2022 年度 3 回生前期学生実験 HW

team02 最終成果物のユーザーマニュアル

最終成果物のユーザーマニュアルの作成者: 植田健斗
グループメンバー：
伊藤舜一郎 (学籍番号:1029-32-7548)
植田健斗　(学籍番号:1029-32-6498)

提出期限：6 月 9 日 13 時　提出日: 2022 年 6 月 3 日

1　概要

授業資料の仕様を満たすプロセッサをもとにして、拡張機能の追加をしたプロセッサを設計した。このプロセッサは、高速化を目標にして、拡張機能を追加した。このプロセッサの実装で特に力を入れた箇所は以下の箇所である。

- 5 段のパイプライン処理
- 即値命令 ADDI の実装

2　性能と特長

2.1　動作周波数

このプロセッサは入力 CLK に 40MHz のクロック信号を与えることで、FPGA 上で内部的に 50MHz で動作する。

2.2　CPI

5 命令が並列処理されているときの CPI は以下のようになっている。

- ロード命令 (LD) は 1 だが、ロード命令で書きこむレジスタの内容を次の命令で読み出す場合 (詳しくはハザード検出ユニットを参照のこと) は 2
- 入力命令 (IN) は 1 以上で、exec ボタンが押されるまでのクロック数
- 条件分岐・無条件分岐命令はいずれも、分岐が不成立の場合は 1、成立する場合は 2
- その他の命令は、すべて 1

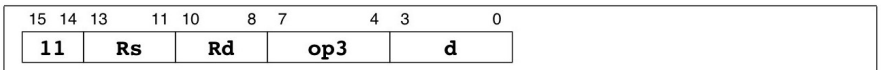
2.3　特徴

このプロセッサの特徴を以下に記す。

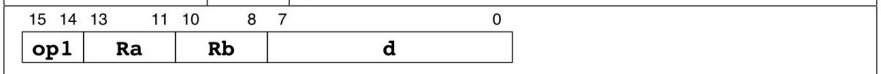
- 5 段のパイプライン処理
- 即値命令 ADDI

3　命令セット・アーキテクチャ

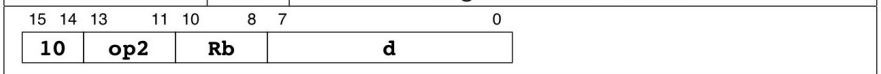
命令セットアーキテクチャは以下の図 1 のようになる。



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)
IN Rd,d	1100	r[Rd] = input
OUT Rs	1101	output = r[Rs]
NOP	1110	
HLT	1111	halt()



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]



mnemonic	op2	function
LI Rb,d	000	r[Rb] = sign_ext(d)
ADDI Rb,d	001	r[Rb] = r[Rb] + sign_ext(d)
(reserved)	010	
(reserved)	011	
B d	100	PC = PC + 1 + sign_ext(d)
(reserved)	101	
(reserved)	110	
(条件分岐命令)	111	

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)
(reserved)	100	
(reserved)	101	
(reserved)	110	
(reserved)	111	

図 1: 命令セット・アーキテクチャ

以下命令形式と各命令について、詳しく説明する。

3.1 命令形式

命令は全て 1 語 (16 ビット) の固定長であり，以下に示すように 4 種類の命令形式がある．各命令形式とフィールドの意味は以下の通りである．

1. 演算／入出力命令形式

   $I_{15:14}$ **(op1)** 操作コード (11) (operation code, opcode)

   $I_{13:11}$ **(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. ロード／ストア命令形式

   $I_{15:14}$ **(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 (displacement)

3. 即値ロード／無条件分岐命令形式

   $I_{15:14}$ **(op1)** 操作コード (10)

   $I_{13:11}$ **(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)** 即値または変位

4. 条件分岐命令形式

   $I_{15:14}$ **(op1)** 操作コード (10)

   $I_{13:11}$ **(op2)** 操作コード (111)

   $I_{10:8}$ **(cond)** 分岐条件

   $I_{7:0}$ **(d)** 変位

3.2 算術演算 (授業資料をそのまま掲載)

レジスタ Rd と Rs の加算 (ADD: add) または減算 (SUB: subtract) の結果を Rd に格納し，条件コードを設定する．条件コード C には最上位ビットからの桁上げが設定される．

3.3 論理演算 (授業資料をそのまま掲載)

レジスタ Rd と Rs の，ビットごとの論理積 (AND: and), 論理和 (OR: or)，または排他的論理和 (XOR: exclusive-or) の結果を Rd に格納し，条件コードを設定する．但し条件コード C は演算結果に関わらず 0 となる．

3.4 比較演算 (CMP: compare)(授業資料をそのまま掲載)

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．条件コード C には最上位ビットからの桁上げが設定される．

3.5 移動演算 (MOV: move)(授業資料をそのまま掲載)

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．但し条件コード C は Rs の値に関わらず 0 となる．

3.6 シフト演算 (授業資料をそのまま掲載)

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．また条件コード C には，シフト桁数が 0 の時または SLR では 0 が，それ以外では最後にシフトアウトされたビットの値が設定される．条件コード V は常に 0 が設定される．

3.7 ロード／ストア命令 (授業資料をそのまま掲載)

ロード命令 (LD: load) とストア命令 (ST: store) の機能を説明する．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．また実効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．ロード命令の SZCV フラグは未定義である。また、ストア命令の SZCV フラグについてはメモリに格納するデータが正のとき SZCV=0000、負のとき SZCV=1000、0 のとき SZCV=0100 となるように SZCV フラグを設定される。

3.8 即値ロード／無条件分岐命令 (授業資料をそのまま掲載)

SIMPLE の即値ロード命令 (LI: load immediate) と，無条件分岐命令 (B: branch) の機能を以下に示す．即値ロード命令の SZCV フラグについては即値 d を符号拡張した値が正のとき SZCV=0000、負のとき SZCV=1000、0 のとき SZCV=0100 となるように SZCV フラグを設定される。

LI 即値 sign ext(d) をレジスタ Rb に格納する．

B d を符号拡張した値を変位として，PC 相対アドレス指定による分岐を行う．

3.9 条件分岐命令 (授業資料をそのまま掲載)

SIMPLE の条件分岐命令は, 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ない，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) 条件コード Z が 1

BLT (branch on less-than) 条件コード S と V の XOR(S $\hat{}$ V) が 1

BLE (branch on less-than or equal-to) Z または (S $\hat{}$ V) が 1

BNE (branch on not-equal-to) 条件コード Z が 0

3.10 IN 命令

IN 命令が呼ばれると実行を一時的に中断し、外部入力で入力された値をうけとる。実行の中断中に 16 個のディップスイッチを変更し、16bit の値を設定し、exec ボタンを押すことで exec ボタンが押された時のディップスイッチを用いて表現された値を入力として受け取り、実行を再開するようにした。その値を IN 命令中で指定した番号のレジスタに格納する。IN 命令の SZCV フラグは未定義である。

3.11 OUT 命令

OUT 命令が呼ばれると、フェーズが p3 の時に外部出力に Rs フィールドで指定したレジスタの中身が渡される。外部出力では 7SEG-LED を 4 つ用いて 16 進数表示で 16bit のデータを表示する。また、過去 16 回の OUT 命令で出力されたデータを保持し、7SEG-LED に表示する。OUT 命令のときの SZCV フラグについては OUT 命令により出力されるデータが、正のとき SZCV=0000、負のとき SZCV=1000、0 のとき SZCV=0100 となるように SZCV フラグを設定される。

3.12 HLT 命令

命令の実行を中断するための命令である。内部的には exec ボタンが押されたのと同じ動作をする。HLT 命令が呼ばれた後に、exec ボタンを押すとメモリ上で HLT 命令の次の命令から命令の実行を再開する。HLT 命令が呼ばれた後、HLT 命令の次の番地に命令を書いていない場合に、exec ボタンを押すと未定義動作を引き起こす可能性があるので、注意が必要である。

3.13 NOP 命令 (non operation)

レジスタ・メモリ書き込み、外部入出力などの動作を、何も行わない命令である。この命令はハードウェアで、リセットの動作をする際や、パイプラインストールの際に何もしない命令が必要であるためにある。機器の初期化の際には、reset ボタンが押されると Instruction Register の値は 2'b 1100000011100000 となり、NOP 命令の値に設定される。なお“(reserved)”と記された命令は何の動作もせず，単に次の命令に移行する.

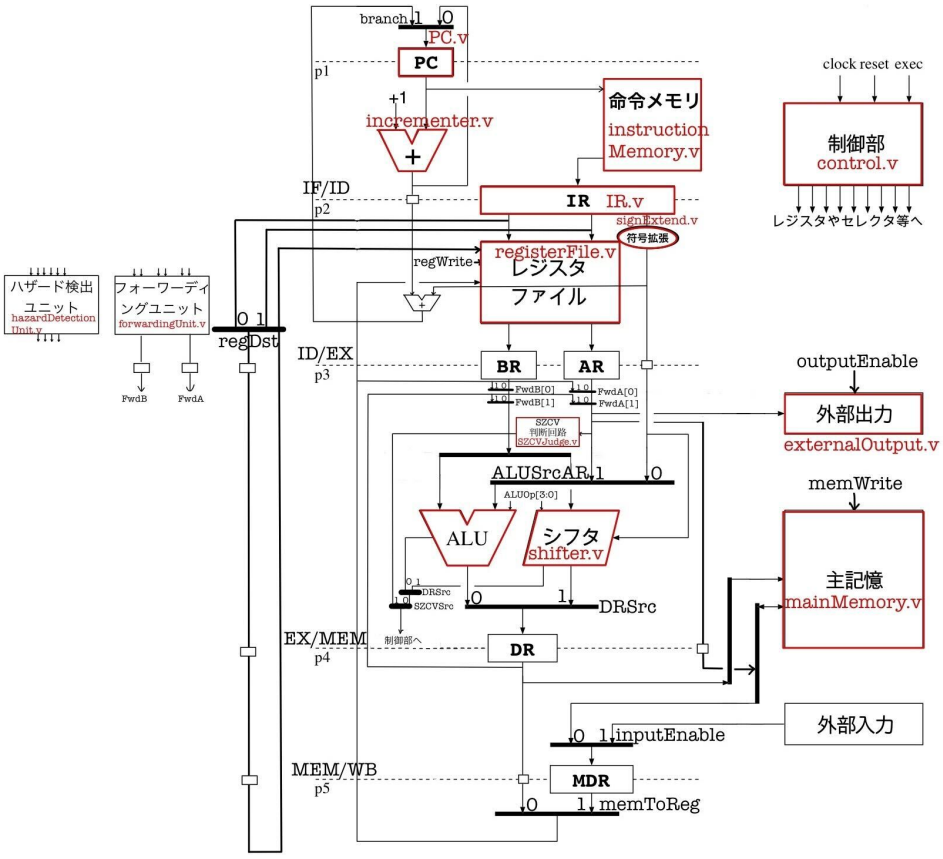
3.14 ADDI 命令 (add immediate)

Rb フィールドで指定したレジスタの中身を読み出し、命令の下 8bit で指定した符号付きの値 d を足し合わせる。足し合わせた値を Rb フィールドで指定したレジスタに格納する。Rb フィールドの値により、条件コードが設定される。条件コード C には最上位ビットからの桁上げが設定される。

4 構造と動作

4.1 構造

現状のプロセッサの構造は以下の図 2 のようになる。



各レジスタ:register16.v,register4.v,register3.v
各マルチプレクサ:multiplexer16.v,multiplexer4.v,multiplexer3.v
全体:processor.bdf

図 2: プロセッサの構造 (最終段階)

パイプライン化に伴い必要な構造を以下で示す。

- メモリを命令メモリとデータメモリに分け、構造ハザードを防いでいる。(ハーバードアーキテクチャ)
- ALU とは別にアドレス計算用の加算器を用意し、アドレス計算と演算の際の ALU での構造ハザードを防いでいる。
- 各フェーズの間でデータを送る際に、配線ではなくレジスタを用意し、フェーズ間での制御信号やデータの受け渡しを行う。
- MEM フェーズもしくは WB フェーズから、ALU のある EX フェーズに、データのフォーワーディングを行うフォーワーディングユニットを追加した。

- ID フェーズでレジスタを読み込みと、WB フェーズでレジスタ書き込みを同じクロック周期内で同じレジスタに対して行うときに、レジスタに書きこもうとするデータをレジスタから読み出されるように、レジスタファイルの設計をした。
- ロード命令 (LD) や入力命令 (IN) でレジスタに値を格納し、その次の演算命令でその値を読み出そうとすることを考える。レジスタから読み出しをする際にレジスタに書きこむ値がまだわからないため、読み出せない問題 (パイプラインストール) が発生する。パイプラインストールが起こったときに、ロード命令 (LD) や入力命令 (IN) の次に NOP 命令を入れて、データハザードを防ぐ、ハザード検出ユニットを追加する。
- 分岐が成立した際に、不成立分岐予測により読み込んだ次の命令を、フラッシュ(NOP 命令に置き換える) する機能を、ハザード検出ユニットに追加する。
- メモリ読み出しを行っているときに 1 になる信号である memRead 信号を追加し、パイプラインストールの判定に用いた。

これらの構造がパイプライン化のために必要な構造である。パイプライン化により、プロセッサ全体の外部仕様は変えずに、処理の高速化を行う。

4.2 動作

このプロセッサは 5 フェーズに処理をわけ、それぞれを p1(IF),p2(ID),p3(EX),p4(MEM),p5(WB) というフェーズとする。p1 から p5 のそれぞれを並列実行するパイプライン処理を行うようにした。それぞれのフェーズ間、つまり、図 2 の破線上にフェーズ間のデータの受け渡しを行うパイプラインレジスタを配置する。クロックの立ち上がりごとにパイプラインレジスタの値は更新され、それぞれのフェーズにおかれた組み合わせ回路の入力となる。その組み合わせ回路からの出力が、次のフェーズの間のパイプラインレジスタに入力され、次のクロックの立ち上がりにパイプラインレジスタの値が更新される。これを繰り返すことで、それぞれの命令は原則 1 クロックあたり 1 フェーズずつ進む。各命令は、1 つのフェーズ内の回路のみを使うため、命令の処理をフェーズごとに独立に行うことができ、それにより、5 つの命令を並列実行している。

このプロセッサに用いられるそれぞれのレジスタはイネーブル信号を入力として受け取る。レジスタのイネーブル信号とは、クロックの立ち上がりでかつ、レジスタのイネーブル信号が 1 であるときのみ、レジスタの中身が書き変わり、それ以外はレジスタの中身は変わらないような信号である。

このプロセッサは制御部内に SystemRunning という 1bit のレジスタをもっており、そのレジスタの値が 0 のときに SystemRunning レジスタ以外のすべてのレジスタのイネーブル信号が 0 になるようになっている。

exec ボタンを押されると、この SystemRunning の反転が行われる。これにより exec ボタンを押すたびに実行停止・再開ができる。

reset ボタンを押すと、以下のことがすべて行われる。

- IR に NOP 命令を表す命令コード 2'b 1100000011100000 が格納される。
- PC に 0 が格納される。
- 外部出力の過去の出力履歴がクリアされる。

- レジスタファイルの内部のレジスタに 0 が格納される。
- SystemRunning に 0 が格納される。
- 制御信号を格納するレジスタで、レジスタ・メモリ書き込み、入出力、メモリ読み出しのための制御信号 (つまり制御信号 outputEnable,memRead,inputEnable,memWrite,branch,regWrite) を 0 にし、書き込み、入出力、ハザード検出が行われないようにする。

これにより reset ボタンを押すことで機器を初期状態にし、実行待機状態にすることができる。この状態で exec ボタンを押すとメモリの 0 番地から順に命令が実行される。

FPGA にプログラムをダウンロードした後に exec ボタンを押さなければ、命令が実行されないことに注意が必要である。

また、HLT 命令は、exec ボタンが押されたのと同じ動作をするので、HLT 命令の後ろに命令を記述しておいた場合、HLT 命令で実行を中断し、そこから、exec ボタンを押すと実行を再開できる。これを用いると、OUT 命令の次に HLT 命令を実行することで、OUT 命令のたびに一時停止するので、出力を確認してから exec ボタンを押し実行を再開することができる。

ただ、HLT 命令の次の番地に命令を記述していない場合に、HLT 命令で止まった後、exec ボタンが押されると未定義動作を引き起こすので注意が必要である。

4.3 ピン配置

ピン配置は以下の図 3 のようになる。OUT 命令、IN 命令での入出力は、それぞれ 7SEG-LED とディップスイッチをもちいてやっている。

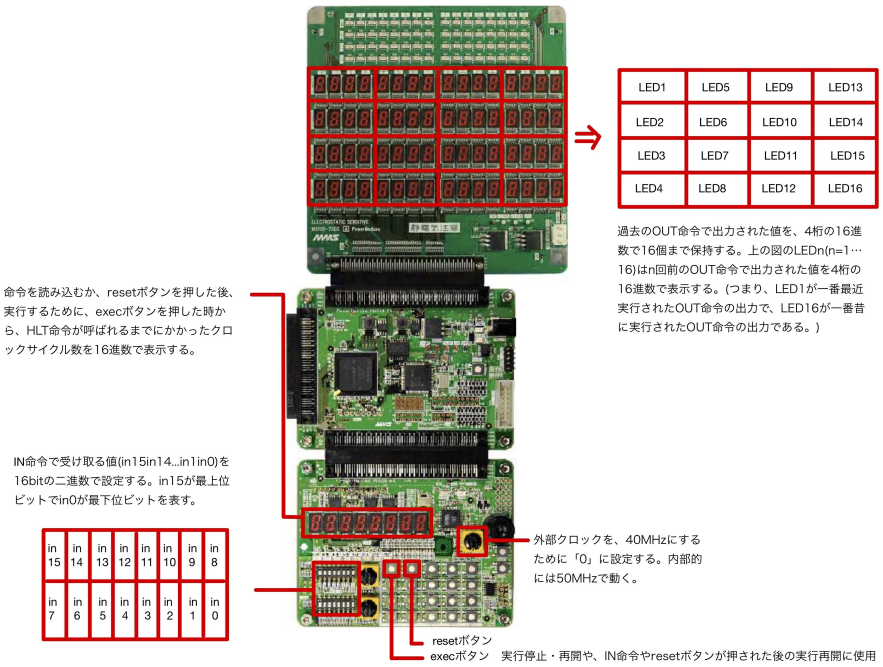


図 3: ピン配置