Group22プロセッサ Ver.3.00  
ユーザーマニュアル

平成29年度　計算機科学実験及演習3A

3回生前期学生実験　HW最終報告

提出期限：６月16日

提出日：6月16日

第22班

1029277526 白石　竜也

1029272870 　谷　勇輝

# 1 概要

Group22プロセッサ Ver.3.00は、京都大学工学部情報学科の2017年度計算機科学実験及演習３において第22班が作成したソフトコア・マイクロプロセッサである。SIMPLEアーキテクチャをベースとした独自の拡張アーキテクチャを持つ5段パイプライン方式のプロセッサであり、FPGA上で論理合成しCPUとして利用することができる。

柔軟なプログラム実現する 拡張命令

スマートな実装の５段パイプライン方式

当プロセッサは、CPUとして動作するのに最低限必要な命令セットに加え、プログラムを簡易・効率化する拡張命令を実装し、使用性の向上を図っている。また、パイプラインの各フェーズでの機能分担を深く吟味し、Group22プロセッサオリジナルの設計を各所に取り入れた。この２つの基本方針の元Group22プロセッサは設計され、高速・高使用性を実現している。

## 1.1 基本情報

レジスタ幅： 16 bit

レジスタ：　 0番から7番まで計８つ  
 （0番レジスタは０レジスタとして働き、値は常に0となる）

メモリ： 2048語（１語16bit）×２

## 1.2 設計情報

記述言語： Verilog HDL 2001

ターゲットFPGA： Altera Cyclone IV E　(EP4CE30F23I7)

設計環境： Altera Quartus II 13.0sp1

開発者　： 白石 竜也 / 谷 勇輝

開発期間： 2017年４月20日

　～ 2017年 6月2日

# 2 性能と特長

## 2.1 性能

最大動作周波数 : 73 MHz

FPGAボード(1)上での周波数： 130MHz

回路規模：

基本プログラムによる性能の目安

* 1024個のバブルソート  
   5,503,666サイクル / 44.0293 ms
* 1024個の単純クイックソート   
   94,002 サイクル / 0.7520 ms
* 1024個の改良クイックソート(+挿入ソート)  
   65,812 サイクル / 0.5062 ms

## 2.2 特長

### ５段パイプライン方式

5つのフェーズが並列して動作する５段パイプライン方式である。フォワーディングによる高速化の他、各フェーズに役割をうまく分担することで最適な動作を実現し、高速化を追及した。最終WBフェーズについては半クロックで動作し早いタイミングでレジスタ書込みを行う設計になっており、データハザード回避による高速化を実現している。

(1) PowerMedusa MU500-RX/RK FPGA：Altera社Cyclone IV EP4CE30F23I7N /クロック：20MHz発信機/拡張ボード：MU500-7SEG

### ハーバード・アーキテクチャ

命令用メモリとデータ用メモリを分離するハーバード・アーキテクチャを採用している。IFフェーズとMAフェーズが別々のメモリを保持し、同時に動作するためメモリアクセスの待機による時間遅延はない。

### 即値加算命令

SIMPLEアーキテクチャの命令セットに加え、即値加算命令ADDIが使用できる。ループ内で使用する変数等の１加算や１減算等に重宝し、基本プログラムであるクイックソートでは約10％のサイクル数削減が達成できた。

### 関数呼出命令

SIMPLEアーキテクチャの命令セットに加え、関数呼出命令JALと、戻り命令JRが使用できる。関数としてまとめて記述された命令列に即値ジャンプを使ってどこからでも移り、戻ることができる。また、ジャンプの際にレジスタの退避が行われるので、関数先では元のレジスタを気にすることなく動作ができる。この機能を利用してレジスタの擬似拡張を行うこともできる。

### 静的分岐予測

分岐命令時に、分岐判断を待たず次の命令を投機実行する静的分岐予測を実装している。分岐は常に成立しない方向に予測され、予測が外れた際には投機実行した命令を破棄し即座に分岐先命令にシフトする。分岐判断はEXフェーズに繰り上げて早い段階で行うため、分岐予測が外れた際に発生するストールは１クロックのみに抑えられている。

# 3 命令セット・アーキテクチャ

## 3.1 命令セット

命令セットを以下の表？？？に示す。

＜＜＜命令セットの表＞＞＞

## 3.2 分岐判断

## 3.3 関数呼出

関数呼出命令JALと戻り命令JRについては、レジスタの一時退避が行われる。

0番～5番までの６つのレジスタはローカルレジスタであり、関数呼出を行うと値の退避が実行され、戻り命令で復帰させることができる。一方6番、7番のレジスタはグローバルレジスタであり、関数呼出命令、戻り命令によって値の退避は行われない。

関数呼出直後、レジスタの値は呼び出し前の値と同一である。従って関数の引値としては0番～7番の全てのレジスタを使用することができる。戻り命令では、6番、7番以外のレジスタの値は関数呼出前の値に戻される。従って関数が返り値を持つ場合、それらは6番、7番のレジスタに格納する必要がある。

# 4 構造と動作

## 4.1 構造

### 入力

以下の表？？に示す４つの入力が存在し、適切な信号を外部から入力して使用する。

|  |  |  |
| --- | --- | --- |
| 入力信号名 | bit幅 | 役割 |
| clock | 1 | プロセッサを動作させるメインクロック |
| reset | 1 | プロセッサをリセットする信号 |
| exec | 1 | プロセッサを一時停止させる信号 |
| in | 16 | in命令によって読み込む16bit整数 |

### 出力

以下の表？？に示す２つの出力が存在し、外部と適切な接続を行い使用する。

|  |  |  |
| --- | --- | --- |
| 出力信号名 | bit幅 | 役割 |
| halt | 1 | プロセッサが動作終了したことを示す信号 |
| out | 16 | out命令によって出力される16bit整数 |

また、表？？に示す12の内部状態確認用出力も用意されており、必要に応じて使用することができる。

|  |  |  |
| --- | --- | --- |
| 出力信号名 | bit幅 | 役割 |
| PC\_next | 16 | IDモジュールに入力されるPCカウンタ+1の値 |
| instruction | 16 | IDモジュールに入力される命令コード |
| control | 6 | EXモジュールに入力される制御コード |
| Rs\_Ra | 16 | EXモジュールに入力される第１演算数 |
| Rd\_Rb | 16 | EXモジュールに入力される第２演算数 |
| stall | 1 | 分岐ハザードの発生通知信号 |
| mem\_in | 16 | MAモジュールに入力されるメモリに書き込まれるデータ |
| result | 16 | MAモジュールに入力される演算結果 |
| LDresult | 16 | WBモジュールに入力されるメモリ読み出し結果 |
| WBdata | 16 | WBフェーズでレジスタに書き込まれるデータ |
| WBadd | 16 | WBフェーズで書き込みを行うレジスタアドレス |
| RegW | 1 | WBフェーズのレジスタ書込み制御信号 |

## 4.2 動作

## 4.3 構造の詳細

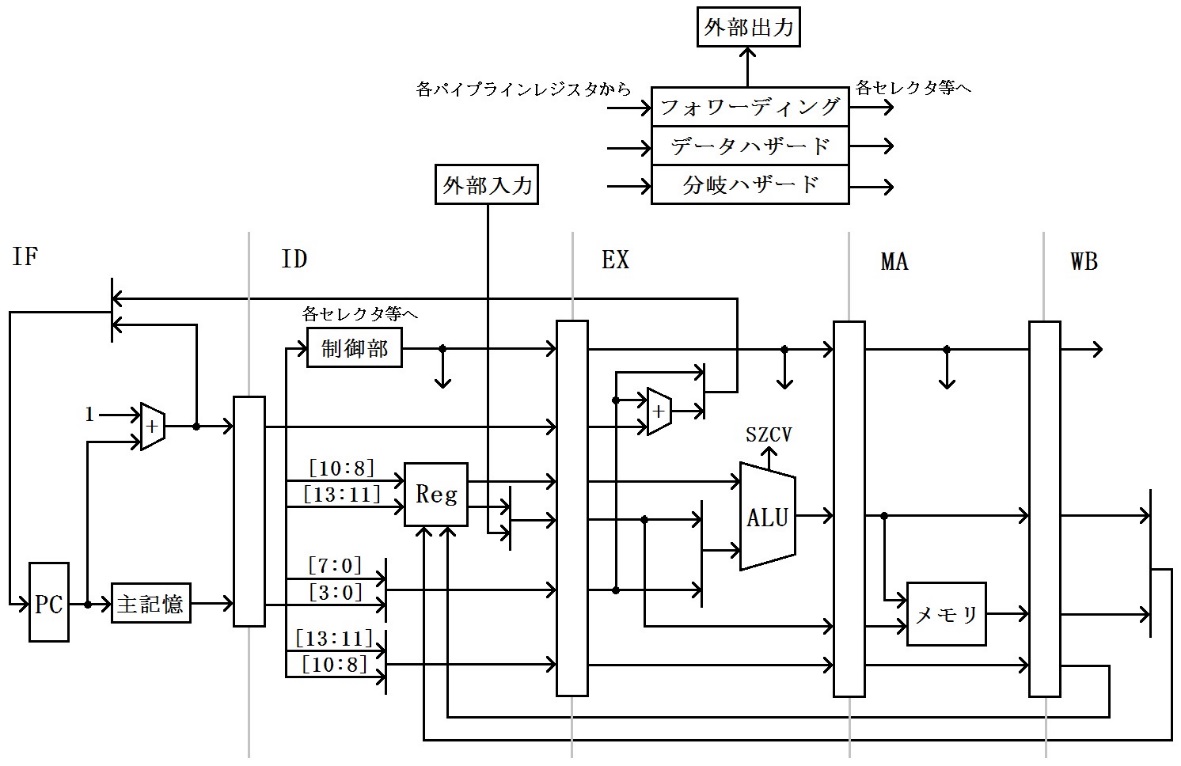
Group22プロセッサは、５段パイプラインの各フェーズを担当する次の５つの主要モジュールを持つ。

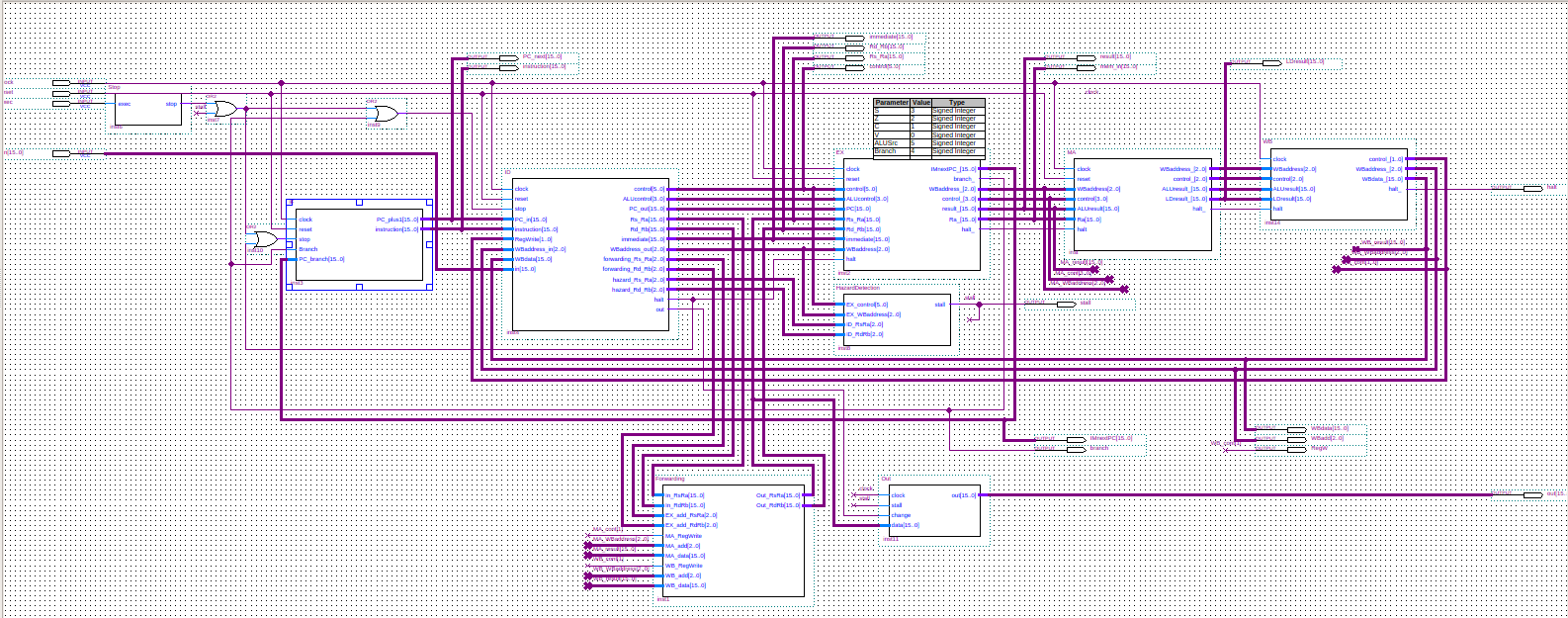
* IFモジュール 命令フェッチを行う
* IDモジュール 命令の解釈、レジスタ読出しを行う
* EXモジュール 各種演算、分岐判断を行う
* MAモジュール メモリ管理を行う
* WBモジュール レジスタ書込みを行う

また、パイプラインの動作を補助する次の４つの補助モジュールを持つ。

* Forwardingモジュール フォワーディング処理を行う
* HazardDetectionモジュール データハザード処理を行う
* Stopモジュール 外部からの一時停止信号を管理する
* Outモジュール 外部出力を管理する

Group22プロセッサは以上の9つのモジュールで構築される。図？？にブロック図を、図？？にトップレベルの回路の概観を示す。





Stop

IF

ID

EX

MA

WB

Forwarding

HazardDetection

Out

# 5 付録

## 5.1 テスト環境用モジュール

　付録として、FPGAボード PowerMedusa MU500-RX/RK 用の入出力・表示支援機能付きテスト環境用モジュールTestEnvironmentが利用可能である。プロセッサの入出力を各種用意された配線に接続することにより、簡単にボードに表示を行い、プロセッサの動作を確認することができる。

## 5.2 アセンブラ

　Group22プロセッサ命令セットに準拠したバイトコードを出力する専用アセンブラCUIAssemblerが利用できる。CUIAssebler.javaファイルが存在するディレクトリで以下のコマンドを入力すると起動し、指示に従ってファイル選択を行うことでアセンブリコードをバイトコードに変換できる。

　> java CUIAssembler