計算機科学実験３A

方式設計仕様書

チーム４：竹田原俊介　Chung Mung Tim

レポート作成者：Chung Mung Tim

2022-05-12

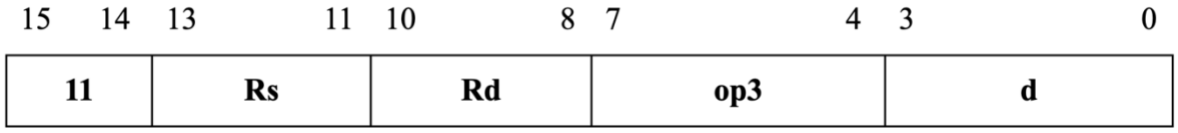
第１章　概要

この実験の目的は、SIMPLE（SIxteen-bit MicroProcessor for Laboratory Experiment）と呼ばれるコンピュータを設計することである．本レポートでは，SIMPLE の命令セットアーキテクチャの詳細と，各命令に対する私たちが設計した CPU 全体としての振る舞いを紹介する．

第２章　命令セット・アーキテクチャ

SIMPLE の命令は全て 1 語 16 ビットの固定長である．SIMPLE は以下 4 種類の命令形式がある．その形式と機能については，2.1 節から 2.4 節で紹介する．

2.1 演算／入出力命令



- $I_{15:14}$ (11)　　操作コード
- $I_{13:11}$ (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（0000～1111）
- $I_{3:0}$ (d)　　シフト桁数

2.1.1　演算命令

SIMPLE の演算命令を表 2.1.1 に示す．演算命令では結果に基づく条件コード S, Z, C, V が設定される．

1. 算術演算
   レジスタ Rd と Rs の加算（ADD）または減算（SUB）の結果を Rd に格納し，条件コードを設定する．
2. 論理演算
   レジスタ Rd と Rs の，ビットごとの論理積（AND），論理和（OR），または排他的論理和（XOR）の結果を Rd に格納し，条件コードを設定する．

3. 比較演算（CMP）
   レジスタ Rd から Rs を減算し，結果に基づく条件コード設定のみを行う．
4. 移動演算（MOV）
   レジスタ Rd に Rs の値を単に格納し，Rd の値に基づき条件コードを設定する．
5. シフト演算
   レジスタ Rd の値を，以下のようにシフトした値を Rd に格納し，条件コードを設定する．
   - SLL（shift left logical）
     左論理シフト．左シフト後，空いた部分に 0 を入れる
   - SLR（shift left rotate）
     左循環シフト．左シフト後，空いた部分にシフトアウトされたビット列を入れる．
   - SRL（shift right logical）
     右論理シフト・右シフト後，空いた部分に 0 を入れる．
   - SRA（shift right arithmetic）
     右算術シフト・右シフト後，空いた部分に符号ビットの値を入れる．

シフト桁数は即値 d（0～15）である．

表 2.1.1：SIMPLE の演算命令

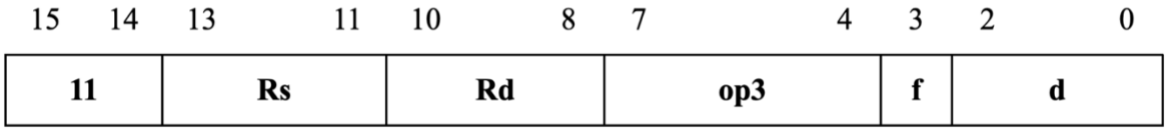
mnemonic		op3	function
ADD	Rd, Rs	0000	r[Rd] = r[Rd] + r[Rs]
SUB	Rd, Rs	0001	r[Rd] = r[Rd] - r[Rs]
AND	Rd, Rs	0010	r[Rd] = r[Rd] & r[Rs]
OR	Rd, Rs	0011	r[Rd] = r[Rd] | r[Rs]
XOR	Rd, Rs	0100	r[Rd] = r[Rd] ^ r[Rs]
CMP	Rd, Rs	0101	r[Rd] - r[Rs]
MOV	Rd, Rs	0110	r[Rd] = r[Rs]
(reserved)		0111	/

SLL	Rd, d	1000	r[Rd] = shift_left_logical (r[Rd], d)
SLR	Rd, d	1001	r[Rd] = shift_left_rotate (r[Rd], d)
SRL	Rd, d	1010	r[Rd] = shift_right_logical (r[Rd], d)
SRA	Rd, d	1011	r[Rd] = shift_right_arithmetic (r[Rd], d)

2.1.2　（拡張）即値オペランドの強化（未実装）

基本アーキテクチャでは，演算命令のオペランドはいずれも汎用レジスタであり，レジスタに即値を加えることは少なくとも２命令を要する．r[Rd] + sign_ext(d)を実行できるようにし，そして r[Rd] + r[Rs]と r[Rd] + sign_ext(d)を切り替えられると望ましい。

そこで演算命令形式を以下のように変更する．元々の d の中の１ビットをフラグ f に用いるようになった.



- f が 0 の場合，r[Rd] + r[Rs]を計算する.
- f が１の場合，r[Rd] + sign_ext(conc(Rs,d))を計算する．ここで，conc(Rs,d)は Rs フィールドの３ビットと d フィルドの３ビットを結合した値を表す.

このようにして得られるオペランドを sw(i, Rs, d)と表記すると，表 2.1.1 は表 2.1.2 に示すものとなる．こうすることで，即値オペランドを使った計算に必要な命令数を減らすことができる.

表 2.1.2　即値オペランドが強化された SIMPLE の演算命令

mnemonic		op3	function
ADD	Rd, Rs	0000	r[Rd] = r[Rd] + sw(i, Rs, d)
SUB	Rd, Rs	0001	r[Rd] = r[Rd] - sw(i, Rs, d)

AND	Rd, Rs	0010	r[Rd] = r[Rd] & sw(i, Rs, d)
OR	Rd, Rs	0011	r[Rd] = r[Rd] | sw(i, Rs, d)
XOR	Rd, Rs	0100	r[Rd] = r[Rd] ^ sw(i, Rs, d)
CMP	Rd, Rs	0101	r[Rd] - sw(i, Rs, d)
MOV	Rd, Rs	0110	r[Rd] = sw(i, Rs, d)
(reserved)		0111	/
SLL	Rd, d	1000	r[Rd] = shift_left_logical (r[Rd], d)
SLR	Rd, d	1001	r[Rd] = shift_left_rotate (r[Rd], d)
SRL	Rd, d	1010	r[Rd] = shift_right_logical (r[Rd], d)
SRA	Rd, d	1011	r[Rd] = shift_right_arithmetic (r[Rd], d)

2.1.3　入出力・制御命令

- IN（input）　　：機器から入力した値をレジスタ Rd に格納する.
- OUT（output）　　：レジスタ Rs の値を機器に出力する・
- HLT（HLT）　　：SIMPLE を停止させる.

表 2.1.3：SIMPLE の入出力・制御命令

mnemonic		op3	function
IN	Rd	1100	r[Rd] = input
OUT	Rs	1101	output = r[Rs]
(reserved)		1110	/
HLT	/	1111	halt()

2.1.4　（拡張）入出力命令の強化（未実装）

基本アーキテクチャでは，FPGA ボードからの入力として IN 命令，ボードへの出力として OUT 命令がある．これらの入出力命令の入出力対象は特定の 16 ビットの入

力／出力に固定されている．しかし，ボードには入力／出力対象ともに 16 ビットを超える数があり，入力／出力ともそれらの中から選択して使える方が望ましい．そこで，入出力先を変更する拡張が考えられる．
特に，Power Medusa EC6S ボードの８桁 7SEG LED は 32 ビット分のデータを表示できる．2.1.2 章で挙げた命令フォマットのうちのフラグ f を用い，８桁の 7SEG LED の上位／下位の切り替えに使う．

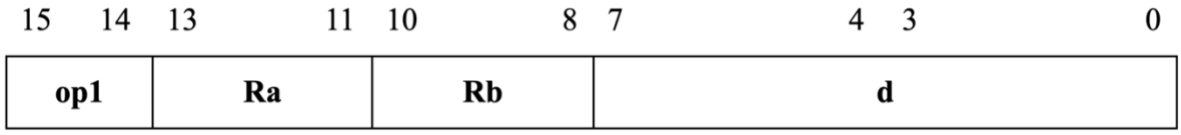
- F が 0 の場合，7SEG LED の４上位を使って出力を表す．
- F が 1 の場合，7SEG LED の４下位を使って出力を表す

また，IN 命令と OUT 命令で未使用のフィールドを使って，入出力先にアドレスを与え，それを 2.1.2 章で定義した sw(i, Rs, d)で指定するように変更するのも考えられる．そこで，表 2.1.3 は表 2.1.4 に示すものとなる．

表 2.1.4　入出力命令が強化された SIMPLE の入出力・制御命令

mnemonic		op3	function
IN	Rd	1100	r[Rd] = input[sw(i, Rs, d)]
OUT	Rs	1101	output[sw(i, Rs, d)] = r[Rs]
(reserved)		1110	/
HLT	/	1111	halt()

2.2　ロード／ストア命令



- $I_{15:14}$ (op1)　操作コード（00/01）
- $I_{13:11}$ (Ra)　ソース・デスティネーションのレジスタ番号
- $I_{10:8}$ (Rb)　ベースレジスタ番号
- $I_{7:0}$ (d)　変位

SIMPLE のロード命令とストア命令の機能を表２に示す．ソース／デスティネーションは，フィールド Ra で指定されたレジスタ Ra である．また実行アドレスはベー

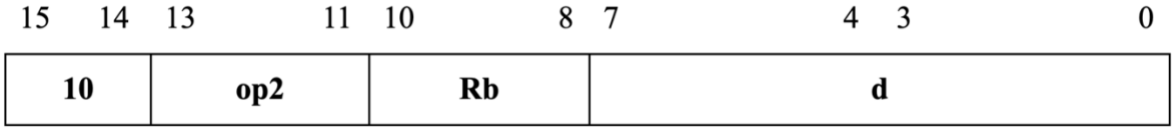
ス・レジスタ・アドレス指定により，フィールド Rb で指定されたレジスタ Rb と，フィールド d を符号拡張した sign_ext(d)を加算して求める．

- LD（load）　：主記憶アドレス(r[Rb] + sign_ext(d))にある 16 ビットの値を Ra にロードする
- ST（store）　：Ra にある 16 ビットの値を主記憶アドレス(r[Rb] + sign_ext(d))のところに書き込む・

表 2.2　SIMPLE のロード・ストア命令

mnemonic		op1	function
LD	Ra, d(Rb)	00	r[Ra] = *(r[Rb] + sign_ext(d))
ST	Ra, d(Rb)	01	*(r[Rb] + sign_ext(d)) = r[Ra]

2.3 即値ロード／無条件分岐命令



- $I_{15:14}$ (10)　操作コード
- $I_{13:11}$ (op2)　操作コード（000～110）
- $I_{10:8}$ (Rb)　ソース・デスティネーション・ベースのレジスタ番号
- $I_{7:0}$ (d)　即値・変位

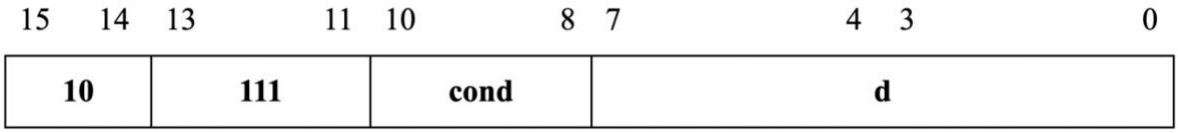
- LI（load immediate）　：即値 sign_ext(d)をレジスタ Rb に格納する
- B（branch）　：d を符号拡張した値 sign_ext(d)を変位として，PC 相対アドレス指定による分岐を行う
- BR（branch register）　：関数呼び出しに使用される・レジスタ r[Rb]の値を分岐アドレスとして分岐する．
- BAL（branch and link）　：関数呼び出しに使用される・次の命令のアドレス PC+1 を復帰アドレスとしてリンクレジスタ r[Rb]に格納する．それと共に，PC 相対アドレス指

定による分岐を行う．なお，分岐アドレスは PC+1 に d を符号拡張した値 sign_ext(d)を加算して求める．

表 2.3　SIMPLE の即値ロード・無条件分岐命令

mnemonic		op1	function
LI	Rb, d	000	r[Rb] = sign_ext(d)
(reserved)		001	/
(reserved)		010	/
(reserved)		011	/
B	d	100	PC = PC + 1 + sign_ext(d)
BR（未実装）	(Rb)	101	PC = r[Rb]
BAL（未実装）	Rb, d	110	r[Rb] = PC + 1 ; PC = PC + 1 + sign_ext(d)
（条件分岐命令）		111	表 2.4 参照

2.4　条件分岐命令形式



1. $I_{15:14}$ (10)　操作コード
2. $I_{13:11}$ (111)　操作コード
3. $I_{10:8}$ (cond)　分岐条件
4. $I_{7:0}$ (d)　変位

SIMPLE の条件分岐命令は以下の表に示すように，フィールド cond で定められる分岐条件が成り立てば PC 相対アドレスによる分岐を行い，成り立たなければ単に次の命令に移行する．各命令の分岐条件は以下の通り：

- BE（branch on equal to）　　：Z が 1
- BLT（branch on less than）　　：　S ^ V が 1
- BLE（branch on less than or equal to）　　：Z または（S ^ V）が 1
- BNE（branch on not equal to）　　：Z が 0
- BGE（branch on great than or equal to）　　：S ^ V が 0
- BGT（branch on greater than）　　：Z || (S ^ V)が 0
- BC（branch on carry）　　：C が 1
- BNC（branch on not carry）　　：C が 0

表 2.4　SIMPLE の条件分岐命令

mnemonic		cond	function
BE	d	000	if (Z) PC = PC + 1 + sign_ext(d)
BLT	d	001	if (S ^ V) PC = PC + 1 + sign_ext(d)
BLE	d	010	if (Z || (S ^ V)) PC = PC + 1 + sign_ext(d)
BNE	d	011	if (!Z) PC = PC + 1 + sign_ext(d)
BGE（未実装）	d	100	if (!(S ^ V)) PC = PC + 1 + sign_ext(d)
BGT（未実装）	d	101	if (!(Z || (S ^ V))) PC = PC + 1 + sign_ext(d)
BC（未実装）	d	110	if (C) PC = PC + 1 + sign_ext(d)
BNC（未実装）	d	110	if (!C) PC = PC + 1 + sign_ext(d)

2.5　ボードから直接入力される命令

reset（リセット信号）

- FPGA ボード上のプッシュスイッチを用いて，スイッチを押すと 1 が，離すと 0 が供給されるようにする．reset が 1 になると，PC 等をクリアし，初期状態に移行する．

exec（起動／停止信号）

- FPGA ボード上のプッシュスイッチを用いて，スイッチを押すと 1 が，離すと 0 が供給されるようにする．
- SIMPLE が停止状態にある時に exec が 0 から 1 に変化すると，SIMPLE は命令の実行を開始する．
- SIMPLE が実行状態にある時に exec が 0 から 1 に変化すると，その時点で実行中の命令が完了してから停止する．

第 3 章　構造と動作

この章では，さまざまな命令を受け取ったときの CPU の振る舞いについて説明する．同じような動作をする命令は，繰り返しを避けるためにまとめてある．第 3 章の説明については，図 3.1 と図 3.2 を参照してください．

図 3.1：SIMPLE のブロック図

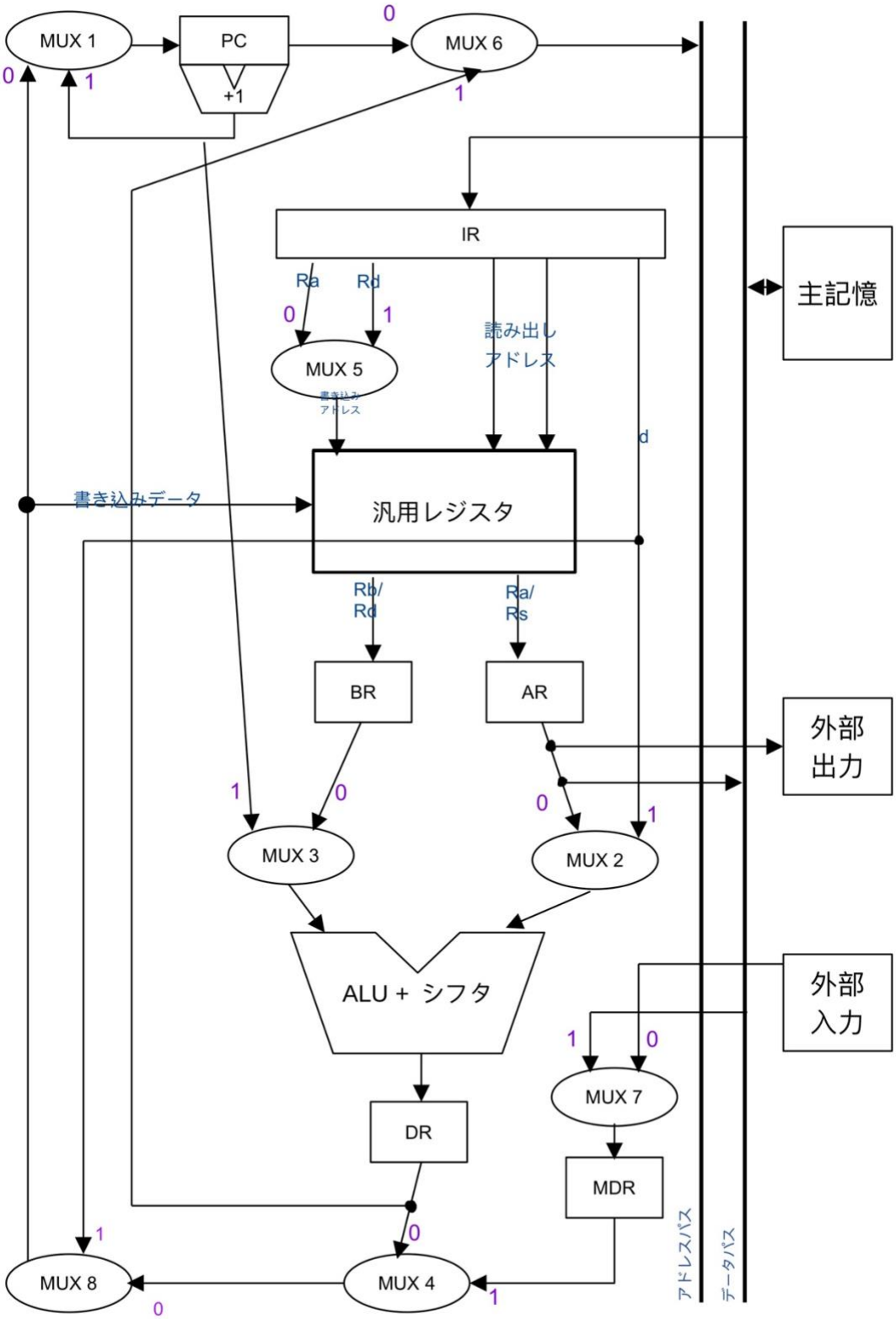
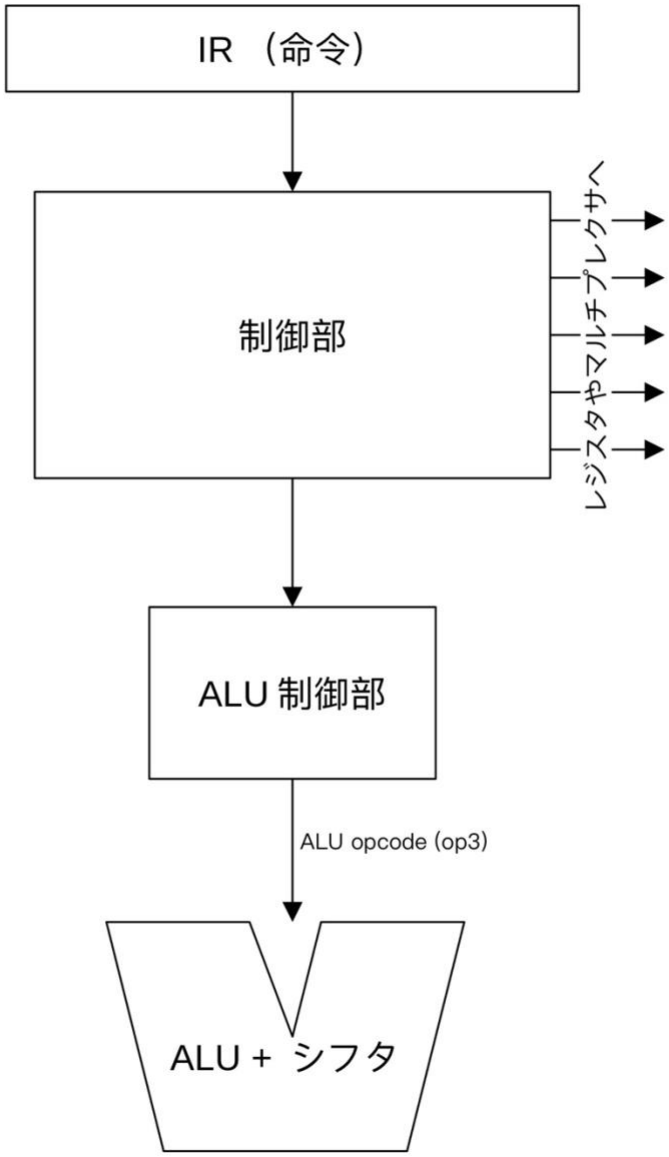


図 3.2　SIMPLE の制御ブロック図



ADD, SUB, AND, OR, XOR, MOV 命令

1. PC（Program Counter）が保持するアドレスの命令を主記憶からフェッチし，IR（Instruction Register）に格納すると共に，PC に 1 を加える．IR にある命令が制御部に届き，制御部が ALU 制御部に必要な演算に対応するコードを送る．

2. IR が保持する命令の，Rs フィールドと Rd フィールドで指定される汎用レジスタのアドレスにある値を読み出し，それぞれレジスタ AR と BR に格納する．
3. マルチプレクサ MUX３と MUX２のセレクタ信号として０が与えられ，AR と BR にある値が ALU の入力になる．ALU により，命令が定める演算を行い，その結果をレジスタ DR（Data Register）に格納する．条件コード S, Z, C, V をセットする．
4. DR にある値を命令の Rd フィールドに指定される汎用レジスタに書き込む．（MUX４のセレクタ信号は０で，MUX５のセレクタ信号は１である．）

CMP 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．IR にある命令が制御部に届き，制御部が ALU 制御部に必要な演算に対応するコードを送る．
2. IR が保持する命令の，Rs フィールドと Rd フィールドで指定される汎用レジスタのアドレスにある値を読み出し，それぞれレジスタ AR と BR に格納する．
3. マルチプレクサ MUX３と MUX２のセレクタ信号として０が与えられ，AR と BR にある値が ALU の入力になる．ALU により，命令が定める演算を行うが，結果を出力しない．条件コード S, Z, C, V をセットする．

SLL, SLR, SRL, SRA 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．IR にある命令が制御部に届き，制御部が ALU 制御部に必要な演算に対応するコードを送る．
2. IR が保持する命令の，Rd フィールドで指定される汎用レジスタのアドレスにある値を読み出し，レジスタ BR に格納する．
3. MUX３のセレクタ信号は０である．MUX２のセレクタ信号は１で，BR にある値と符号拡張した即値 d が ALU の入力になる．ALU により，命令が定める演算を行い，その結果をレジスタ DR に格納する．条件コード S, Z, C, V をセットする．
4. DR にある値を命令の Rd フィールドに指定される汎用レジスタに書き込む．（MUX４のセレクタ信号は０で，MUX５のセレクタ信号は１である．）

IN 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．IR にある命令が制御部に届き，制御部が制御信号を発信する．
2. 入力スイッチの値が MDR に格納される．（ここで MUX７のセレクタは０になる．）
3. MDR にある値を命令の Rd フィールドに指定される汎用レジスタに書き込む．（MUX４のセレクタ信号は１で，MUX５のセレクタ信号は１である．）

OUT 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．IR にある命令が制御部に届き，制御部が制御信号を発信する．
2. フィールド Rs にある値が AR に格納される．
3. AR の値が７SEG LED などへ表示される．（ここで MUX７のセレクタは０になる．）

HLT 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．
2. IR にある命令が制御部に届き，制御部が全ての部品を停止させる．

LD 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．IR にある命令が制御部に届き，制御部が ALU 制御部に必要な演算に対応するコード（加算 ADD）を送る．
2. IR が保持する命令の，Rb フィールドで指定される汎用レジスタのアドレスにある値を読み出し，レジスタ BR に格納する．
3. MUX３のセレクタ信号は０である．MUX２のセレクタ信号は１で，BR にある値と符号拡張した即値 d が ALU の入力になる．ALU により，加算を行い，その結果をレジスタ DR に格納する．

4. DR にある値をアドレスとして主記憶をアクセスする．（ここで MUX６のセレクタは１になる．）読み出した値をレジスタ MDR（Memory Data Register）に格納する．（ここで MUX７のセレクタは１になる．）
5. MUX４のセレクタが１になり，MDR にある値を命令の Ra フィールドに指定される汎用レジスタに書き込む．（MUX５のセレクタは０になる．）

ST 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．IR にある命令が制御部に届き，制御部が ALU 制御部に必要な演算に対応するコード（加算 ADD）を送る．
2. IR が保持する命令の，Rb フィールドで指定される汎用レジスタのアドレスにある値を読み出し，レジスタ BR に格納する．Ra フィールドで指定される汎用レジスタのアドレスにある値を読み出し，レジスタ AR に格納する．
3. MUX３のセレクタ信号は０である．MUX２のセレクタ信号は１で，BR にある値と符号拡張した即値 d が ALU の入力になる．ALU により，加算を行い，その結果をレジスタ DR に格納する．
4. DR にある値をアドレスとして主記憶をアクセスする．（ここで MUX６のセレクタは１になる．）AR が保持する値をそのアドレスに書き込む．

LI 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．
2. MUX8 のセレクタ信号が１になり，書き込みデータとして符号拡張した d を汎用レジスタに書き込む．書き込みアドレスを決めるために，MUX５のセレクタ信号が１になる．

B, BE, BLT, BLE, BNE 命令

1. PC が保持するアドレスの命令を主記憶からフェッチし，IR に格納すると共に，PC に１を加える．IR にある命令が制御部に届き，制御部が ALU 制御部に必要な演算に対応するコード（加算 ADD）を送る．
5. MUX3 のセレクタは１で，PC＋1 の値が ALU の一つの入力になる．MUX２のセレクタは１で，符号拡張した d の値が ALU のもう一つの入力になる．ALU が加算を行い，その結果をレジスタ DR に格納する．

6. DR にある値を分岐先アドレスとして PC に書き込む．（ここで MUX１は０，MUX４は０．）

reset（リセット信号）

1. 制御部へ１ビットの reset 信号が届く．
2. 制御部で SIMPLE を初期状態にさせる．

exec（起動／停止信号）

1. 制御部へ１ビットの exec 信号が届く．
2. 制御部でフラグを使って，その時点で実行中の命令が完了してから SIMPLE を停止する．

参考文献：
富田真治，中島浩『コンピュータハードウェア（情報系教科書シリーズ第６巻）』