データパスの構築 I
～シングルサイクル・データパス～

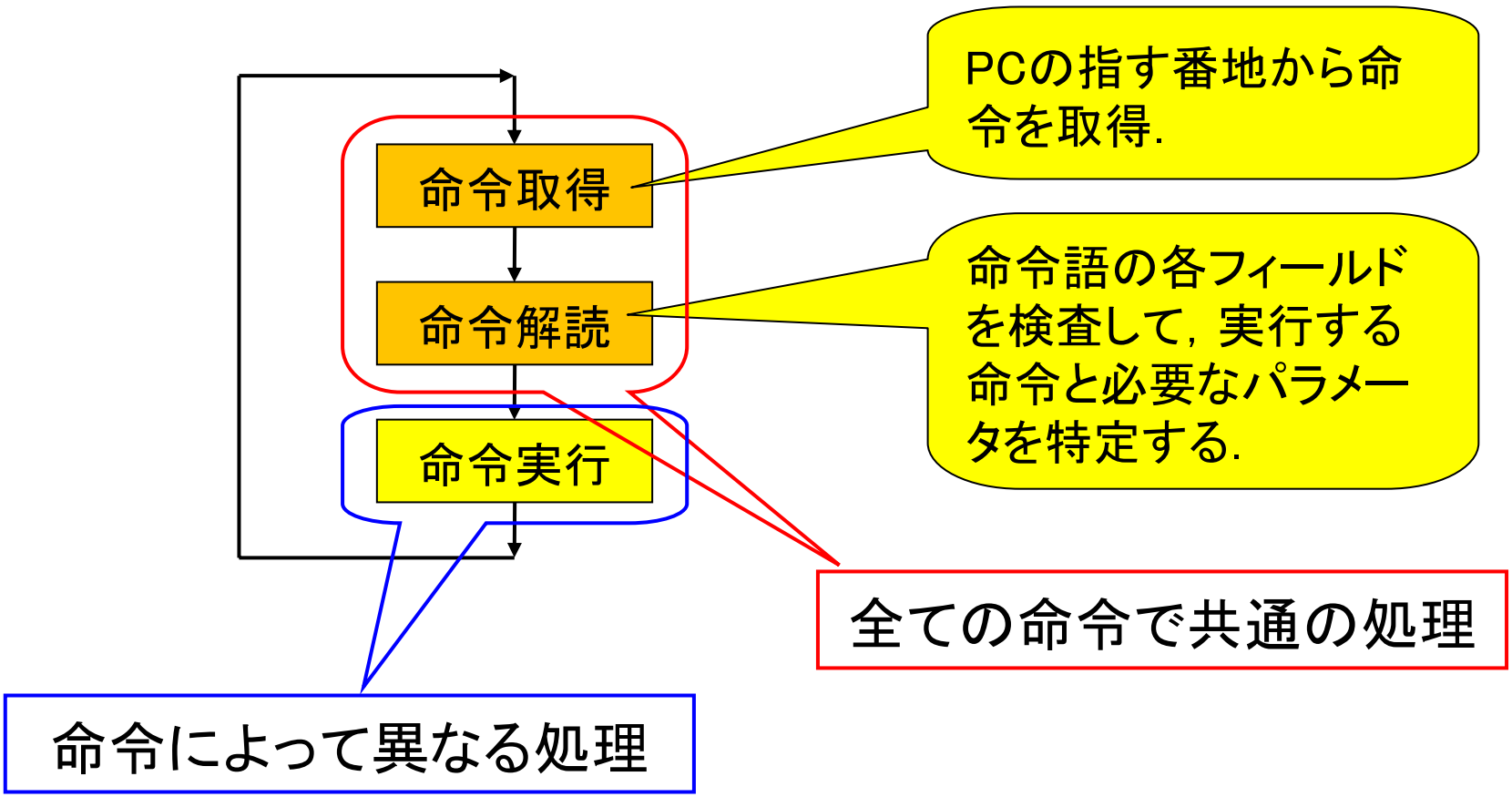
（教科書5.1～5.3節）

プロセッサの構成：データパスと制御

- MIPS命令セットを実現するプロセッサを設計する！
- ただし，以下の命令セットに限定
  - 算術論理演算命令：add，sub，and，or，slt
  - メモリ参照命令：lw，sw
  - 分岐命令：beq

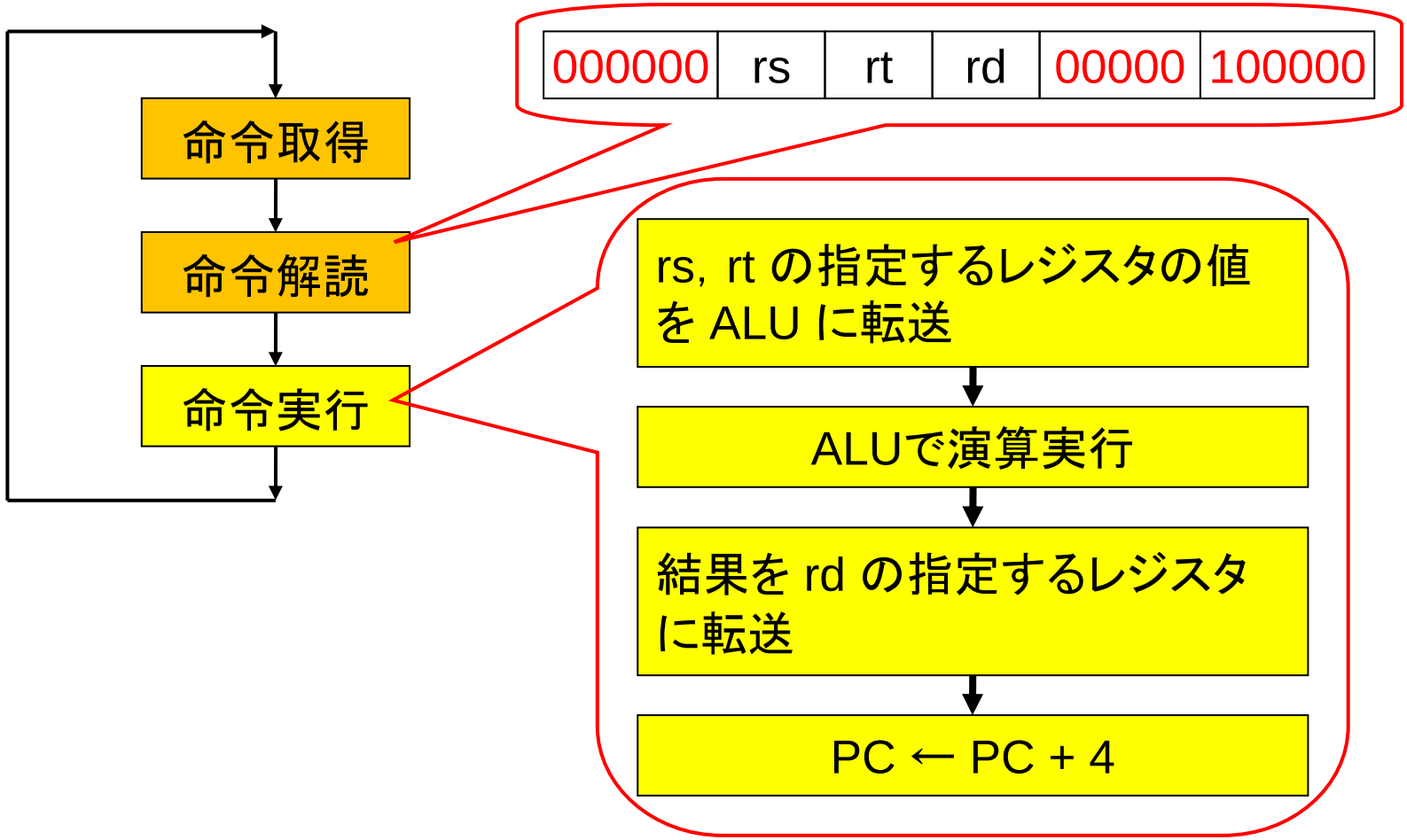
命令区分	命令の内容	例	意味
算術論理演算	加算	add $s1, $s2, $s3	$s1 = $s2 + $s3
	減算	sub $s1, $s2, $s3	$s1 = $s2 − $s3
	論理積	and $s1, $s2, $s3	$s1 = $s2 and $s3 (ビット毎の論理積)
	論理和	or $s1, $s2, $s3	$s1 = $s2 or $s3 (ビット毎の論理和)
	比較(セット)	slt $s1, $s2, $s3	もし、$s2<$s3なら$s1=1, $s2>=$s3なら$s1=0
データ転送	ロードワード	lw $s1, 100($s2)	$s1に，メモリの[$s2+100]番地のワードデータを読込み
	ストアワード	sw $s1, 100($s2)	メモリの[$s2+100]番地に，$s1のワードデータを書込み
分岐	条件付	beq $s1, $s2, L	もし、$s1==$s2ならラベルLへ分岐

命令実行の基本動作

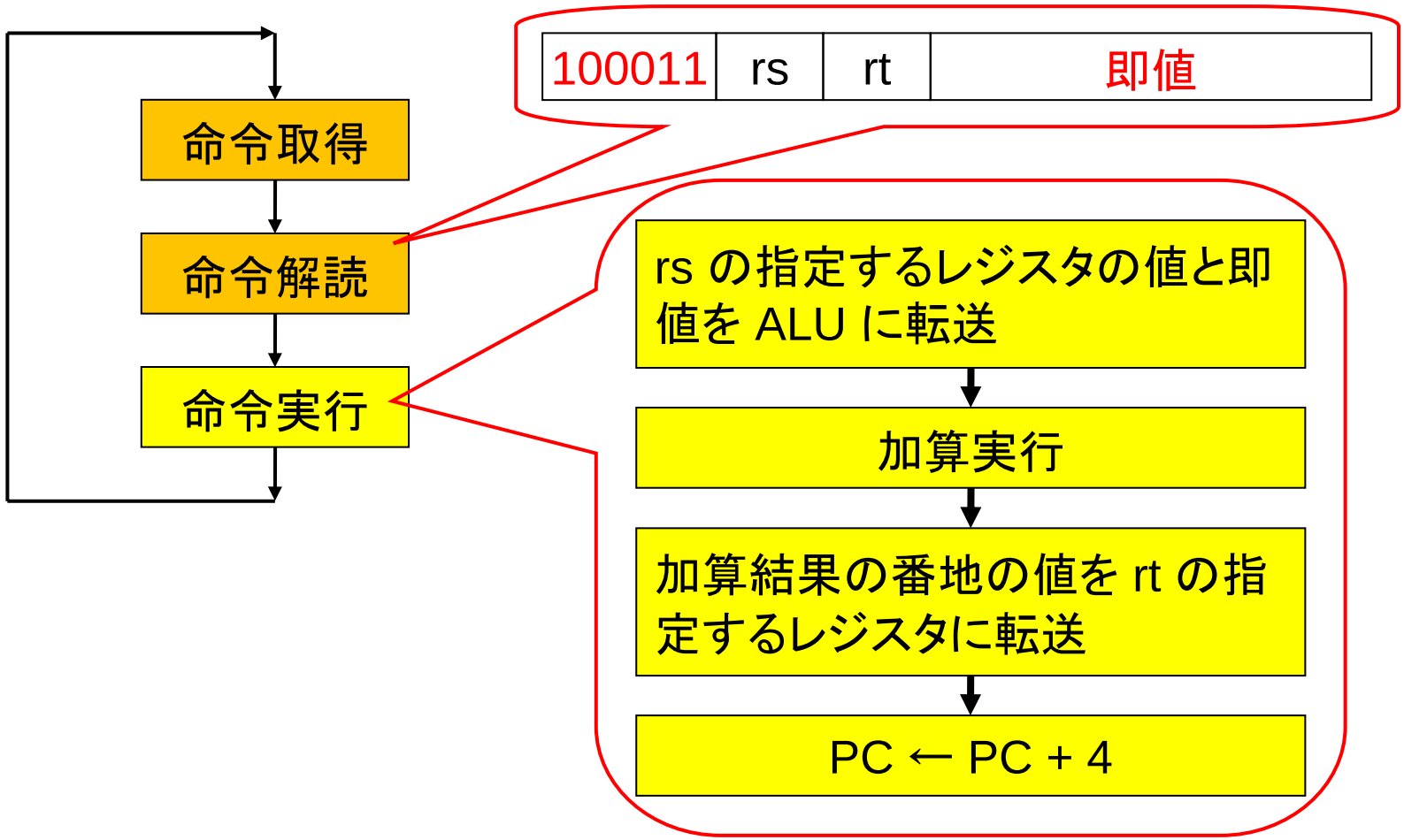

命令取得
命令解読
命令実行
PCの指す番地から命令を取得.
命令語の各フィールドを検査して，実行する命令と必要なパラメータを特定する.
全ての命令で共通の処理
命令によって異なる処理

命令実行の基本動作（add命令）

（sub, and, or, slt も同様）



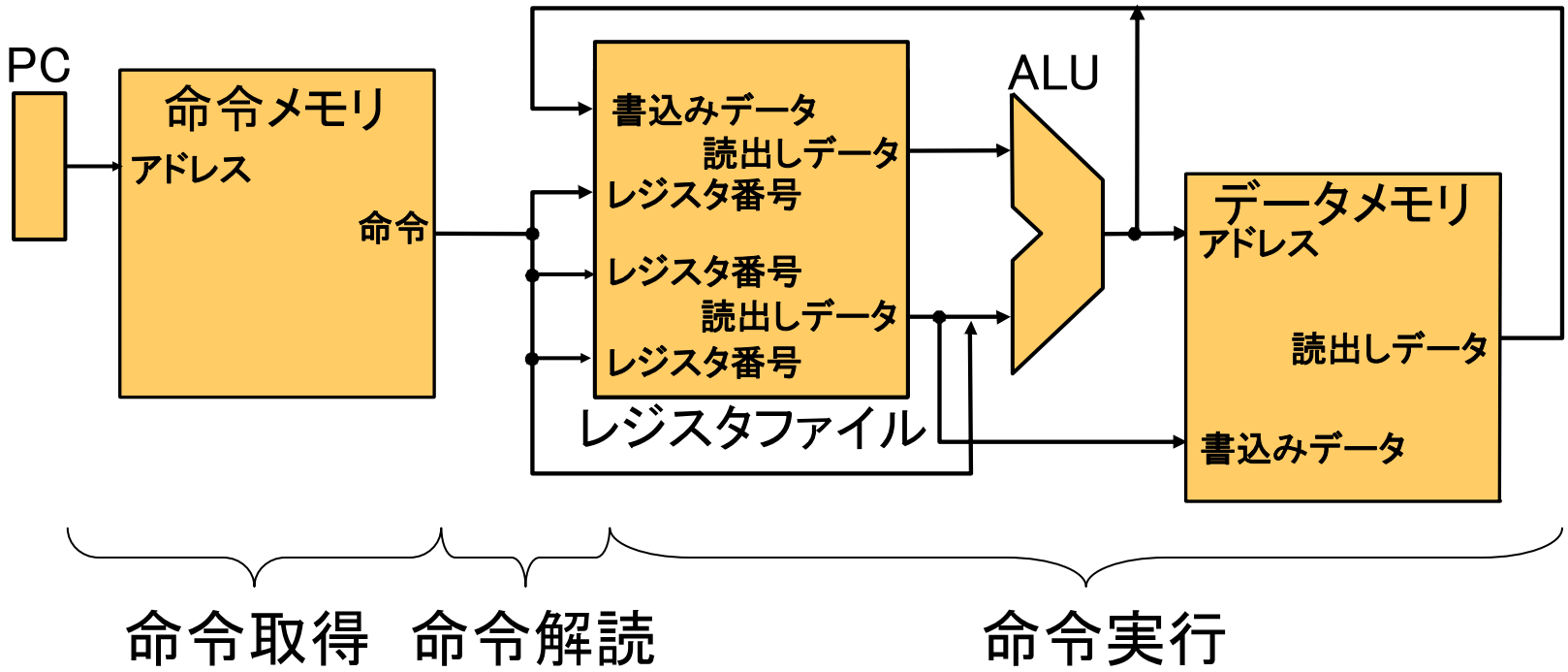
命令実行の基本動作(lw命令)



各命令実行で必要な処理

命令の種類	命令	命令実行で必要となる処理				
		Step1	Step2	Step3	Step4	Step5
算術論理演算	add/sub/ and/or/slt	•レジスタファイルよりrsとrtを読出し	•ALUでの演算	•演算結果をrdへ書き込み	•PCの更新（＋4）	
データ転送	lw	•レジスタファイルよりrsを読出し •命令より即値を選択	•ALUでのアドレス計算（＋）	•メモリから値を読出し（ロード）	•ロード値をrtへ書込み	•PCの更新（＋4）
	sw	•レジスタファイルよりrsとrtを読出し •命令より即値を選択	•ALUでのアドレス計算（＋）	•メモリへrtの値を書込み	•PCの更新（＋4）	
分岐	beq	•レジスタファイルよりrsとrtを読出し •命令より分岐先オフセットを選択	•ALUでの一致判定（－）	•PCの更新（分岐先の値）		

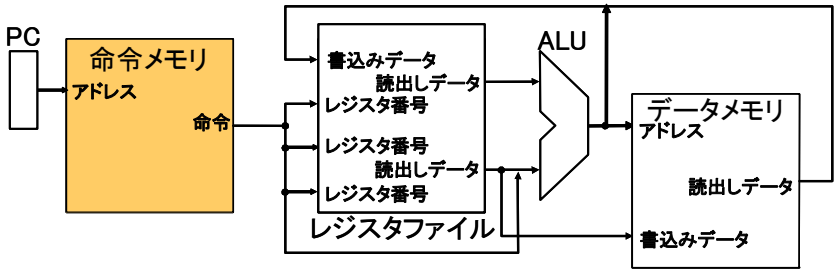
データパスの概観

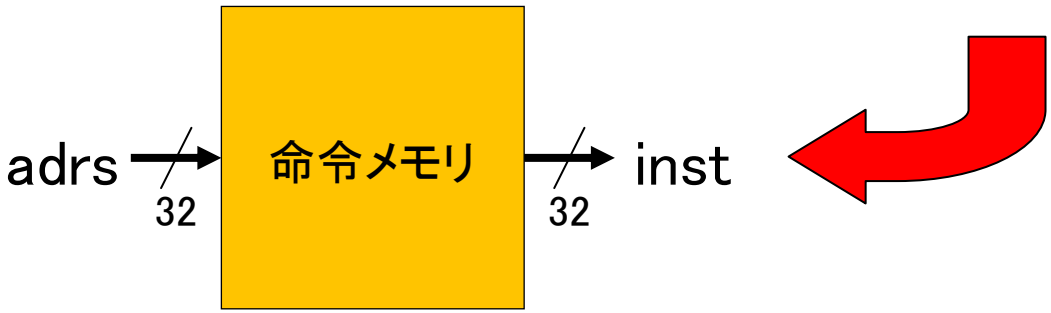

PC
命令メモリ
アドレス
命令
書込みデータ
読出しデータ
レジスタ番号
レジスタ番号
読出しデータ
レジスタ番号
レジスタファイル
ALU
データメモリ
アドレス
読出しデータ
書込みデータ
命令取得
命令解読
命令実行

部品の準備(命令メモリ)

命令メモリ:実行対象となる命令(プログラム)を格納

信号の意味	入出力	信号名	ビット幅
読み出す命令の番地	入力	adrs	32
読み出された命令	出力	inst	32

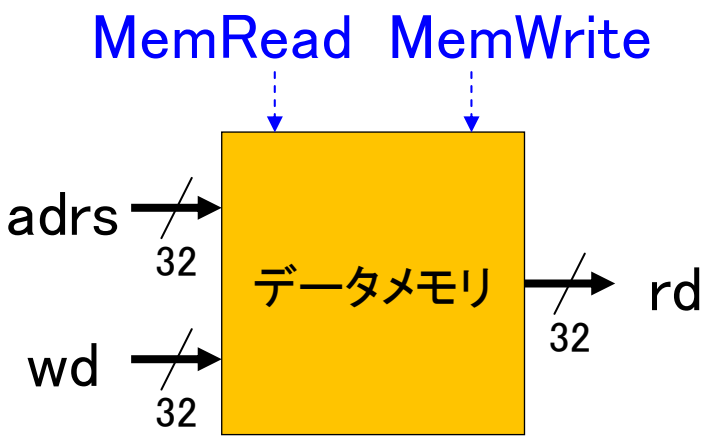


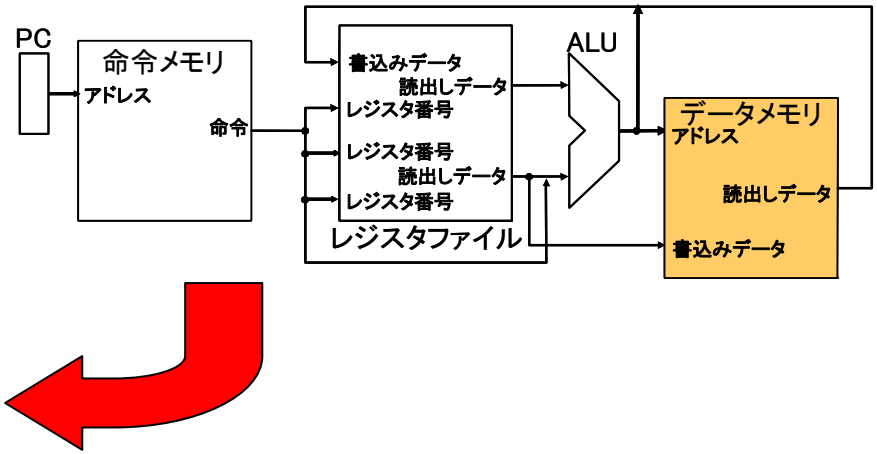


部品の準備(データメモリ)

データメモリ:プログラム実行において必要となるデータを格納

信号の意味	入出力	信号名	ビット幅
読み書きする番地	入力	adrs	32
書き込みデータ	入力	wd	32
読み出しデータ	出力	rd	32
書き込み要求(1のとき要求)	入力	MemWrite	1
読み出し要求(1のとき要求)	入力	MemRead	1

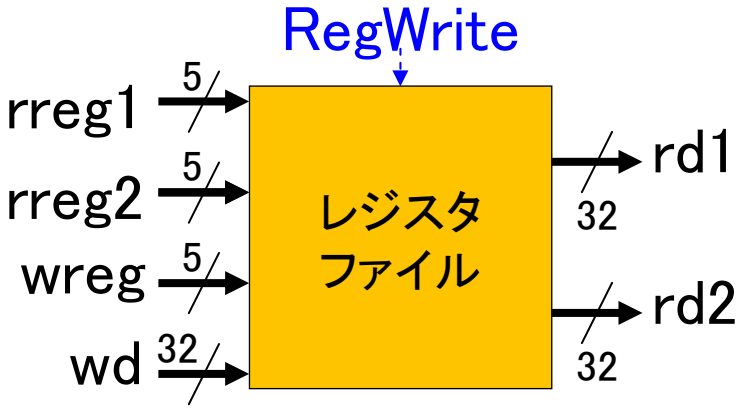


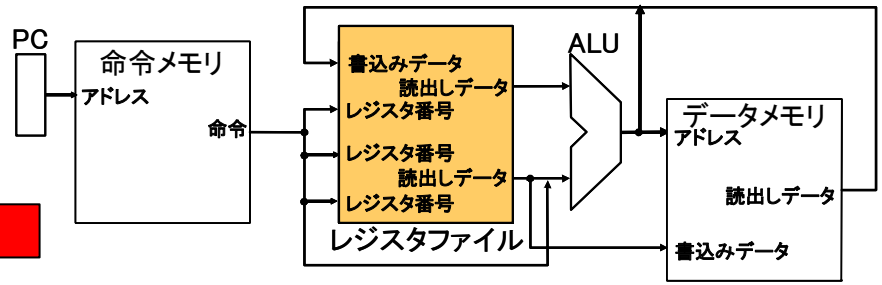


部品の準備(レジスタファイル 1/2)

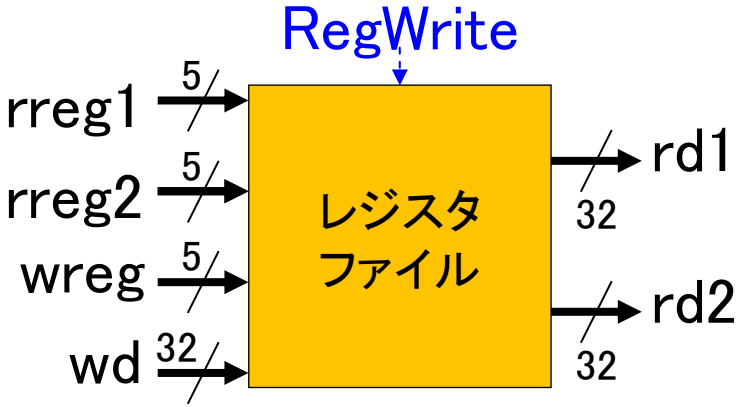
レジスタファイル:複数のレジスタにより構成

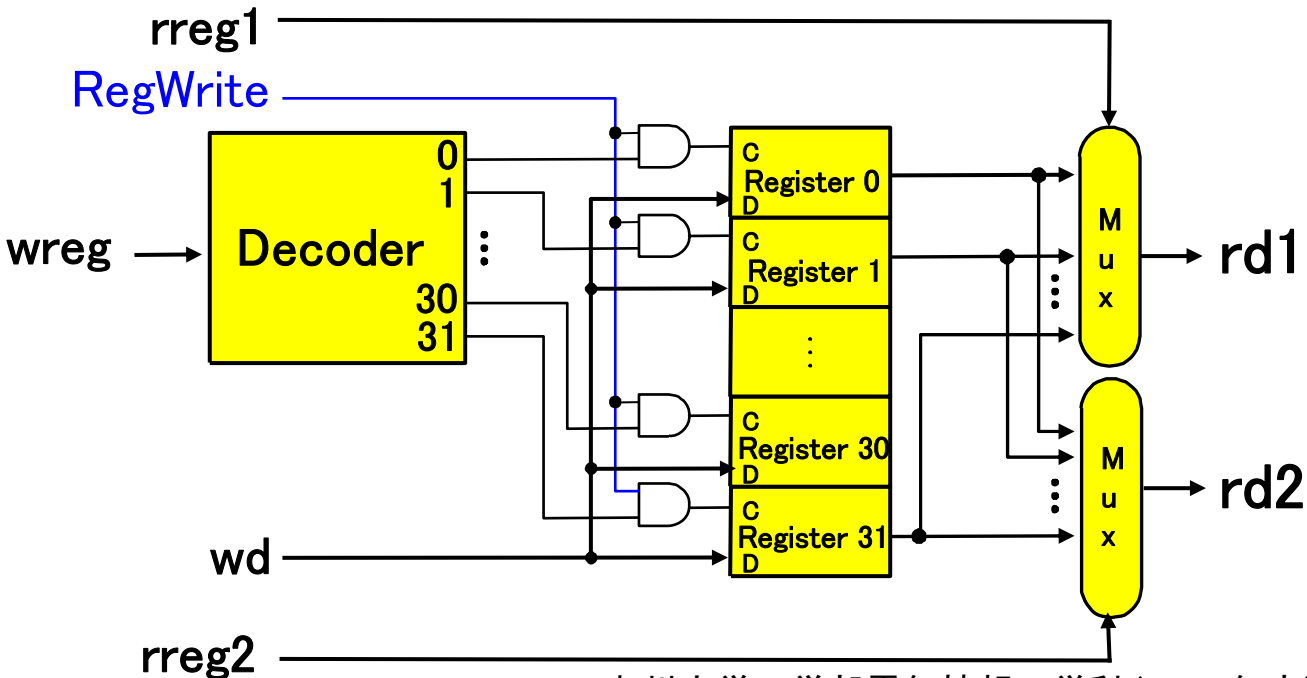
信号の意味	入出力	信号名	ビット幅
レジスタの読み出し値1	出力	rd1	32
レジスタの読み出し値2	出力	rd2	32
レジスタの書き込み値	入力	wd	32
rd1 に値を出力するレジスタを選択	入力	rreg1	5
rd2 に値を出力するレジスタを選択	入力	rreg2	5
wd の値を書き込むレジスタを選択	入力	wreg	5
レジスタに書き込みを行うとき1	入力	RegWrite	1





部品の準備(レジスタファイル 2/2)

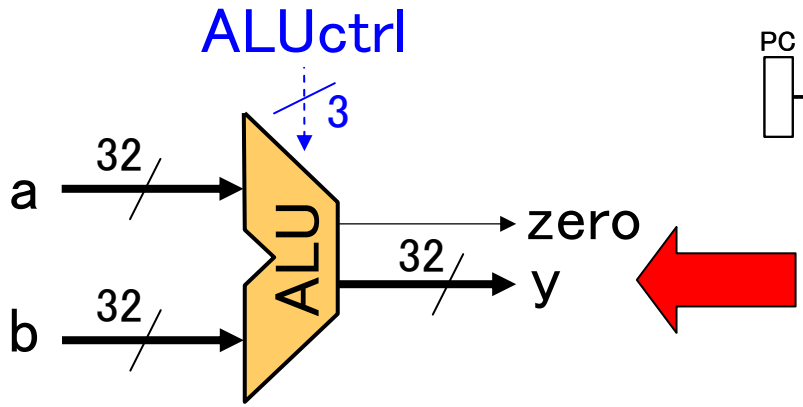


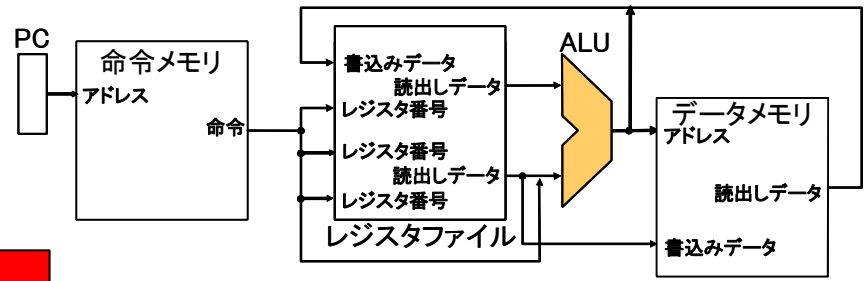


部品の準備（ALU）

ALU：指定された算術論理演算を実行

信号の意味	入出力	信号名	ビット幅
演算対象となる入力オペランド	入力	a	32
演算対象となる入力オペランド	入力	b	32
演算結果出力	出力	y	32
ゼロ判定（結果が 0 のときに 1）	出力	zero	1
ALU制御（000: AND, 001: OR, 010: +, 110: −, 111: slt）	入力	ALUctrl	3



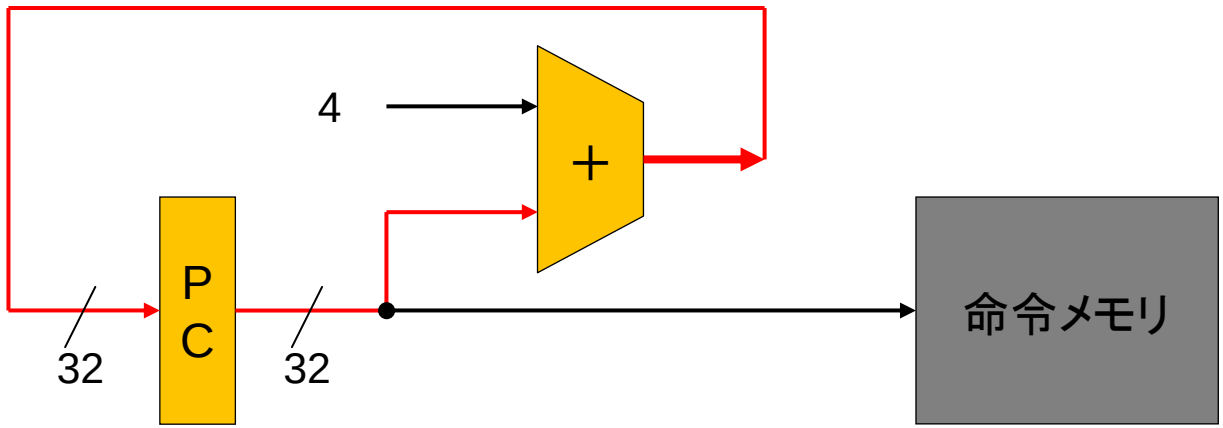


プログラムカウンタ周りの設計

プログラムカウンタ(PC): 次に実行する命令の番地を記憶

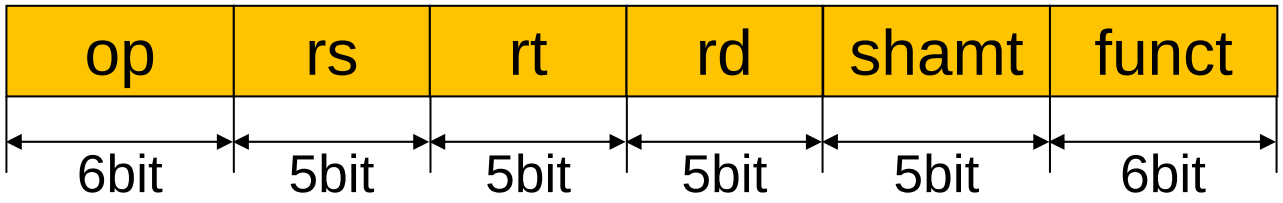
命令を実行するたびに PC を更新

- 分岐命令以外: 次の命令 PC ← PC + 4
- 分岐命令の場合:
  - 分岐条件成立時: PC ← 分岐先命令の番地
  - 分岐条件非成立時: 次の命令 PC ← PC + 4



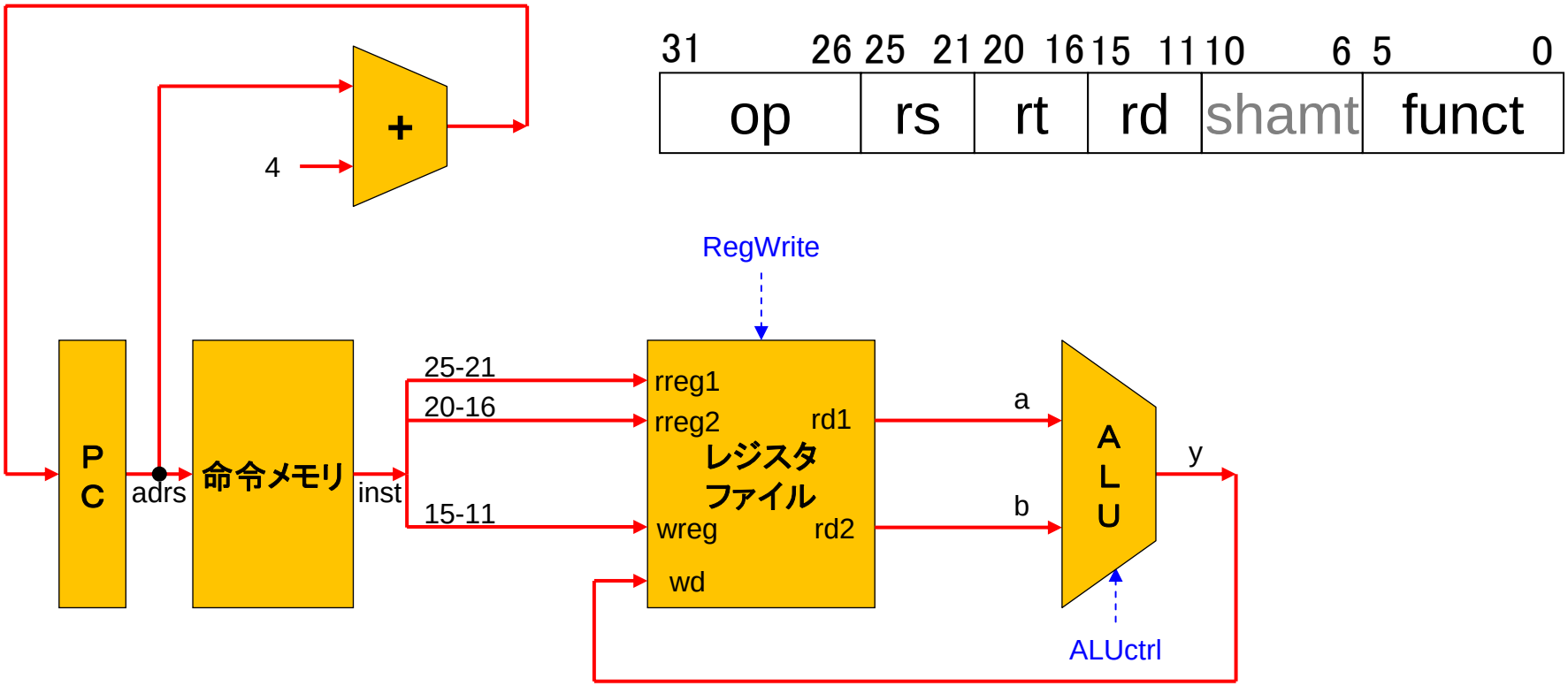
算術論理演算命令用データパスの設計(1)

R形式



命令	op	rs	rt	rd	shamt	funct
add $○, $△, $□	000000	$△	$□	$○	00000	100000
sub $○, $△, $□	000000	$△	$□	$○	00000	100010
slt $○, $△, $□	000000	$△	$□	$○	00000	101010
and $○, $△, $□	000000	$△	$□	$○	00000	100100
or $○, $△, $□	000000	$△	$□	$○	00000	100101

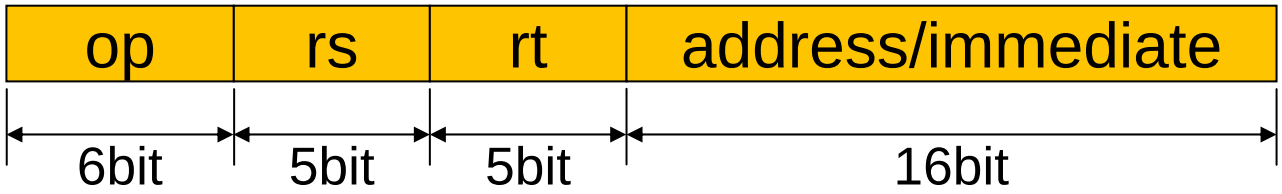
算術論理演算命令用データパスの設計(2)



命令区分	命令の内容	例	意味
算術論理演算	add, sub, and, or, slt	add $s1, $s2, $s3	$s1 = $s2 + $s3

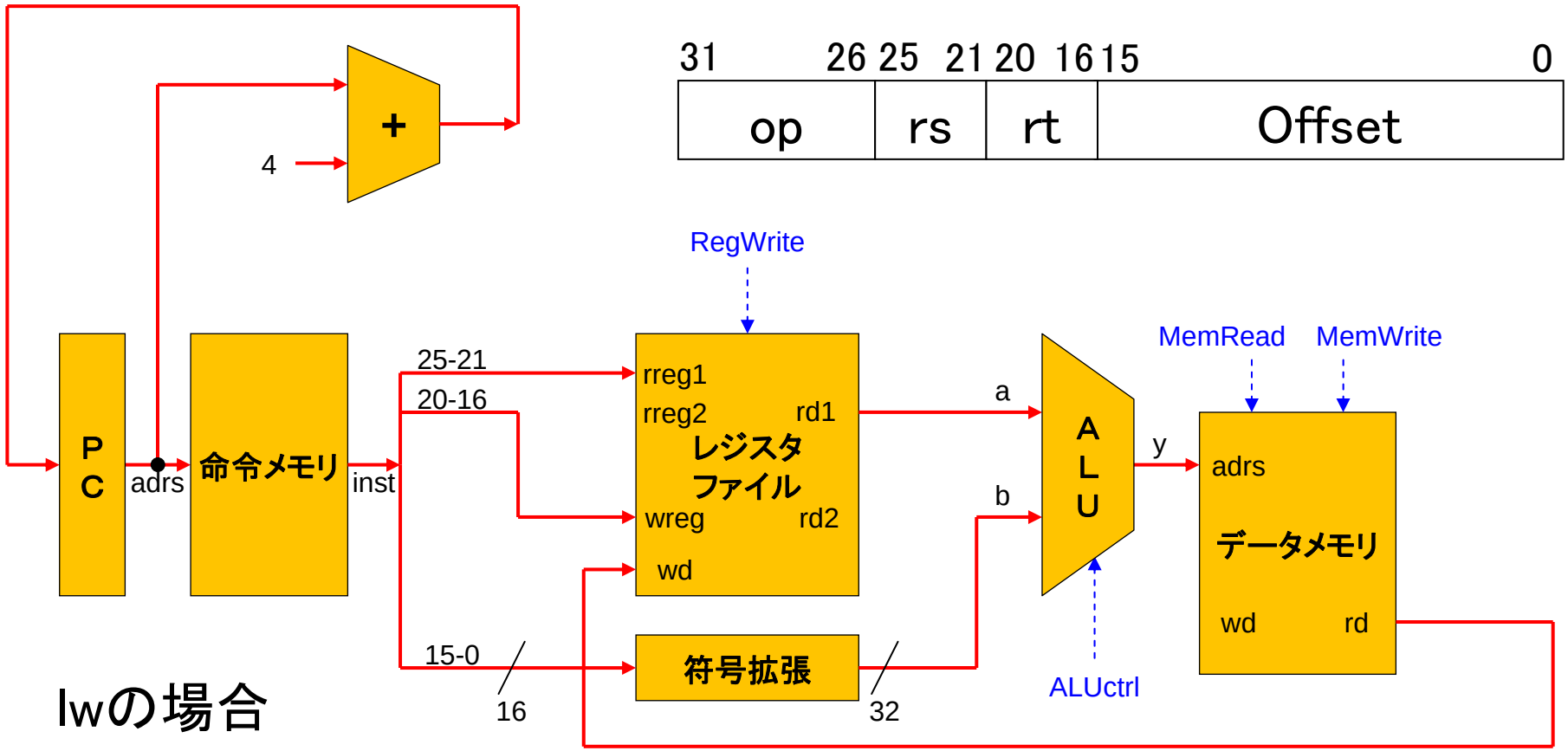
lw/sw命令用データパスの設計（1）

I形式



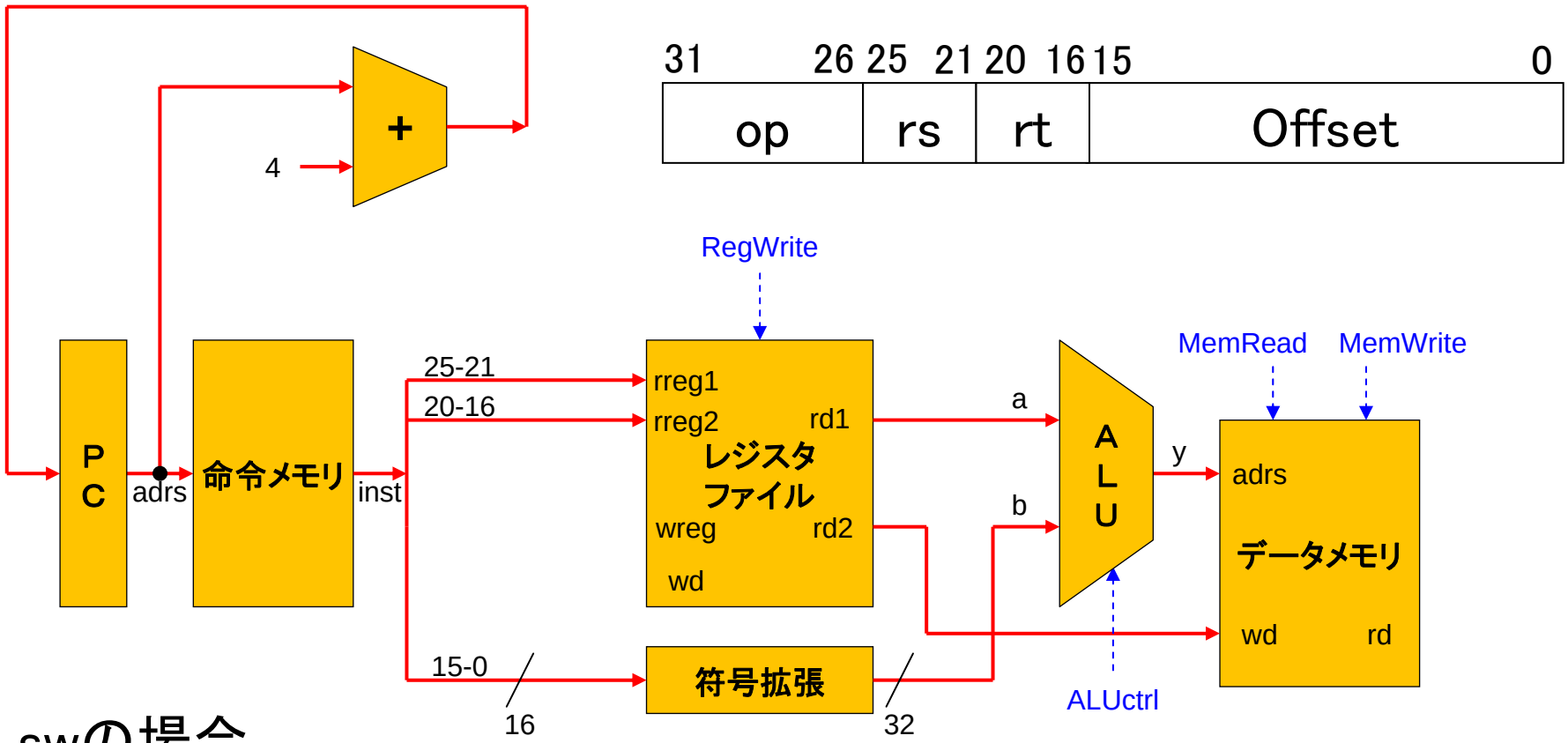
命令	op	rs	rt	address/immediate
lw $○, n($△)	100011	$△	$○	n
sw $○, n($△)	101011	$△	$○	n

lw/sw命令用データパスの設計(2)



命令区分	命令の内容	例	意味
データ転送	ロードワード	lw $s1, 100($s2)	$s1に，メモリの[$s2+100]番地のワードデータを読込み

lw/sw命令用データパスの設計(3)

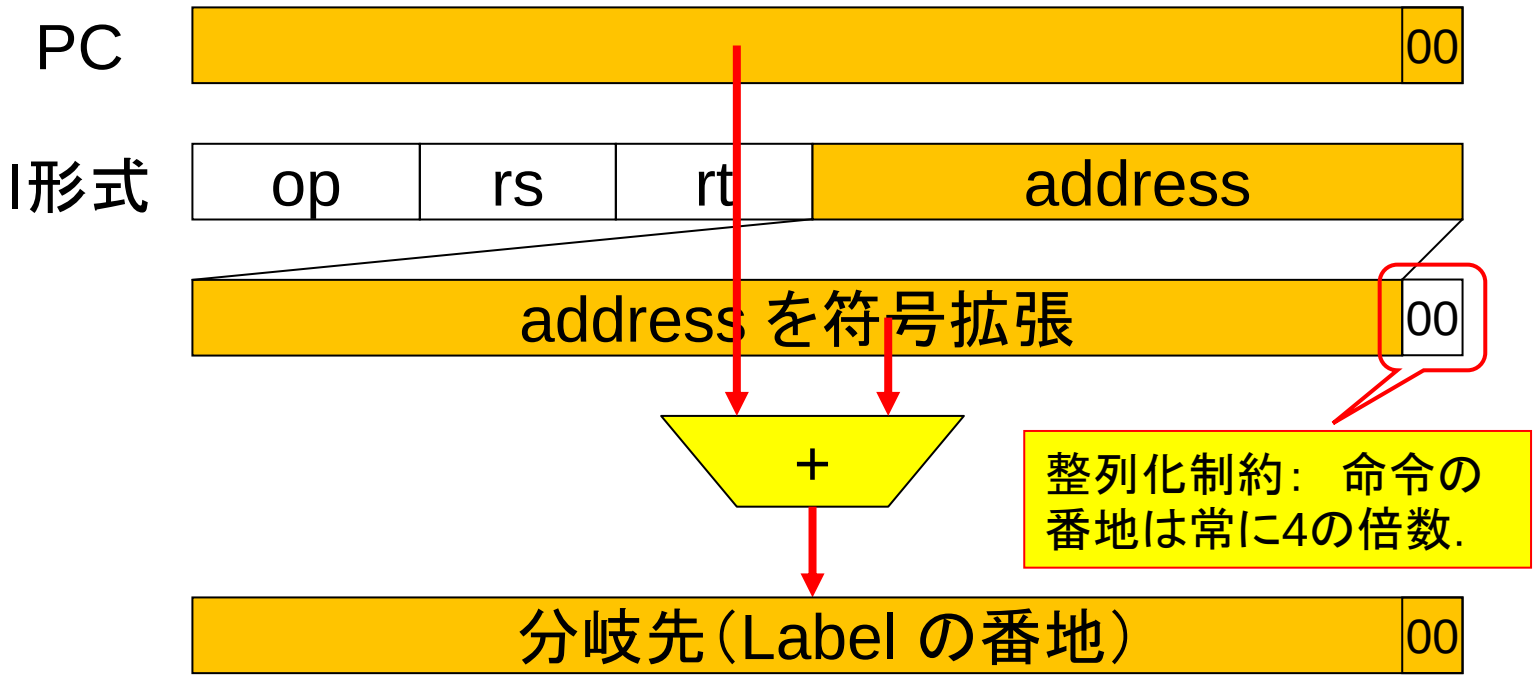


swの場合

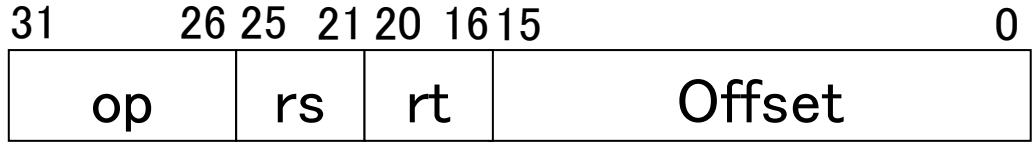
命令区分	命令の内容	例	意味
データ転送	ストアワード	sw $s1, 100($s2)	メモリの[$s2+100]番地に，$s1のワードデータを書込み

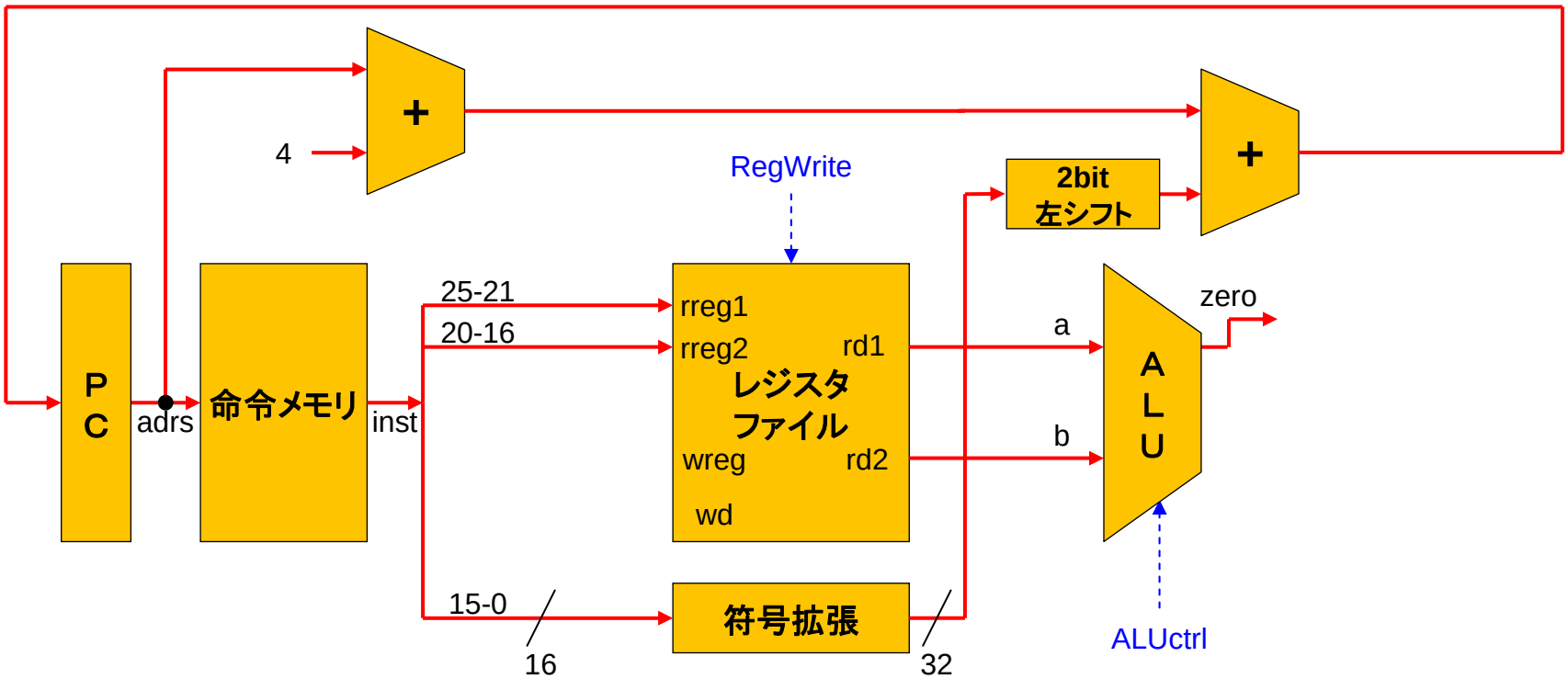
beq 命令用データパスの設計(1)

PC相対アドレッシング: beq, bne 命令の分岐先番地



beq 命令用データパスの設計(2)

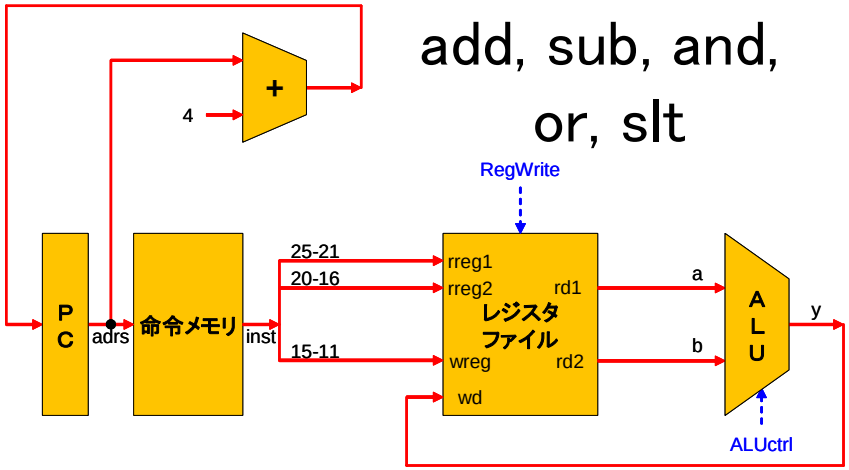




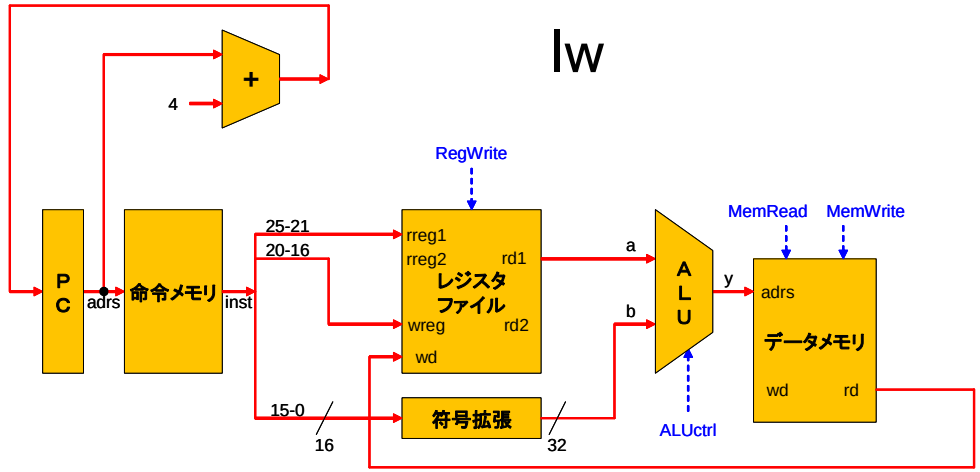
命令区分	命令の内容	例	意味
分岐	条件付	beq $s1, $s2, L	もし、$s1==$s2ならラベルLへ分岐

各命令用のデータパス

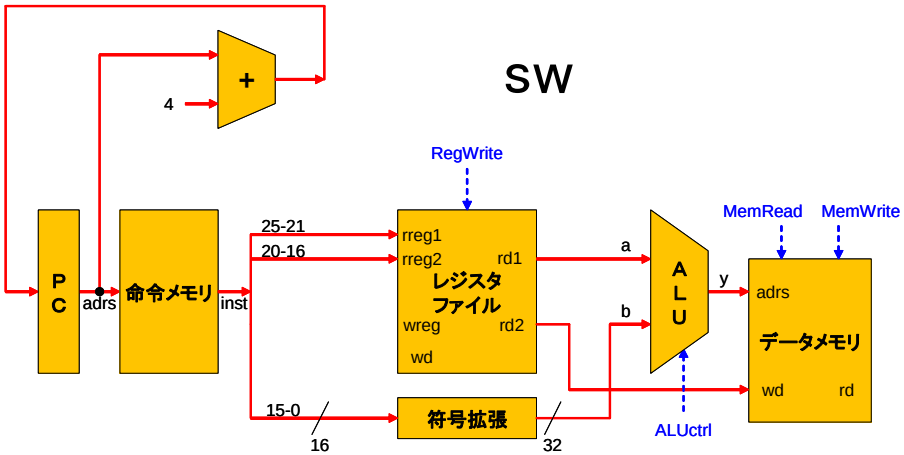
add, sub, and,
or, slt



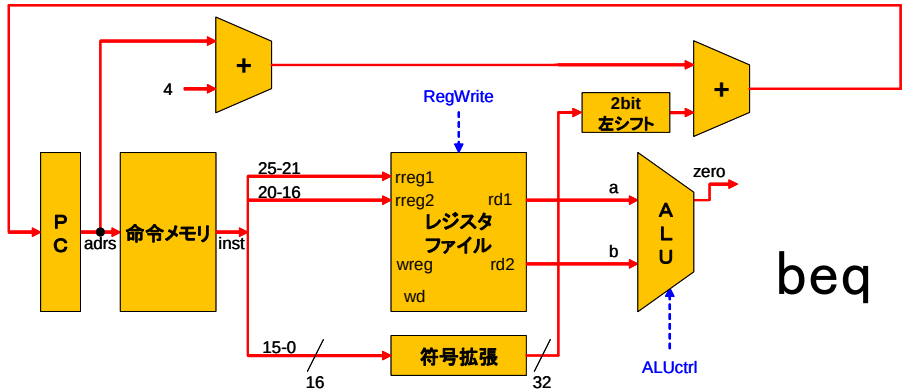
lw



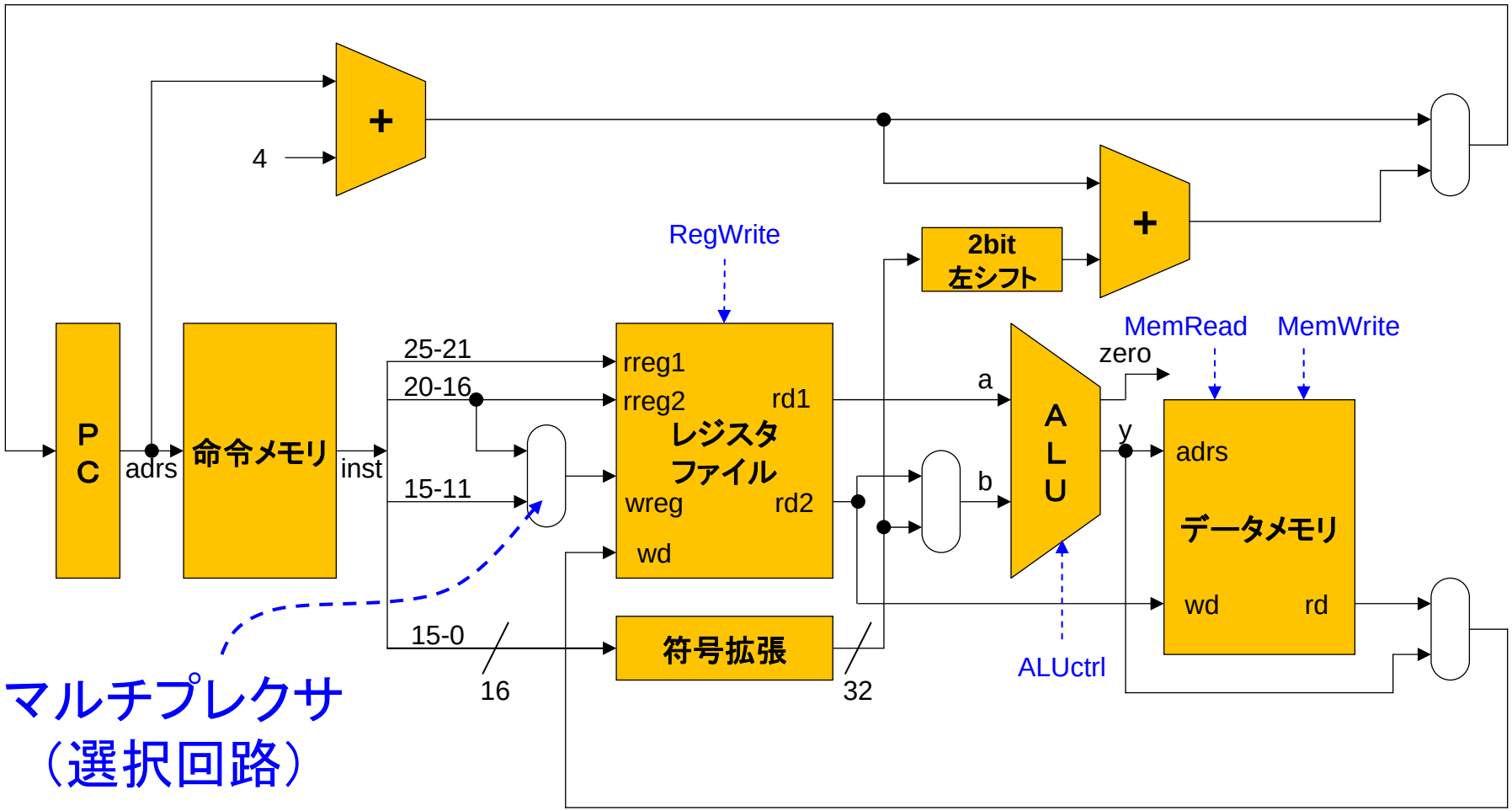
sw



beq



データパスの統合


+
4
+
RegWrite
2bit
左シフト
MemRead
MemWrite
zero
a
y
adrs
PC
adrs
命令メモリ
inst
25-21
20-16
15-11
15-0
rreg1
rreg2
rd1
レジスタ
ファイル
wreg
rd2
wd
b
ALU
データメモリ
wd
rd
符号拡張
16
32
ALUctrl
マルチプレクサ
（選択回路）