# 方針設計仕様書

19 班 佐竹誠 (1029283837)・中村敏行 (1029297986)

2018/5/10 提出

# 1　概要など

## 1.1　概要

SIMPLE/B に命令セットの追加・変更及び高速化したハードウェアを設計する。

## 1.2　目的

作成したマイクロコンピュータ上でソートプログラムを実際に動作させ、性能を確認する。

## 1.3　目標

ソートプログラムが正常に動くプロセッサを設計する。またその性能を向上させる。

## 1.4　設計方針

SIMPLE/B を拡張して設計する。具体的には即値 ADD、即値 SUB 命令の追加、出力命令の強化、条件分岐の 1 命令化、フェーズの並列実行を実装する。

## 1.5　特徴

命令の追加、変更によりクロックサイクル数の低下、出力性能の向上が見込める。またフェーズの並列実行により、命令サイクルの短縮ひいては高速化も見込める。

# 2　命令セット・アーキテクチャ

SIMPLE アーキテクチャに以下の変更を加える。
(a) 即値 ADD、即値 SUB の追加
(b) 出力命令の強化
(c) 条件分岐の 1 命令化

## 2.1　各変更後の命令の命令形式

それぞれの命令形式とフィールドの意味は以下のとおりである。
(a) 即値 ADD、即値 SUB

- $I_{15:14}$ (op1) 操作コード (11)
- $I_{13:11}$ (Rs) ソース・レジスタ番号
- $I_{10:8}$ (Rd) デスティネーション・レジスタ番号
- $I_{7:4}$ (op3) 操作コード (0111:ADDi,1110:SUBi)
- $I_{3:0}$ (d) 即値

(b) 出力命令

- $I_{15:14}$ (op1) 操作コード (11)
- $I_{13:11}$ (Rs1) ソース・レジスタ番号 1
- $I_{10:8}$ (Rs2) ソース・レジスタ番号 2
- $I_{7:4}$ (op3) 操作コード (1101)
- $I_{3:0}$ (d) 即値

(c) 条件分岐

- $I_{15:14}$ (op1) 操作コード (10)
- $I_{13:11}$ (op2) 操作コード (001:BE,010:BLT,101:BLE,110:BNE)
- $I_{10:8}$ (Rs1) ソース・レジスタ番号 1
- $I_{7:5}$ (Rs2) ソース・レジスタ番号 2
- $I_{4:0}$ (d) 変位

### 2.2 各変更の詳細

1. 即値 ADD、即値 SUB の追加演算命令に即値 ADD、即値 SUB を追加する。SIMPLE/B で reserved となっている操作コードである 0111 と 1110 にそれぞれ即値 ADD、即値 SUB を割り当てる。レジスタ Rs と変位 d の加算または減算の結果を Rd に格納する。
2. 出力命令の強化出力命令時にレジスタ内容を 2 つ同時に表示できるようにする。操作コードは SIMPLE/B で OUT に割り当てられているものをそのまま使用する。レジスタ Rs1,Rs2 の値を 7SEG LED を 8 つ用いてそれぞれ 16 進数 4 桁で表示する。
3. 条件分岐の 1 命令化 Rs1 と Rs2 の値を用いて条件分岐を行い、プログラムカウンタに変位 d を足した値にプログラムカウンタを書き換える。条件分岐命令は op2 の reserved となっている箇所を用いて BE,BLT,BLE,BNE の 4 命令を実装する。ただしその 1 命令化した条件分岐命令の名称はそれぞれ BE',BLT',BLE',BNE' とする。各命令の詳しい分岐条件は SIMPLE 設計資料表 4:SIMPLE に準拠する。

## 3 構造と動作

### 3.1 構造

SIMPLE 設計資料図 2:SIMPLE/B のブロック図を基に、後述のフェーズ並列処理のためにパイプラインレジスタ、分岐専用 ALU を追加した構造になっている。ブロック図の全体を図 2 に示す。黄色く塗られたモジュールがパイプラインレジスタであり、細部を確認するためにパイプラインレジスタごとに区切って拡大したブロック図も合わせて図 2 から図 6 に示す。

### 3.2 動作

ハードウェアの動作を示すものとして、フェーズ・フロー・チャートを図 7 に、データ・フロー・チャートを図 8 から図 12 に示す。

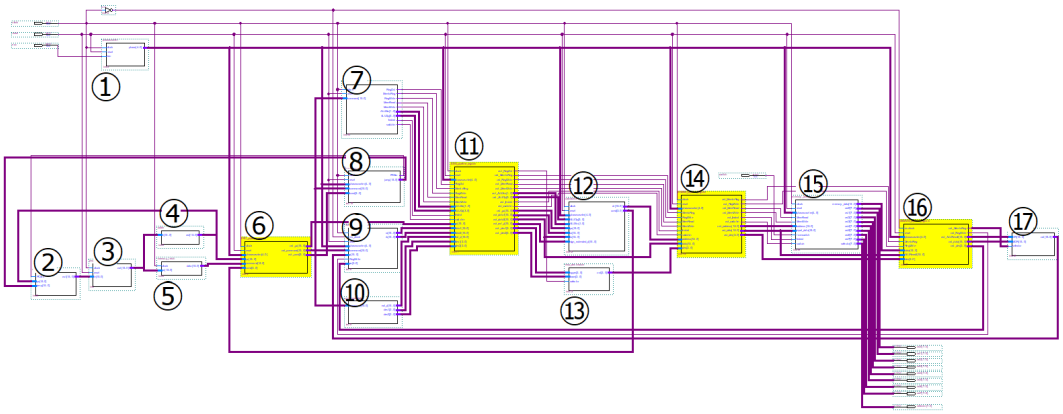


図 1　設計するハードウェアのブロック図の全体

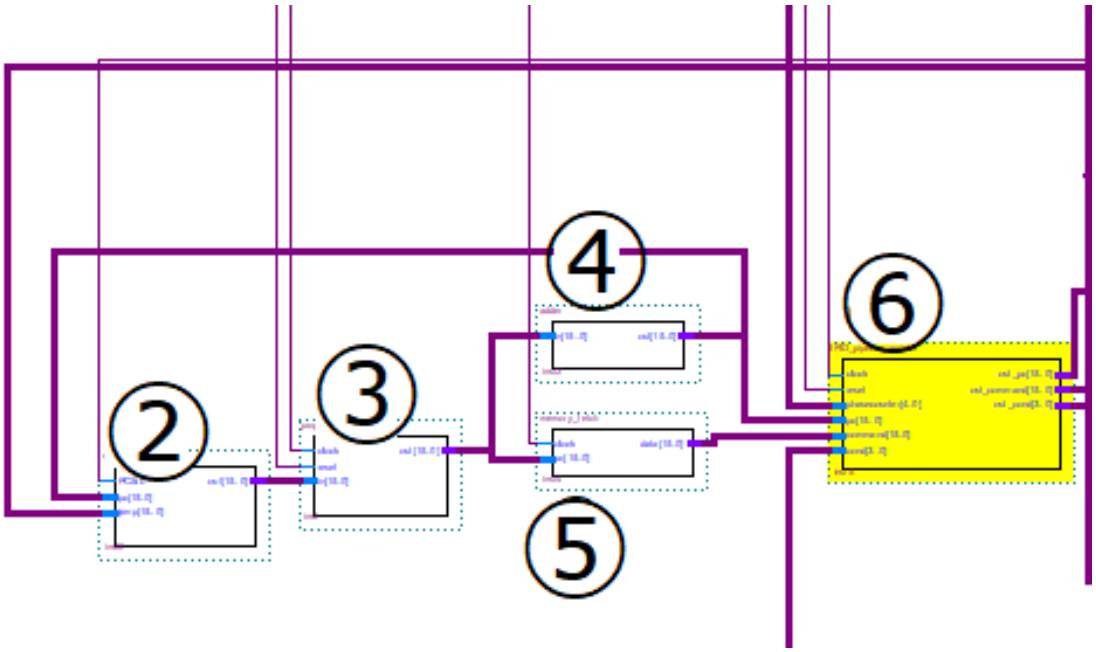


図 2　設計するハードウェアのブロック図の phase1 部分

SIMPLE/B からの変更点としては、後述のフェーズ並列実行のためにフェーズ 2 で分岐処理を完了させていることである。これにより制御ハザードを起こさずに実行を進められるようになっている。

# 4　高速化/並列処理の方式

## 4.1　高速化

SIMPLE 設計資料 5.1 フェーズの並列実行内に例 2 としてあげられている p1/p3 と p2/p5 の並列実行を実現する。

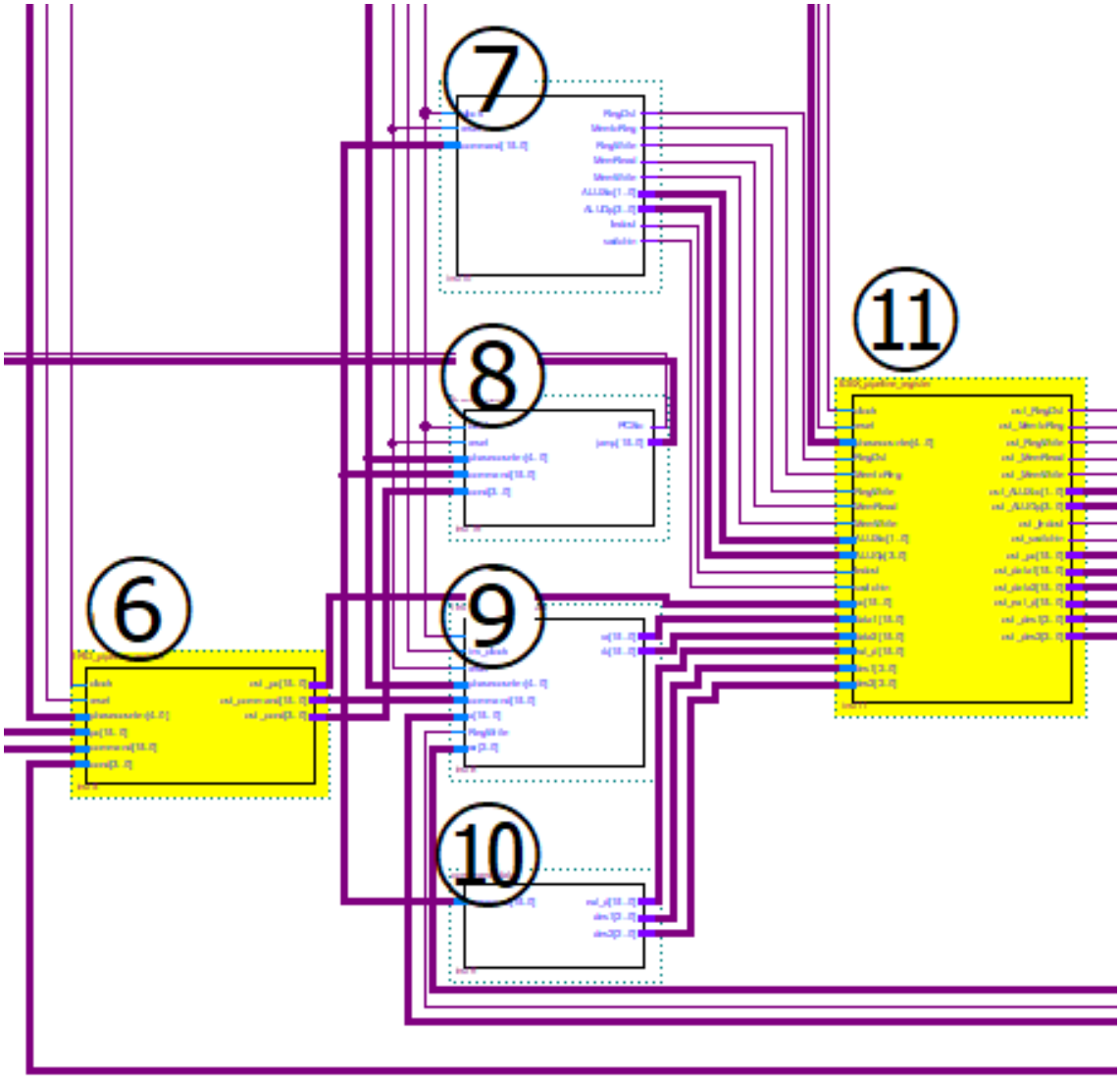


図 3　設計するハードウェアのブロック図の phase2 部分

動作の項目でも述べたようにフェーズ 2 で分岐処理を完了させることでフェーズ 1 とフェーズ 3 の並列を実現し、フェーズ 2 のレジスタからデータの読み取りを invclock により制御することでクロック内の後半に行い、フェーズ 5 のレジスタへの書き込みをクロック内の前半で行うことでフェーズ 2 とフェーズ 5 の並列を実現する。

## 4.2　並列処理

今回は実装しない。

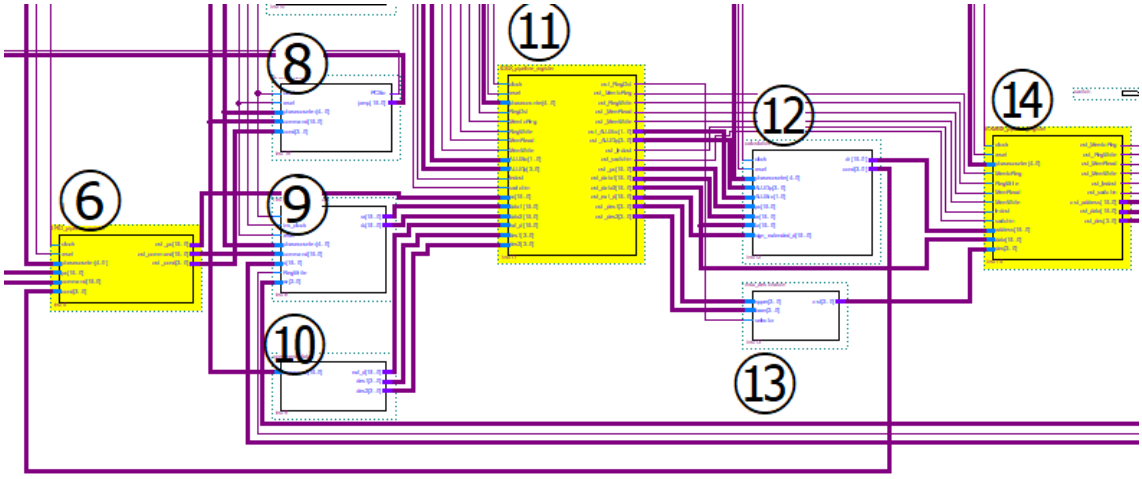


図 4　設計するハードウェアのブロック図の phase3 部分

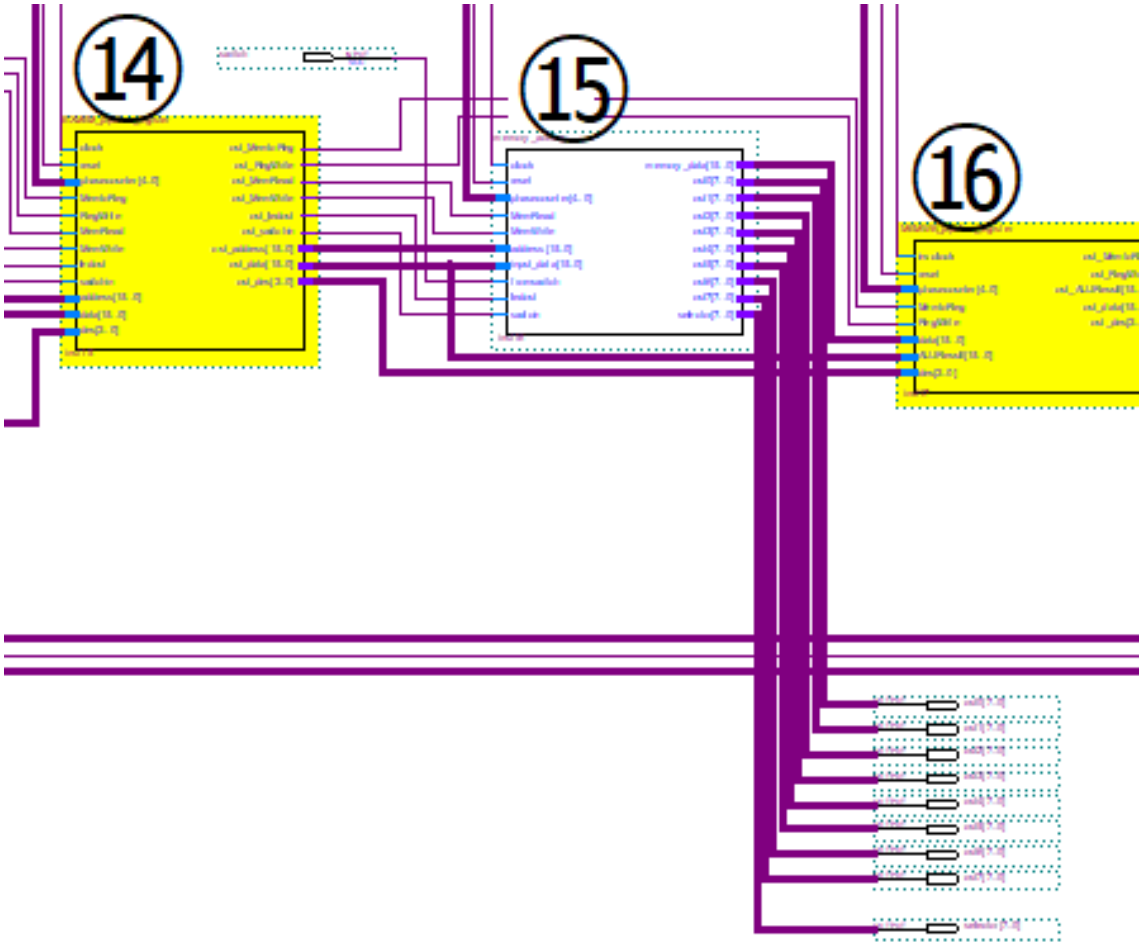


図 5　設計するハードウェアのブロック図の phase4 部分

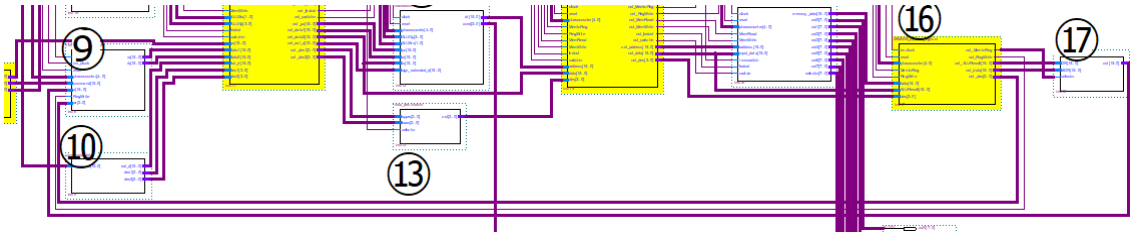


図 6　設計するハードウェアのブロック図の phase5 部分

## 5　性能/コストの予測

SIMPLE/B 実装段階においてクロック制約を 100MHz にすると最大周波数が 103.04MHz、ロジックエレメント数が 1010 個となっている。この事実と資料のソート速度コンテストのデータを参考にして性能と回路規模を予測する。性能は高速化により約 2 倍になるが最も処理に時間がかかるであろう 1 命令化した分岐を実装できていない状態なので、130MHz 程度になると予測される。、回路規模は分岐処理に使用するコンポーネントとパイプラインレジスタの追加が主なコスト増大であるので、ロジックエレメント数は 1500 個程度になることが予測される。

| 命令 | P1 | P2 | P3 | P4 | P5 |
|---|---|---|---|---|---|
| ADD | ・programconter→access_memory<br>・programconter→adder<br>・adder→pc<br>・memory_fetch→command | ・レジスタRs→ra<br>・レジスタRd→rb | ・ALUで(ra+rb)→dr<br>・条件コードを設定 | | ・dr→レジスタRd |
| SUB | | | ・ALUで(ra−rb)→dr<br>・条件コードを設定 | | |
| AND | | | ・ALUで(ra&rb)→dr<br>・条件コードを設定 | | |
| OR | | | ・ALUで(ra\|rb)→dr<br>・条件コードを設定 | | |
| XOR | | | ・ALUで(ra^rb)→dr<br>・条件コードを設定 | | |
| ADDi | | ・レジスタRs→ra<br>・d→ext_d(符号拡張) | ・ALUで(ra+ext_d)→dr<br>・条件コードを設定 | | |
| SUBi | | | ・ALUで(ra−ext_d)→dr<br>・条件コードを設定 | | |
| SLL | | ・レジスタRd→ra<br>・d→ext_d(符号拡張) | ・ALUで(ra << d)→dr<br>・空いたビットに0を挿入<br>・条件コードを設定 | | |
| SLR | | | ・ALUで(ra << d)→dr<br>・空いたビットにシフトアウトされたビット列を挿入<br>・条件コードを設定 | | |
| SRL | | | ・ALUで(ra >> d)→dr<br>・空いたビットに0を挿入<br>・条件コードを設定 | | |
| SRA | | | ・ALUで(ra >> d)→dr<br>・空いたビットにシフトアウトされたビット列を挿入<br>・条件コードを設定 | | |
| IN | | | | ・スイッチからの入力→dr | |
| OUT | | ・レジスタRs→ra<br>・レジスタRd→rb | | ・ra,rbを7SEG LEDに表示 | |
| HALT | | HALT | | | |
| LD | | ・レジスタRa→ra<br>・レジスタRb→rb<br>・Ra→des1<br>・d→ext_d(符号拡張) | ・ALUで(rb+ext_d)→dr | ・アドレスがdrの値のメモリーデータ→MDR | |
| ST | | | | ・raの値をアドレスがdrの値のメモリーに書き込む | |
| LI | | ・d→ext_d(符号拡張) | | | ・enx_d→レジスタRb |
| B | | ・pc→branch_ALU<br>・ext_d→branch_ALU<br>・branch_ALUで(pc+ext_d)→programcounter | | | |
| BE | | ・d→branch_ALU<br>・branch_ALUで条件分岐→programcounter | | | |
| BLT | | | | | |
| BLE | | | | | |
| BNE | | | | | |
| BE' | | ・レジスタRa→branch_ALU<br>・レジスタRb→branch_ALU<br>・d→branch_ALU<br>・branch_ALUで条件分岐→programcounter | | | |
| BLT' | | | | | |
| BLE' | | | | | |
| BNE' | | | | | |

図7　フェーズ・フロー・チャート

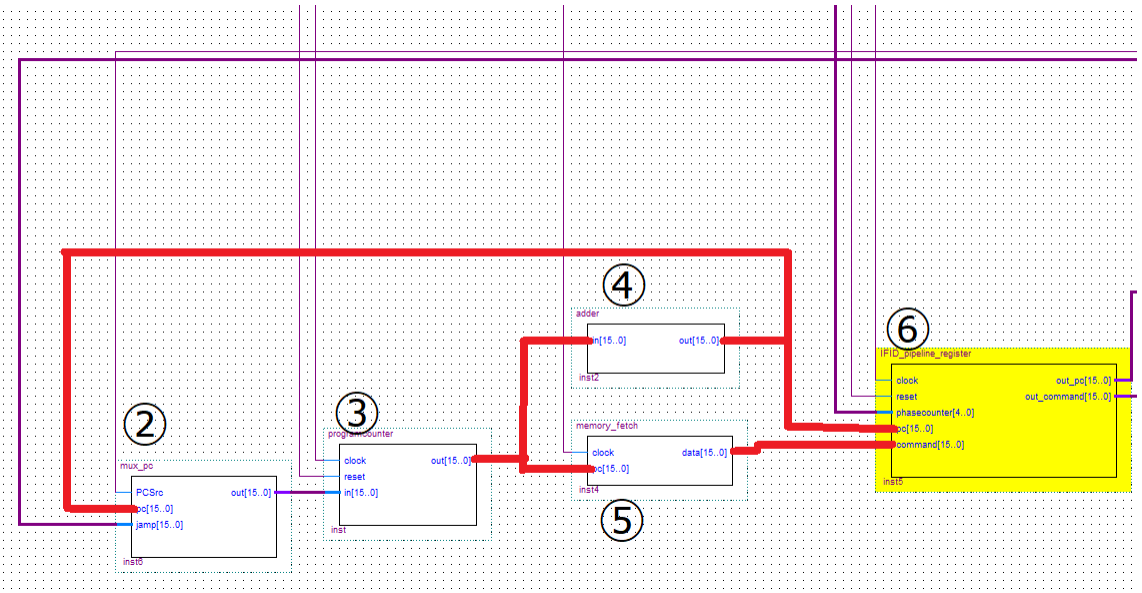


図 8　データ・フロー・チャート (フェーズ 1)

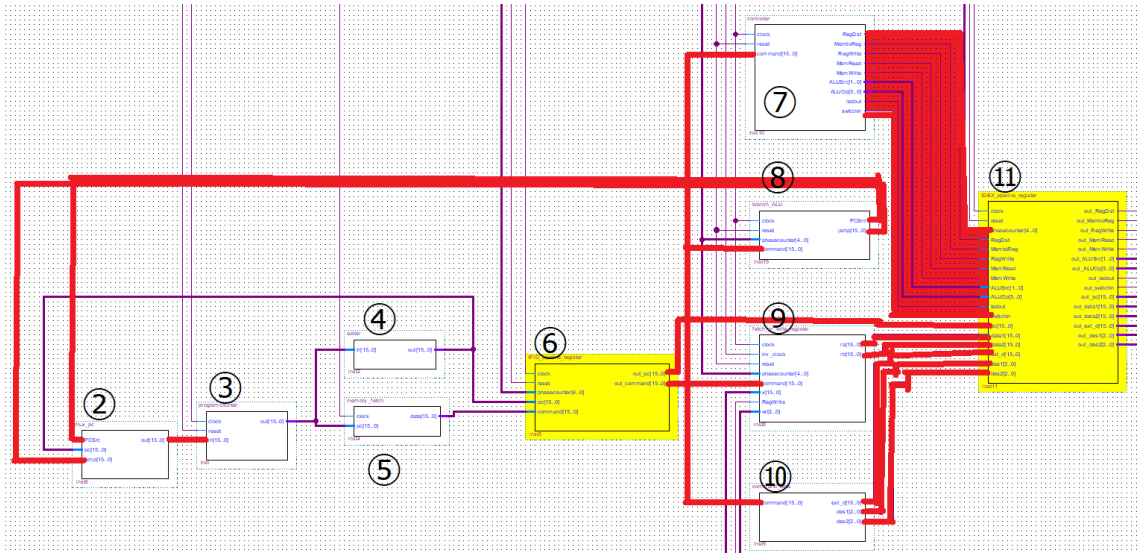


図 9　データ・フロー・チャート (フェーズ 2)

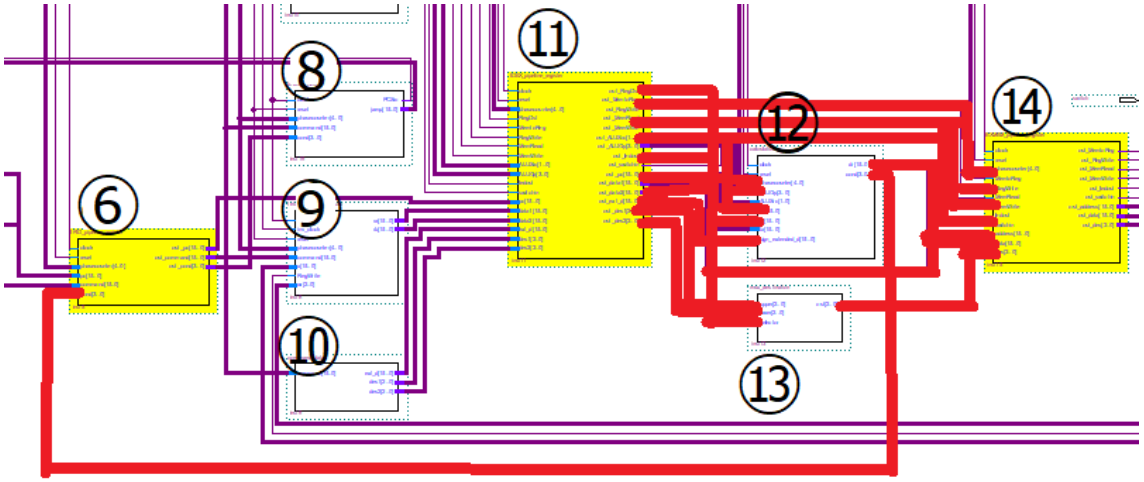


図 10　データ・フロー・チャート (フェーズ 3)

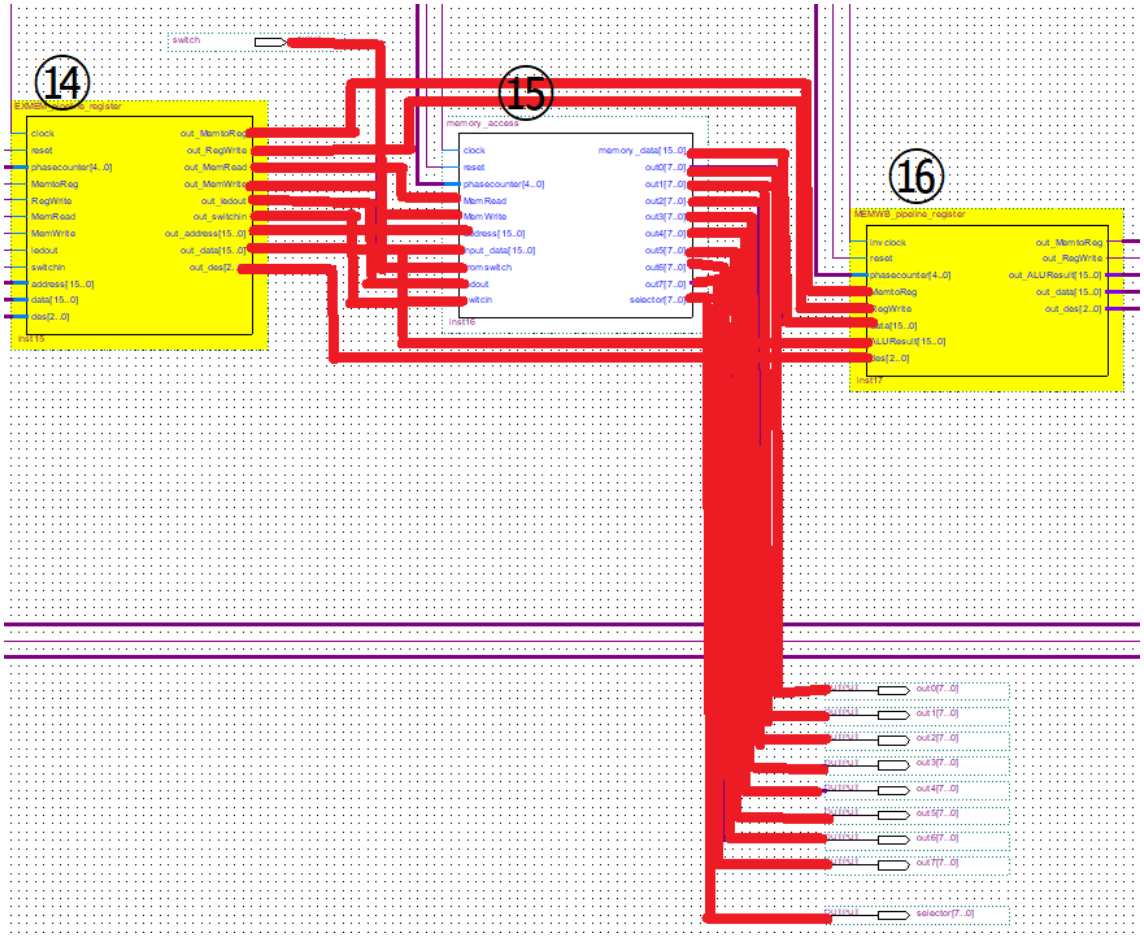


図 11　データ・フロー・チャート (フェーズ 4)

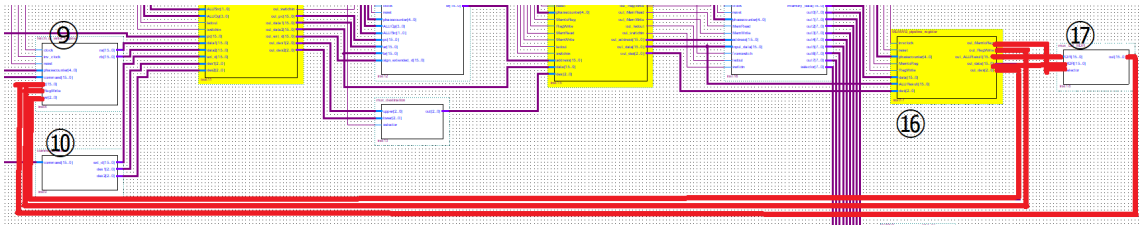


図 12　データ・フロー・チャート (フェーズ 5)