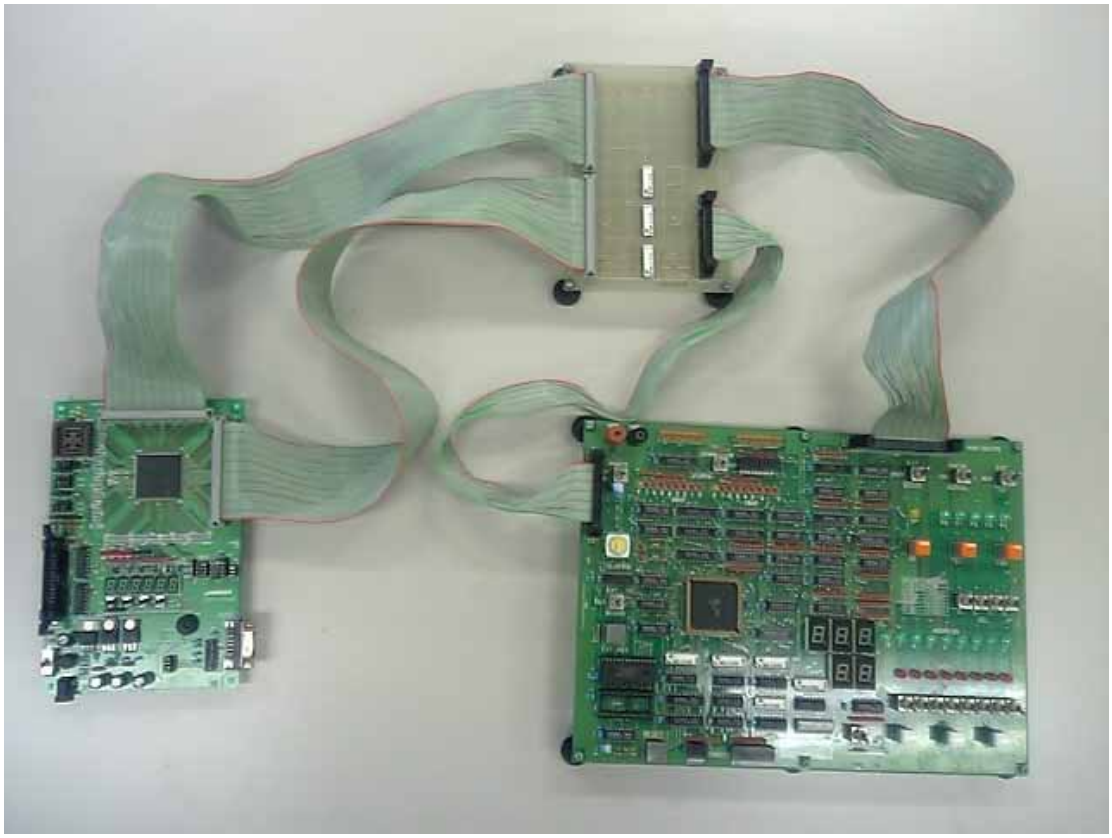

図25：FPGAボードとKUE-CHIP2教育用ボードの接続

FPGAボードとKUE-CHIP2教育用ボードではピン配置やケーブルの種類が異なり、抵抗器による調整も必要であるため、互いを繋ぐためのインターフェース回路の設計を行った。

ボードの電源電圧が、教育用ボードが5V、FPGAボードが3.3Vと異なるため、教育用ボードの電源電圧を3Vに引き下げて接続を行なった。その結果、教育用ボードの7セグメントディスプレイの発色が薄く、見づらいという問題が生じたが、その他LED、スイッチ類の正常な操作の確認がとれた。FPGAボードの7セグメントディスプレイを用いることで検証機能を満たすことができる。

5.3　実機上での動作検証

5.3.1　動作検証の方法

FPGAボードとKUE-CHIP2教育用ボードの接続による検証方法について述べる。

(1)　メモリへのプログラムとデータの書き込み

メモリへの書き込みの方法はコンフィギュレーション時に静的に書き込む方法と、コンフィギュレーション後のスイッチ操作による書き込みの2種類の方法がある。

- コンフィギュレーション時の書き込み

この方法では、実行対象のプログラムやデータをあらかじめCOEファイルと呼ばれるテキストファイルに保存しておき、コンフィギュレーション時にFPGA内のメモリに書き込む。図26にCOEファイルの内容を示す。

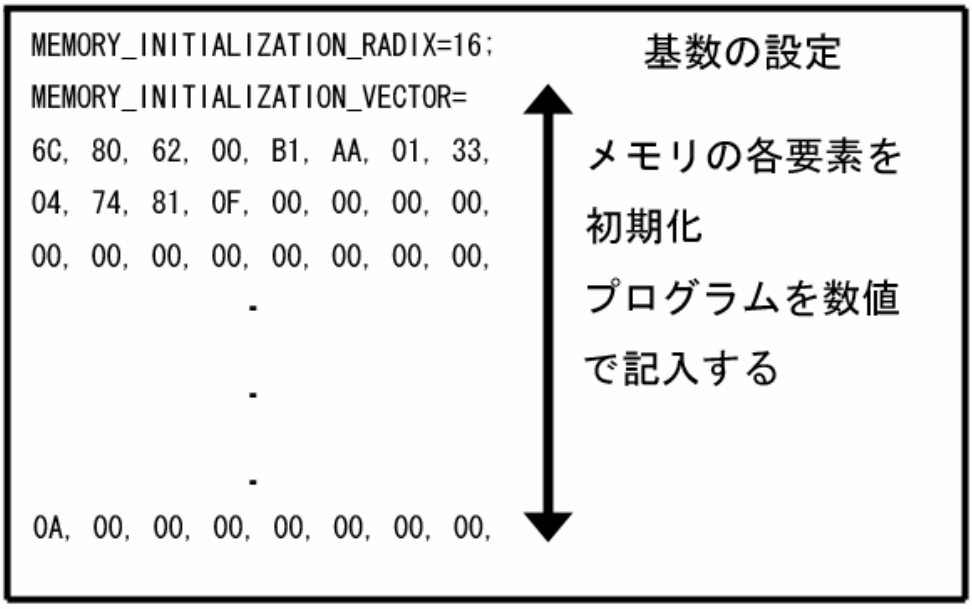

図26：COEファイルによるメモリの書き込み

サンプルプログラムごとのCOEファイルを用意しておくことでコンフィギュレーション時に切り替えてメモリの一括書き込みができる。FPGAを設計する際、実機上に何度もコンフィギュレーションを重ねるため、その度メモリの書き込み作業が短縮されるという利点を持つ書き込み方式である。

- コンフィギュレーション後の書き込み

この方法は従来の教育用ボードが持つメモリの書き込み方式である。コンフィギュレーション後にオンボード上のデータトグルスイッチ、SETスイッチでメモリ内容を書き込むことができる。実装後の対象データやプログラムの一部を変更するのに適している。

(2)　命令の実行とFPGA内部の観測

命令の実行は、1クロックフェーズ実行するシングルフェーズモードSP、1命令実行のシングルインストラクションモードSI、停止命令まで実行するスタートストップモードSSの3つの実行モードから選んで実行する。

教育用ボードの4ビットOB_SELスイッチによってFPGA内部のレジスタや、メモリ出力の値を動的に切り替えて7セグメントLED上で観測することができる。この機能の利用により、設計したアーキテクチャの検証、デバッグが容易に行なうことができる。表4にOB_SELスイッチと観測内容の対応表を示す。

表4：OB_SELスイッチと観測内容の対応

観測内容	OB_SEL の値
MEM	0000or0001
PC	0010
FLAG	0011
ACC	0100
IX	0101
DBi	0110
DBo	0111
MAR	1000
IR	1001

※他のOB_SELの値は未設定

(3)命令実行後のメモリ内容の観測

プログラム実行後、メモリに正しい値が書き込まれているかどうか確認する必要がある。教育用ボードでは、INC、DECスイッチにより、メモリアドレスレジスタMARの値をインクリメント、デクリメントすることができ、OB_SELの値をMEMにすることでメモリの中身を確認することができる。また、離れたアドレスの値を確認したい場合は、OB_SELの値を1000(MAR)に設定し、DATAトグルスイッチを参照したいアドレスの値を設定した上

でSETボタンを押すと、参照したいアドレスがMARに書き込まれる。その後でOB_SELの値を0000or0001(MEM)にすると、メモリの値が表示される。このような方法により、設計したアーキテクチャの検証、デバッグが可能となった。

5.3.2 動作検証

設計したプロセッサの検証用サンプルプログラムに、論理検証で利用した「Nまでの和」「最大値探索」「多倍長の加算」「ユークリッド互除法」「バブルソート」「符号なし1バイト乗算」「符号なし1バイト除算」「CRC計算」を用いた。正常な動作の比較対照として、京都工芸繊維大学の山村周史氏がWeb上で公開しているKUE-CHIP2シミュレータ[22]を参考し、動作検証を行なった。検証の結果、上記プログラムの正常な動作を確認することができた。

5.4 考察

FPGAボードと教育用ボードを接続する事により、プロセッサ検証において、豊富な操作が可能となり、観測性が動的に切り替えることができるという点で高くなった。しかし、FPGAの性能向上により、教育用ボードとのテクノロジのギャップが広がり、電源電圧が教育用ボードと同じ5Vのコンフィギュアブルデバイスは既にxilinx社では生産を終了し、3.3V以下のデバイスしか販売されていない。また、この電圧の差によって教育用ボードの7SEGLEDの発光が暗くて見えづらい事や、外部インターフェースが不安定である事、ボード間やPCと接続することによりスペースを広く占有してしまい、実際に学習実験として行なうには不便であるなどの問題も今回の検証で挙げられる。そこで、プロセッサの設計環境の改善案としては教育用ボードの操作性、観測性を保つために、豊富なスイッチ、LED出力を搭載したFPGAボードを新たに設計することが挙げられる。FPGAボードにKUE-CHIP2の構成情報を保存したROMを搭載させることで、上記問題は改善され、学習実験に適したプロセッサの設計環境になると思われる。

6 おわりに

本論文ではハードウェア記述言語Verilog-HDLを用いて教育用マイクロプロセッサKUE-CHIP2の設計を行った。シミュレーションツールによる回路の論理検証の後、論理合成、配置配線を行い、FPGAボード上への実装を行なった。また、KUE-CHIP2教育用ボードの充実した操作、観測性能を実現するため、FPGAボードと教育用ボードの接続を行なった。これらの経緯からプロセッサアーキテクチャの動作、EDKツールによるハードウェアのトップダウン設計、ハードウェア記述言語Verilog-HDLの文法を習得することができ、今後につながる基礎的な知識を身につけることができた。また、本学の学生実験で行なわれているアセンブリプログラミング演習にこのようなプロセッサアーキテクチャの設計を組み合わせることによって、ハードウェアとソフトウェア両方の相互関係をさらに直感的に学習することができるのではないかと考えている。

今後の課題としては、学生実験に適した教育用FPGAボードコンピュータの検討と試作である。本研究ではFPGAボードと教育用ボードの接続を行なったが、これを1つの教育用FPGAボードにすることで、実際の教育現場での有用性が考えられる。パイプラインやスーパースカラ等の様々なアーキテクチャの実装においても、優れた操作性、観測性を保つ仕様を検討し、設計の自由度の高いFPGAボードコンピュータの設計と試作を行ないたいと考えている。

謝辞

本研究の機会を戴き、貴重な助言、ご指導を頂きました山崎勝弘教授、小柳滋教授、そしてKUE-CHIP2の設計者で、数多くのご相談に応じていただき、貴重な助言を頂きました京都高度技術研究所の神原弘之氏に深く感謝致します。また、本研究の指導係にあたる中谷浩之氏をはじめ、同研究室内の方々の様々な励ましや助言に対し、深く感謝致します。

参考文献

[1] 神原弘之，越智裕之，澤田宏，浜口清治，岡田和久，上嶋明，安浦寛人：KUE-CHIP2 設計ドキュメント version1.10，京都高度技術研究所，1993

[2] 神原弘之，越智裕之，澤田宏，浜口清治，岡田和久，上嶋明，安浦寛人：KUE-CHIP2 教育用ボードリファレンスマニュアル version1.11，京都高度技術研究所，1993

[3] 立命館大学理工学部情報学科,情報学実験Ⅲ「教育用ボードコンピュータ」テキスト

[4] John L. Hennessy, David A. Patterson 著，成田光彰 訳： コンピュータの構成と設計（上）（下），日経 BP 社，1999

[5] 西村，額田，天野：教育用パイプライン処理マイクロプロセッサ PICO｀2′の開発 電子情報通信学会技術研究報告，Vol.99，No.532(CPSY99 106-116)，pp 61-68，2000.01.12.

[6] 田中,久我,末吉,小羽田：教育用マイクロプロセッサ KITE とその開発支援環境,情報処理学会研究報告,Vol.93, No.49(ARC-100), pp.59-66 ,1993.06..

[7] 大八木,池田,山崎,小柳：ハード/ソフト・コラーニングシステムにおけるアーキテクチャ選択可能なプロセッサシミュレータの設計,情報処理学会 第 66 回全国大会論文集,2004.

[8] 池田,中村,大八木,Tuan,山崎,小柳:ハード/ソフト・コラーニングシステムにおける FPGA ボードコンピュータの設計,情報処理学会 第 66 回全国大会論文集,2004.

[9] 大八木, 池田, 山崎：HDL による RISC プロセッサの設計経験(Ⅰ)-命令セットアーキテクチャと設計-, FIT2002, c-7, 2002.

[10] 池田,大八木,山崎：HDL による RISC プロセッサの設計経験(Ⅱ)-性能評価と考察-, FIT2002, c-8, 2002.

[11] 池田修久：ハード/ソフト･コラーニングシステム上での FPGA ボードコンピュータの設計と実装，立命館大学理工学研究科修士論文，2004

[12] 大八木睦：ハード/ソフト・コラーニングシステム上でのアーキテクチャ選択可能なプロセッサシミュレータの設計と試作,立命館大学大学院理工学研究科,修士論文,2004.

[13] 中村浩一郎：FPGA ボードコンピュータの開発,立命館大学理工学部情報学科,卒業論文,2004.

[14] 古川達久： マルチサイクル・パイプライン方式による教育用マイクロプロセッサの設計と検証，立命館大学理工学部卒業論文，2003

[15] 中村央志：ハードウェア記述言語による教育用マイクロプロセッサの設計（I），立命館大学理工学部卒業論文，2003

[16] 藤原淳平：ハードウェア記述言語による教育用マイクロプロセッサの設計（II），立命館大学理工学部卒業論文，2003

[17] 立命館大学 VLSI センター：社会人向け VLSI 設計セミナーレクチャー用マニュアル，2004

[18] 並木秀明，前田智美，宮尾正大：実用入門ディジタル回路と Verilog-HDL，技術評論社，2004

[19] ザイリンクス：http://www.xilinx.co.jp/

[20] 東京エレクトロンデバイス：http://www.teldevice.co.jp/

[21] ヒューマンデータ：http://www.hdl.co.jp/

[22] 京都工芸繊維大学コンピュータシステム研究室：
http://www.ark.dj.kit.ac.jp/Index.html