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中間報告)

機能設計仕様書

提出期限：5月11日

提出日：5月11日

第22班

**1029272870 　谷　勇輝**

目次

[1 コンポーネント分割と担当 2](#_Toc482265465)

[1.1 コンポーネント分割 2](#_Toc482265466)

[1.2 担当 2](#_Toc482265467)

[2 外部仕様 3](#_Toc482265468)

[2.1 EX 3](#_Toc482265469)

[2.1.1 概要 3](#_Toc482265470)

[2.1.2 構造 3](#_Toc482265471)

[2.1.3 動作 4](#_Toc482265472)

[2.2 MA 7](#_Toc482265473)

[2.2.1 概要 7](#_Toc482265474)

[2.2.2 構造 7](#_Toc482265475)

[2.2.3 動作 8](#_Toc482265476)

[2.3 WB 9](#_Toc482265477)

[2.3.1 概要 9](#_Toc482265478)

[2.3.2 構造 9](#_Toc482265479)

[2.3.3 動作 10](#_Toc482265480)

[2.4 TestEnvironment 11](#_Toc482265481)

[2.4.1 概要 11](#_Toc482265482)

[2.4.2 構造 11](#_Toc482265483)

[2.4.3 動作 12](#_Toc482265484)

[3 内部仕様 13](#_Toc482265485)

[3.1 EX 13](#_Toc482265486)

[3.1.1 構造 13](#_Toc482265487)

[3.1.2 動作 13](#_Toc482265488)

[3.2 MA 15](#_Toc482265489)

[3.2.1 構造 15](#_Toc482265490)

[3.2.2 動作 15](#_Toc482265491)

[3.3 WB 16](#_Toc482265492)

[3.3.1 構造 16](#_Toc482265493)

[3.3.2 動作 16](#_Toc482265494)

[3.4 Test Environment 17](#_Toc482265495)

[3.4.1 構造 17](#_Toc482265496)

[3.4.2 動作 18](#_Toc482265497)

# 1 コンポーネント分割と担当

## 1.1 コンポーネント分割

プロセッサは、最上位レベルの分割として、図１に示す8つのコンポーネントで構成される。各コンポーネント内部の設計（次レベルの分割等）はそれぞれの設計担当者が行う。

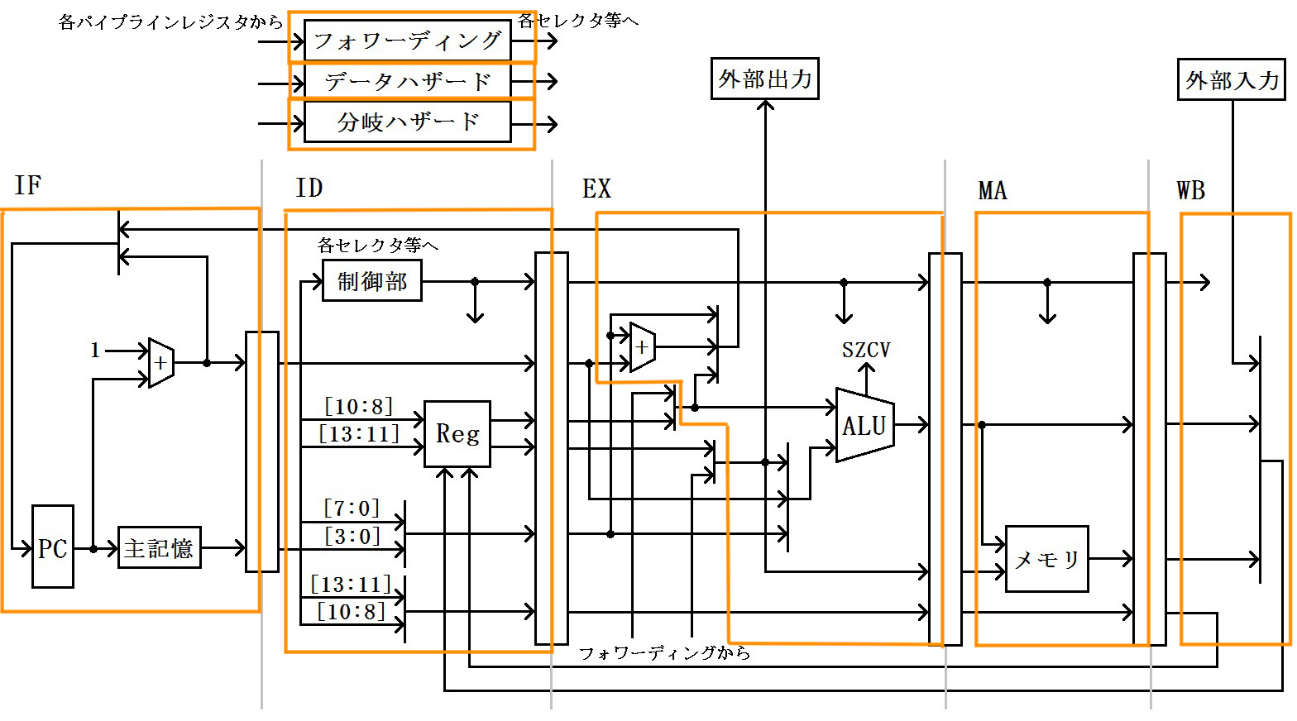


図 1　コンポーネント分割

## 1.2 担当

2017年5月11日現在、私が設計・実装を担当したプロセッサのコンポーネントは以下のとおりである。

* EX
* MA
* WB

また、本体プロセッサとは別に、以下の設計・実装も担当した。

* FPGAボード入出力用環境　TestEnvironment

# 2 外部仕様

## 2.1 EX

### 2.1.1 概要

EXモジュールは、演算を担当するEXフェーズの一連の機能を提供する。

上流モジュールIDからの制御に応じて同期的に動作し、内包するALU、シフタを用いて下流モジュールMAへ演算結果を渡す。また、分岐の判断を行いIFに結果を伝搬する。

### 2.1.2 構造

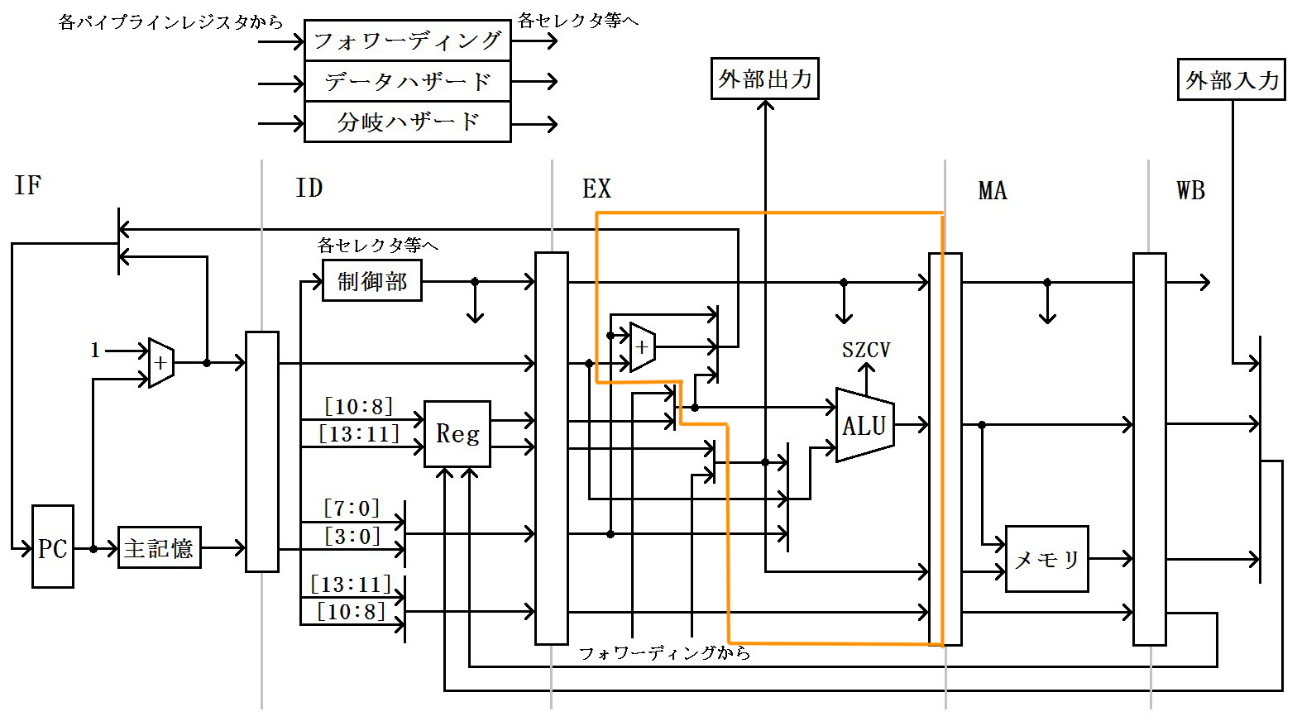
全体内の位置づけを図２に、入力構造を表１に、出力構造を表２に示す。

図 2　EXコンポーネントの位置づけ

表 1　入力構造 (EX)

|  |  |  |  |
| --- | --- | --- | --- |
| 入力信号名 | bit幅 | 接続 | 内容 |
| clock | 1 |  | クロック信号 |
| reset | 1 |  | リセット信号（負論理） |
| PC | 16 | ID | プログラムカウンタ値+1 |
| WBaddress | 3 | ID | レジスタ書き込みアドレス (or 分岐種類) |
| control | 6 | ID | 制御信号 |
| ALUcontrol | 4 | ID | 演算器の機能制御コード |
| immediate | 16 | ID | 即値（or 演算器の第２入力） |
| Rs\_Ra | 16 | ID | 演算器の第２入力（or メモリの指定アドレス） |
| Rd\_Rb | 16 | ID | 演算器の第１入力 |

表 2　出力構造 (EX)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 出力信号名 | bit幅 | 接続 | 同期 | 内容 |
| IMnextPC\_ | 16 | IF | × | 分岐先プログラムカウンタ値 |
| branch\_ | 1 | IF | × | 分岐判定 (0:分岐無し1:分岐有り) |
| WBaddress\_ | 3 | MA | ○ | レジスタ書き込みアドレス |
| control\_ | 4 | MA | ○ | 制御信号 |
| result\_ | 16 | MA | ○ | 演算結果 |
| Ra\_ | 16 | MA | ○ | メモリの指定アドレス |
| ConditonCode\_ | 4 | MA | ○ | コンディションコード (S,Z,C,V) |

### 2.1.3 動作

#### 分岐処理　(ID→EX→IF)

* PC (プログラムカウンタ+1の値) + immediate (即値) を

IMnextPC\_(分岐先PCアドレス) として出力

* control [4] (分岐制御信号Branch) が１(分岐命令)であり、直前の演算のコンディションコードの状態がWBaddress(分岐種類コードcond)の示す分岐条件(表３)に合致するならば、branch\_(分岐判定)に1を出力

表 3　分岐コードと分岐条件の対応

|  |  |  |
| --- | --- | --- |
| 分岐コード | 分岐条件 | 主な使用法 |
| 000 | Zが1 | 直前の演算結果が０ |
| 001 | S XOR V が1 | 直前の減算結果が理論的に負 |
| 010 | Zが1または  S XOR V が1 | 直前の減算結果が理論的に0以下 |
| 011 | Zが0 | 直前の演算結果が０でない |

#### 演算処理 (ID→EX→MA)

* control[5] (第２入力制御信号ALUSrc)が0ならば演算に使用する第２入力としてRs\_Raを選択
* control[5] (第２入力制御信号ALUSrc)が1ならば演算に使用する第２入力としてimmediate (即値)を選択
* 第１入力をRd\_Rb、第２入力を選択した信号として、ALUControl (演算器機能制御コード) の値に従って演算を行い(表４)、clockと同期して演算結果をresult\_に出力

表 4　演算器の機能制御コード

|  |  |  |
| --- | --- | --- |
| 機能コード | 動作 | 代用コード |
| 0000 | 加算 (+) |  |
| 0001 | 減算 (- ) | 0101 |
| 0010 | 論理積 (AND) |  |
| 0011 | 論理和 (OR) |  |
| 0100 | 排他的論理和 (XOR) |  |
| 0110 | 第２入力を出力 | 1100 ,1101 ,1111 |
| 1000 | 左論理シフト |  |
| 1001 | 左循環シフト |  |
| 1010 | 右論理シフト |  |
| 1011 | 右算術シフト |  |

* 演算結果に従って４つのコンディションコード(表５)を設定し、clockと同期してConditionCode\_に出力

表 5　コンディションコード

|  |  |  |
| --- | --- | --- |
| コード | 内容 | 例外 |
| S | 負ならば1 |  |
| Z | 0ならば1 |  |
| C | 桁上げがあれば1 | 論理演算、第２入力出力、左循環シフト演算では０  その他のシフト演算では最後にシフトされた値で判定 |
| V | オーバーフローで1 | シフト演算では0 |

* WBaddress (レジスタ書き込みアドレス)をclockと同期してWBaddress\_にそのまま出力
* Rs\_Ra(メモリの書き込みアドレス)をclockと同期してRa\_にそのまま出力
* controlのうち、使用していない下4bitをclockと同期してcontrol\_に出力

#### リセット機構（外部→EX→MA）

* clock同期時resetが0であれば全ての同期式出力を0にする

## 2.2 MA

### 2.2.1 概要

MAモジュールは、データメモリの書き込み、読み出しを管理するMAフェーズの一連の機能を提供する。

上流モジュールEXを伝播してきた制御信号に応じて同期的に動作し、内包するメモリを操作して下流モジュールWBに結果を伝搬する。

### 2.2.2 構造

全体内の位置づけを図３に、入力構造を表６に、出力構造を表７に示す。

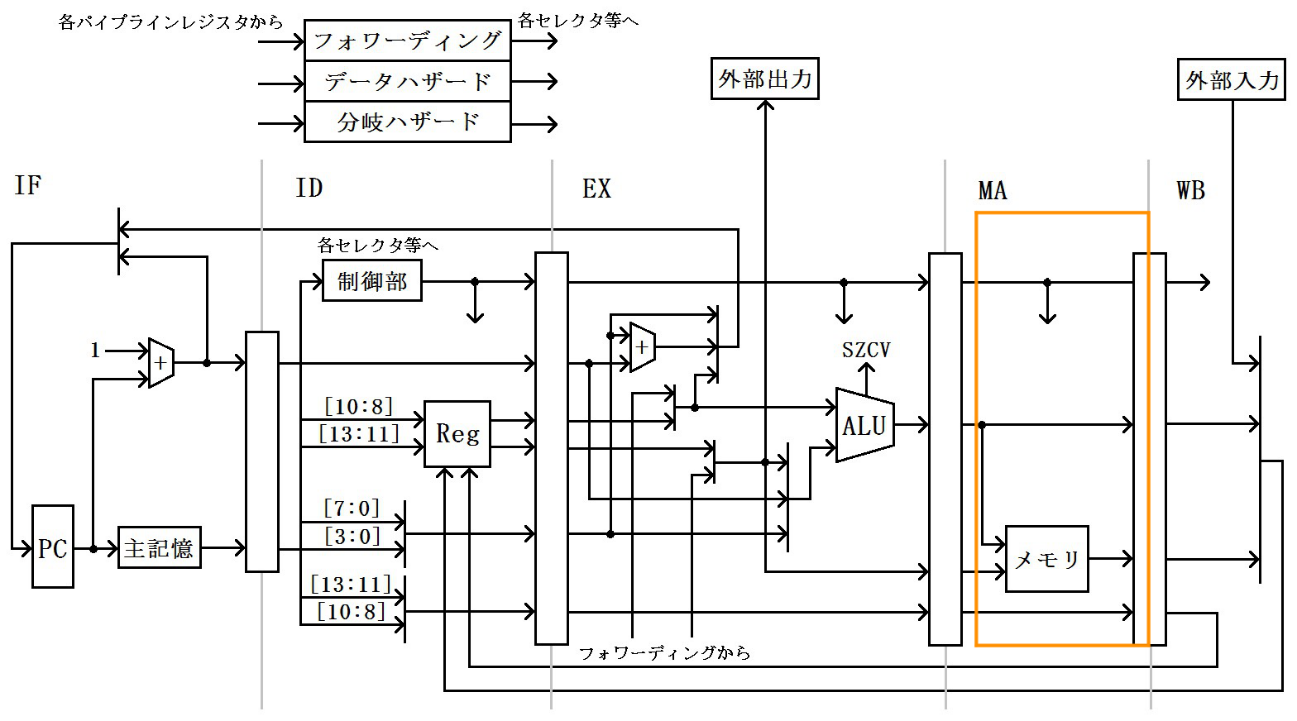


図 3　MAコンポーネントの位置づけ

表 6　入力構造 (MA)

|  |  |  |  |
| --- | --- | --- | --- |
| 入力信号名 | bit幅 | 接続 | 内容 |
| clock | 1 |  | クロック信号 |
| reset | 1 |  | リセット信号（負論理） |
| WBaddress | 3 | EX | レジスタ書き込みアドレス |
| control | 4 | EX | 制御信号 |
| ALUresult | 16 | EX | 演算の結果 |
| Ra | 16 | EX | メモリの指定アドレス |

表 7　出力構造 (MA)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 出力信号名 | bit幅 | 接続 | 同期 | 内容 |
| WBaddress\_ | 3 | WB | ○ | レジスタ書み込みアドレス |
| control\_ | 2 | WB | ○ | 制御信号 |
| ALUresult\_ | 16 | WB | ○ | 演算結果 |
| LDresult\_ | 16 | WB | ○ | メモリ読み出し結果 |

### 2.2.3 動作

#### メモリ処理 (EX→MA→WB)

* control[3](メモリ読み出し制a御信号MemRead) が1ならばメモリからRa(メモリ指定アドレス)番地に格納されているデータを読み出し、clockと同期してLDresult\_に出力
* control[2](メモリ書き込み制御信号MemWrite) が1ならばメモリのRa(メモリ指定アドレス)番地にALUresultを格納
* WBaddress (レジスタ書き込みアドレス)をclockと同期してWBaddress\_にそのまま出力
* ALUresult (演算結果)をclockと同期してALUresult\_にそのまま出力
* controlのうち、使用していない下2bitをclockと同期してcontrol\_に出力

#### リセット機構（外部→MA→WB）

* clock同期時resetが0であれば全ての同期式出力を0にする

## 2.3 WB

### 2.3.1 概要

WBモジュールは、レジスタ書き込みを行うWBフェーズの機能の一部を提供する。

上流モジュールMAを伝播してきた制御信号に応じて動作し、IDモジュールにメモリ書き込みを依頼する。クロックと書き込みの管理はIDモジュールに委任する。

### 2.3.2 構造

全体内の位置づけを図４に、入力構造を表８に、出力構造を表９に示す。

図 4　WBコンポーネントの位置づけ

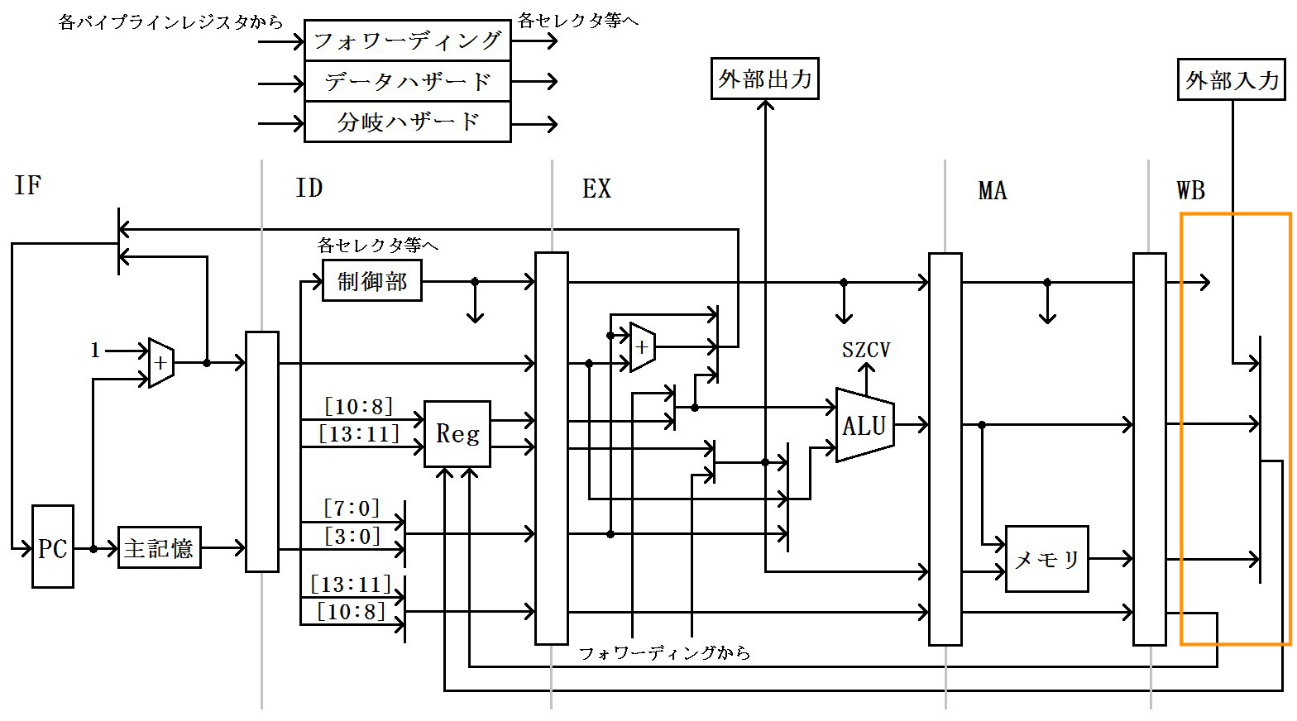


表 8　入力構造 (WB)

|  |  |  |  |
| --- | --- | --- | --- |
| 入力信号名 | bit幅 | 接続 | 内容 |
| clock | 1 |  | クロック信号（拡張時用） |
| WBaddress | 3 | MA | レジスタ書き込みアドレス |
| control | 2 | MA | 制御信号 |
| ALUresult | 16 | MA | 演算結果 |
| LDresult | 16 | MA | メモリ読み出し結果 |

表 9　出力構造 (WB)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 出力信号名 | bit幅 | 接続 | 同期 | 内容 |
| RegWrite\_ | 1 | ID | × | レジスタ書き込み制御 |
| WBaddress\_ | 3 | ID | × | レジスタ書み込みアドレス |
| WBdata\_ | 16 | ID | × | レジスタ書き込みデータ |

### 2.3.3 動作

#### レジスタ書き込みデータの選択 (MA→WB→ID)

* control[0] (レジスタ書き込みデータ制御信号MemtoReg)が0ならば、ALUresult(演算結果)をWBdata\_に出力
* control[0] (レジスタ書き込みデータ制御信号MemtoReg)が1ならば、LDresult(メモリ読み出し結果)をWBdata\_に出力
* WBaddress (レジスタ書き込みアドレス)をWBaddress\_にそのまま出力
* controlのうち、使用していない1bit (第1番信号)をRegWrite\_に出力

## 2.4 TestEnvironment

### 2.4.1 概要

FPGAボード (PowerMedusa MU500-RX/RK, 拡張MU500-7SEG)のほぼ全ての入出力を簡易に利用できる環境を提供する。主な特徴は以下の３点である。

* 7segLEDは、信号線を擬似出力に接続するだけで16進数を表示可能。
* 拡張ボードを含む全ての7segLEDについて、任意のパターンを独立に表示可能。（72個の7segLEDが全て使用可能となる）
* 全ての入力と拡張LEDランプを除く全ての出力が利用可能

各種入力はFPGAボードの入力に、各種出力はFPGAボードの出力に対応している。接続対象のモジュールをTestEnvironmentモジュール(以下、環境モジュールと示す)の中に配置し、接続対象モジュールの入出力を環境モジュールの入出力や擬似入出力に接続することで、あたかも外部の入出力に直接繋いだような状態とできる。

### 2.4.2 構造

入力構造を表10に、出力構造を表11に示す。内部の擬似入出力については内部仕様に記す。

表 10　入力構造（TestEnvironment）

|  |  |  |  |
| --- | --- | --- | --- |
| 入力名 | 幅 | 内容 | 接続ピン |
| clock | 1 | 20MHz固定クロック | A12 |
| clock2 | 1 | 可変クロック | B12 |
| dipSW\_A | 8 | ディップスイッチ群Ａ | G11,G10,F10,E10,D10,C10,B10,A10 |
| dipSW\_B | 8 | ディップスイッチ群Ｂ | F13,E13,D13,C13,B13,A13,F11,E11 |
| rotSW\_A | 4 | ローテーションスイッチＡ | F14,E14,B14,A14 |
| rotSW\_B | 4 | ローテーションスイッチＢ | D15,C15,B15,A15 |
| pushSW\_A | 5 | プッシュスイッチ群Ａ行 | A16,H15,G15,F15,E15 |
| pushSW\_B | 5 | プッシュスイッチ群Ｂ行 | A17,G16,F16,E16,B16 |
| pushSW\_C | 5 | プッシュスイッチ群Ｃ行 | B18,A18,D17,C17,B17 |
| pushSW\_D | 5 | プッシュスイッチ群Ｄ行 | A20,D19,C19,B19,A19 |

表 11　出力構造（TestEnvironment）

|  |  |  |  |
| --- | --- | --- | --- |
| 出力名 | 幅 | 内容 | 接続ピン |
| LED7segA\_D | 8 | UIボード7segLED,A~D | A3,B6,A6,A5,B4,B3,A4,B5 |
| LED7segE\_H | 8 | UIボード7segLED,E~H | C6,F7,E7,C7,B7,D6,A7,D7 |
| slctA\_D | 4 | UI 7seg セレクタA~D | E6,E5,C4,C3 |
| slctE\_H | 4 | UI 7seg セレクタE~H | G7,G8,G9,H10 |
| LED | 8 | UIボード LEDランプ群 | F9,E9,B9,A9,F8,C8,B8,A8 |
| BZ | 1 | UIボード　ブザー | B20 |
| LED7ep0 | 8 | 拡張ボード 7segLED 0列 | Y6,AB5,W6,AB4,AA5,AA4,V5 |
| LED7ep1 | 8 | 拡張ボード 7segLED 1列 | U8,AA7,T8,V7,Y7,U7,V6 |
| LED7ep2 | 8 | 拡張ボード 7segLED 2列 | V9,AB8,U9,W8,AA8,V8,Y8,AB7 |
| LED7ep3 | 8 | 拡張ボード 7segLED 3列 | AB10,W10,AA10,AB9,V10,AA9,U10,T9 |
| LED7ep4 | 8 | 拡張ボード 7segLED 4列 | V13,V12,U13,V11,U12,U11,T12,Y10 |
| LED7ep5 | 8 | 拡張ボード 7segLED 5列 | V14,R14,U14,Y13,AB13,W13,AA13,T13 |
| LED7ep6 | 8 | 拡張ボード 7segLED 6列 | W15,T15,V15,AA14,R15,W14,AB14,T14 |
| LED7ep7 | 8 | 拡張ボード 7segLED 7列 | AB16,U16,AA16,AB15,T16,AA15,R16,U15 |
| slctEp | 8 | 拡張ボード 7seg セレクタ | V16,AA17,W17,AB17,Y17,AB19,AA18,AA19 |

### 2.4.3 動作

#### FPGAボードの使用（ボード入力→TestEnvironment→ボード出力）

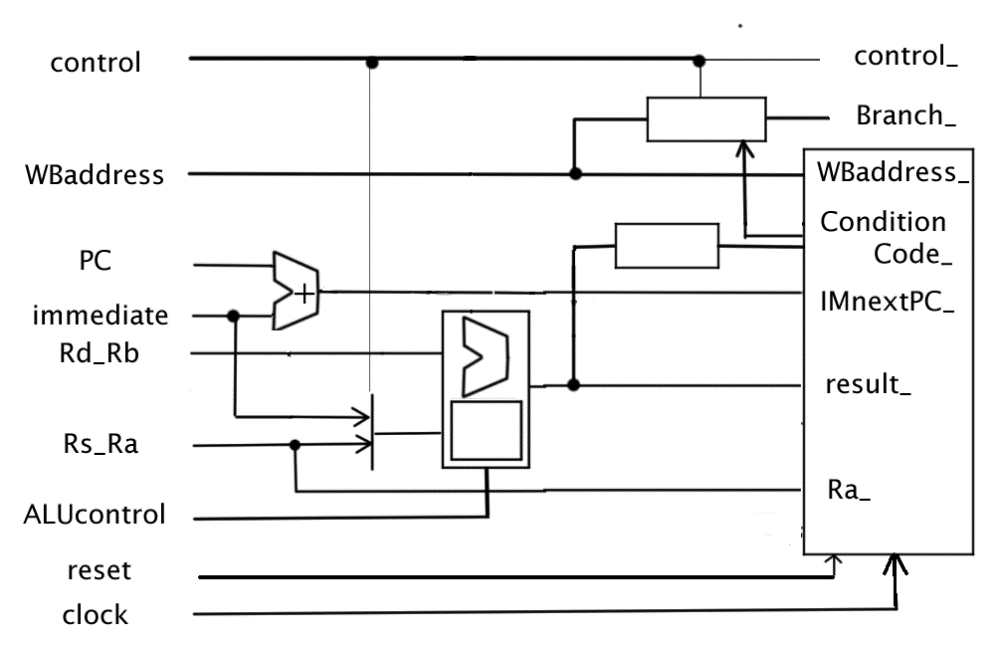
* ボードの入力を環境モジュール内に取り込み、作成した内部機構の動作に応じてボードの出力に表示する。

# 3 内部仕様

## 3.1 EX

### 3.1.1 構造

ブロック図を図5に示す。



①

②

③

④

⑤

図 5　EX ブロック図

### 3.1.2 動作

#### ①　分岐判断 ( control, WBaddress → Branch\_)

control[4]==1 ∧ *judge*(WBaddress, ConditionCode\_)

→ Branch\_ = 1

bool関数judgeを表12に示す。(ConditionCode\_はCCと略記)

表 12　関数judge

|  |  |  |
| --- | --- | --- |
| WBaddress(分岐コード) | 関数値 | 内容 |
| 000 | CC[2] | Zが1でtrue |
| 001 | CC[3] ^ CC[0] | S XOR Vが1でtrue |
| 010 | CC[2]||(CC[3]^CC[0]) | 上２つのいずれかでtrue |
| 011 | !CC[2] | Zが０でtrue |

#### ②　分岐先アドレス計算（PC, immediate → IMnextPC ）

IMnextPC = PC + immediate

#### ③　第２入力選択（immediate, Rs\_Ra, control → In2）

control == 0 → In2 = Rs\_Ra

control == 1 → In2 = immediate

#### ④　演算 (Rd\_Rb, In2, ALUcontrol → result\_)

外部仕様の表４（演算器の機能制御コード）に従う演算@の元で、

　　result = Rd\_Rb @ In2

演算器は、加算、減算、各種論理演算を実装した**ALU**と表４内にある４種のシフト演算を提供する**バレルシフタ**から成る。

#### ⑤　コンディションコード設定（result → ConditionCode\_）

外部仕様の表５（コンディションコード）に従って設定する

　ConditonCode\_[3] = S

　ConditionCode\_[2] = Z

ConditionCode\_[1] = C

ConditionCode\_[0] = V

設定の詳細はSIMPLEの仕様による。

#### その他

control\_ = control [3..0]

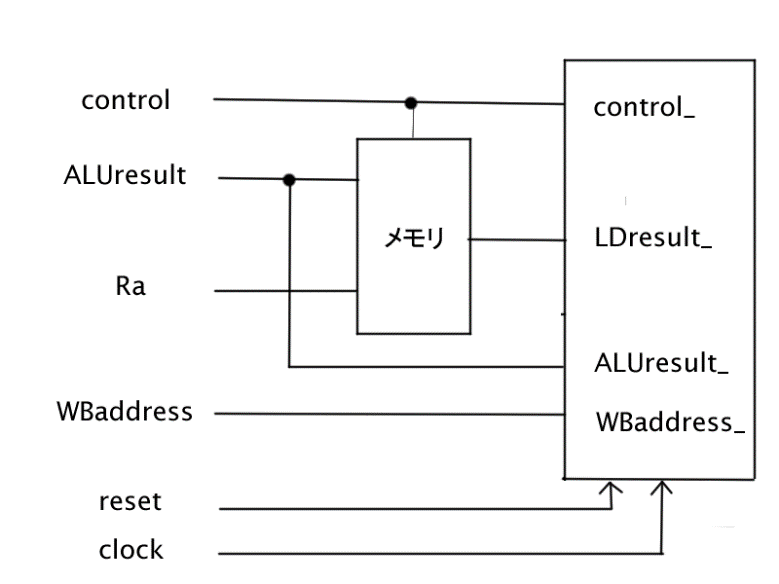
WBaddress\_ = WBaddress

Ra\_ = Rs\_Ra

## 3.2 MA

### 3.2.1 構造

ブロック図を図６に示す。



①

②

図 6　MA ブロック図

### 3.2.2 動作

#### ①メモリ書き込み（control, Ra, ALUresult → Mem）

control [3] ==1 → （ ALUresult ⇒　Mem(Ra) ）

#### ②メモリ読み出し（control, Ra → LDresult\_）

control [2] ==1 →　LDresult\_ = Mem(Ra)

#### その他

control\_ = control [1..0]

ALUresult\_ = ALUresult

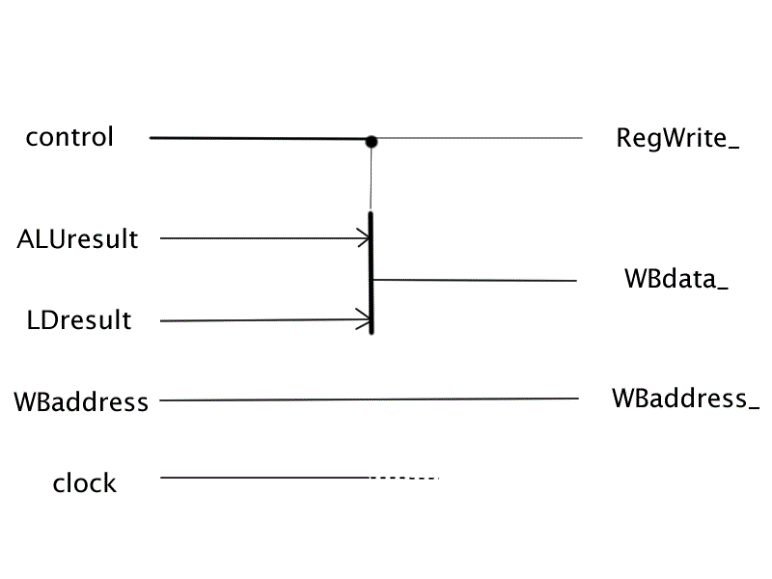
WBaddress\_ = WBaddress

## 3.3 WB

### 3.3.1 構造

ブロック図を図７に示す。

図 7　WB ブロック図



①

図 8　WB ブロック図

### 3.3.2 動作

#### ①　データの選択（ALUresult, LDresult, control → WBdata\_）

control [0] == 0　→　WBdata\_ = ALUresult

control [0] == 1　→　WBdata\_ = LDresult

#### その他

RegWrite\_ = control [1]

WBaddress\_ = WBaddress

## 3.4 Test Environment

### 3.4.1 構造

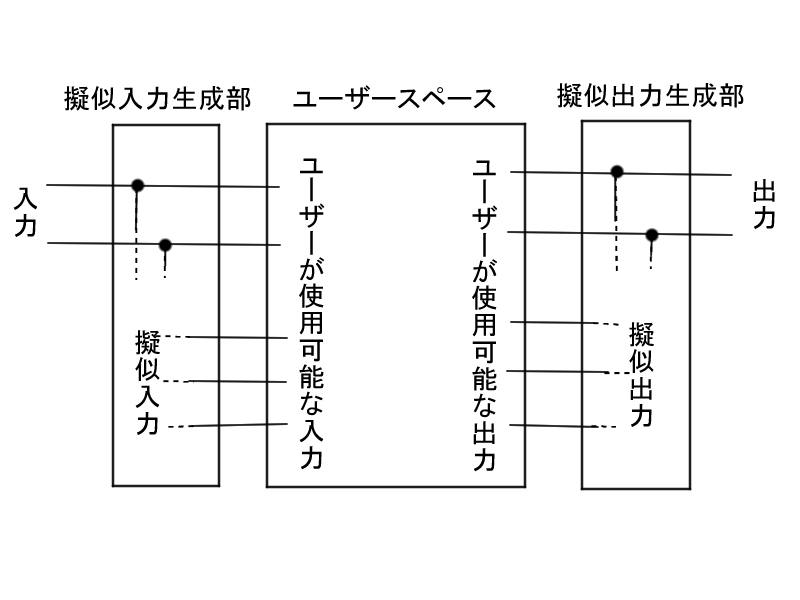
大きく分けて、①擬似入力生成部、②ユーザースペース、③擬似出力生成部の3つの領域からなる。環境モジュールの利用者は、②内部のみを編集し、実行を行う。（図９）

図 9　TestEnvironmentの主な構成

#### 擬似入力

環境モジュール内にて入力と同等の役割をもつ擬似入力を表13に示す。擬似入力のビット幅は全て１bitである。

表 13　擬似入力

|  |  |  |
| --- | --- | --- |
| 擬似入力名 | 内容 | 実装 |
| clock\_x4 | 1/４倍速クロック信号 | 4進カウンタにclockを通す |
| clock\_x8 | 1/８倍速クロック信号 | 8進カウンタにclockを通す |
| clock\_x10 | 1/10倍速クロック信号 | 10進カウンタにclockを通す |
| clock\_x100 | 1/100倍速クロック信号 | 100進カウンタにclockを通す |
| clock\_x1000 | 1/1000倍速クロック信号 | 1000進カウンタにclockを通す |
| clock\_x2p16 | 1/(2の16乗)倍速クロック信号 | 2^16進カウンタにclockを通す |
| vdd | 電源 | 恒久的に値が1のレジスタ |

#### 擬似出力

環境モジュール内にて出力と同等の役割をもつ擬似出力を表14に示す。擬似出力のビット幅は全て８bitである。

|  |  |  |
| --- | --- | --- |
| 擬似出力名 | 内容 | 実装 |
| LED\_A | 7segLED (UI)  の独立表示 | LED4setモジュールを使用して時間差表示 |
| …..LED\_H |
| LED\_exA[0..7] | 7segLED (拡張)  の独立表示 | LED8setモジュールを使用して時間差表示 |
| …..LED\_exH[0..7] |

### 3.4.2 動作

#### LED4setモジュール（7segLEDの独立表示）

7segLEDは、４つ（拡張ボードは８つ）の表示盤が組になっており、組につき１つの表示パターンしか指定できない。そのため通常は同じ組の表示盤には同じパターンしか表示できない。しかし、短い時間間隔で表示盤を指定するセレクタと表示パターンを切り替えていくことにより、あたかも同じ組の表示盤に別のパターンが表示されているように見せることはできる。この機構を提供するのがLED4setモジュールである。

* LED4setは、同組の7segLEDに表示したい４つの表示パターンとclockを入力すると、1clockごとに切り替わる１つの表示パターンと同じタイミングで切り替わるセレクタ信号を出力する。
* TestEnvironmentでは入力クロックとして5MHzクロックである擬似入力clock\_x4を使用する。