電気電子システム学科

コンピュータ工学 I

Part 3

Rev. 2020.02.02

情報処理技術者試験

情報処理技術者試験は、「情報処理の促進に関する法律」に基づき経済産業省が、情報処理技術者としての「知識・技能」が一定以上の水準であることを認定している国家試験です。

（https://www.jitec.ipa.go.jp/ より引用）

外部検定試験による単位認定

詳細は、履修ガイドを参照

(2)「情報科学セミナーⅠ・Ⅱ・Ⅲ」

外部検定試験	認定科目	単位数		科目群	対象学科
・ドットコムマスターアドバンス試験 (ドットコムマスターシングルスター試験) ・ITパスポート試験 ・情報セキュリティマネジメント試験 のうちいずれかの試験	情報科学セミナーⅠ	2		専門教育科目	教育学部・経営学部・獣医学部を除く全学科 (情報工学科は2年生以上ITコースのみ)
情報活用試験1級	情報科学セミナーⅠ	2			
プログラマ認定 システムエンジニア認定	情報科学セミナーⅡ	2			
基本情報技術者試験	情報科学セミナーⅠ	2	4		教育学部・経営学部・獣医学部を除く全学科
	情報科学セミナーⅡ	2			
ソフトウェア開発技術者試験 または応用情報技術者試験 以上のレベルの試験　＊	情報科学セミナーⅠ	2	6		
	情報科学セミナーⅡ	2			
	情報科学セミナーⅢ	2			

コンピュータの基本構成とCPU

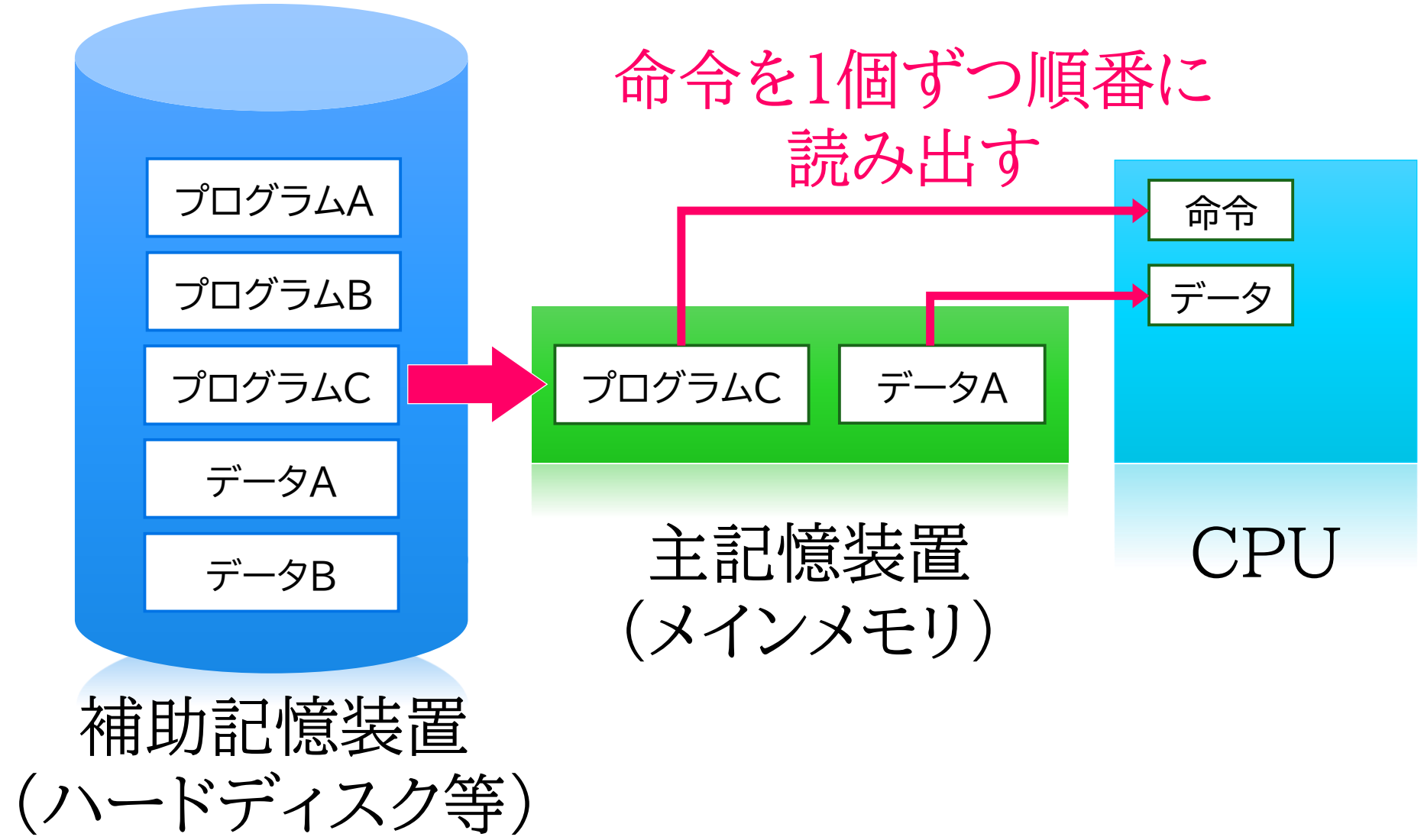
✤内容

1. CPUの構成要素
2. 命令サイクル
3. アセンブリ言語
4. アドレッシング方式
5. CPUの高速化
6. CPUの性能評価

コンピュータの構成装置

- **中央処理装置（CPU）**
  主記憶装置から命令を読み込み、実行を行う。
- **主記憶装置**
  CPUで実行するプログラム（命令の集合）やデータを記憶する。
- **補助記憶装置**
- **入力装置**
- **出力装置**

プログラムの実行

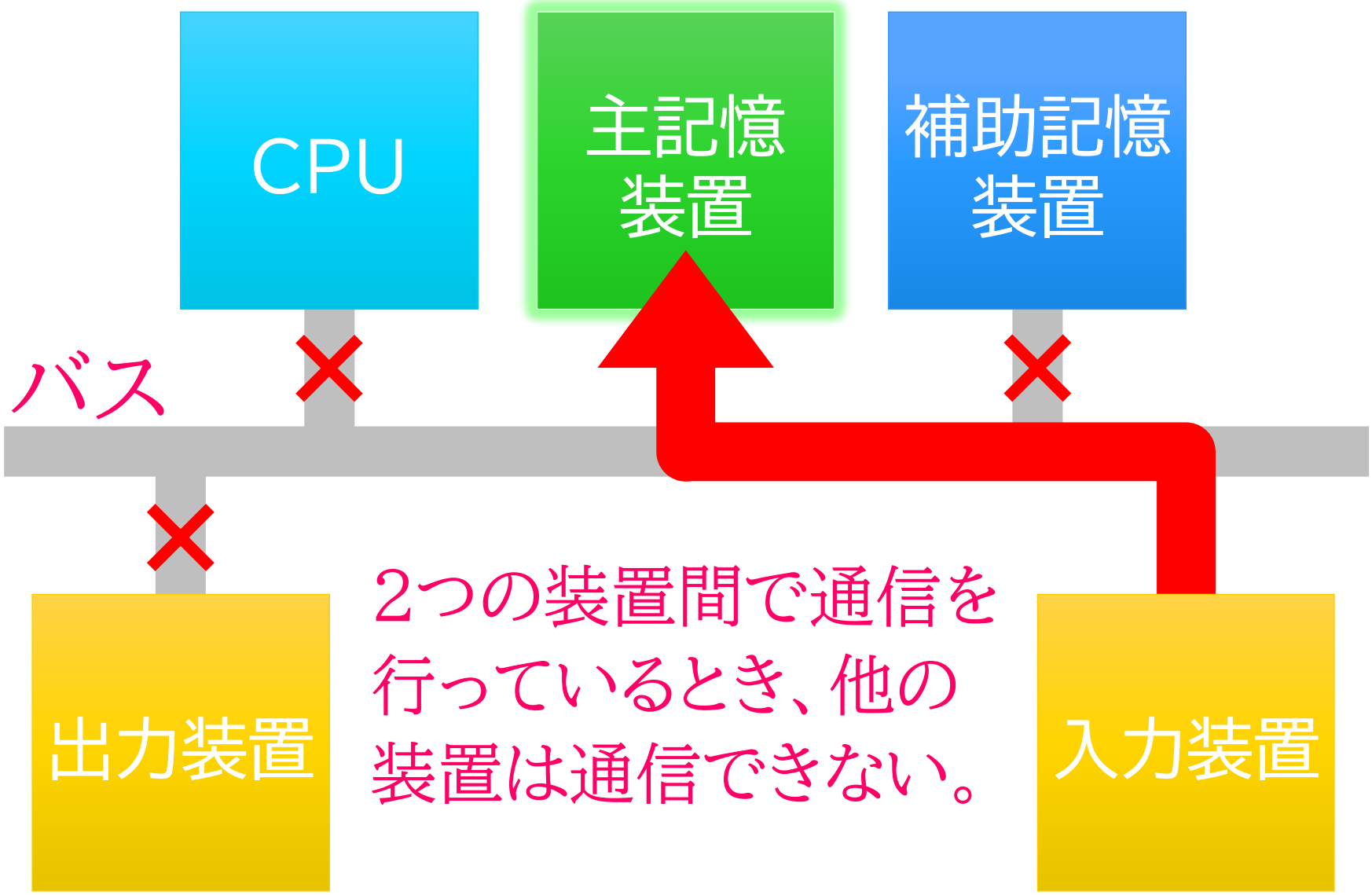

命令を1個ずつ順番に
読み出す
プログラムA
プログラムB
プログラムC
データA
データB
プログラムC
データA
命令
データ
補助記憶装置
（ハードディスク等）
主記憶装置
（メインメモリ）
CPU

バス方式

バス

コンピュータ内の各装置がデータ通信に用いる共通の通信経路

バスによる装置間の通信



バスの種類

重要

- **データバス**

  データを送る。

- **アドレスバス**

  主記憶装置のアドレス（番地）を送る。

- **コントロールバス**

  データの読み書きの制御信号を送る。

主記憶装置

記憶場所

数値化された命令やデータを記憶

主記憶装置内のデータを読み書きするときは、対象のデータのアドレスを指定する必要がある。

機械語とアセンブリ言語

CPUの機能	命令番号	命令語
データ転送	10	LD
加算	24	ADDA
減算	25	SUBA
論理積	34	AND
論理和	35	OR

機械語　アセンブリ言語

CPUの機能1つ1つに命令番号が与えられている。

CPUの構成要素

重要

✤ レジスタ

略称	名称
MDR	記憶データ用レジスタ
MAR	記憶番地指定用レジスタ
PC	プログラムカウンタ
IR	命令レジスタ
GR0～GR7	汎用レジスタ
FR	フラグレジスタ

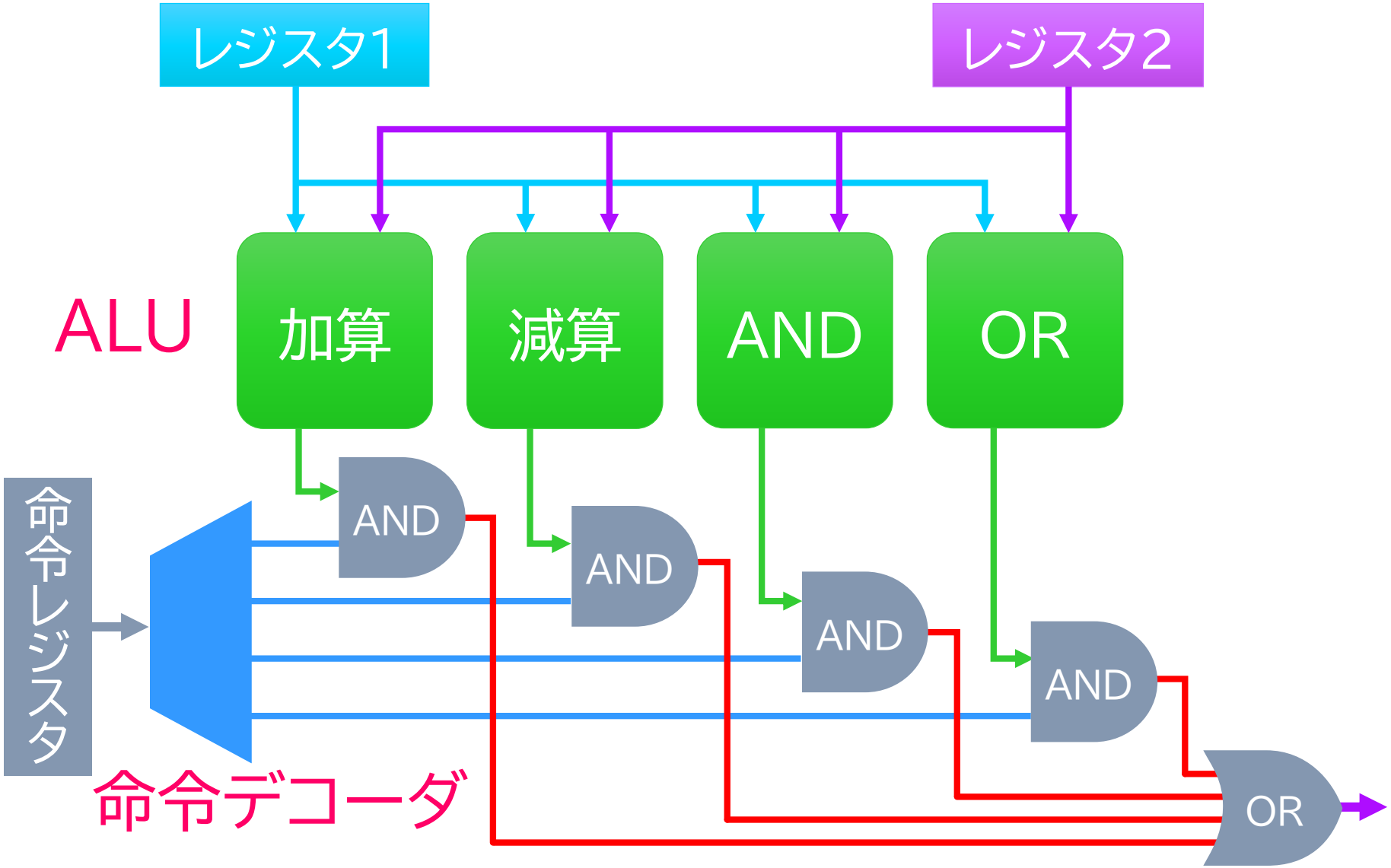
CPUの構成要素

重要

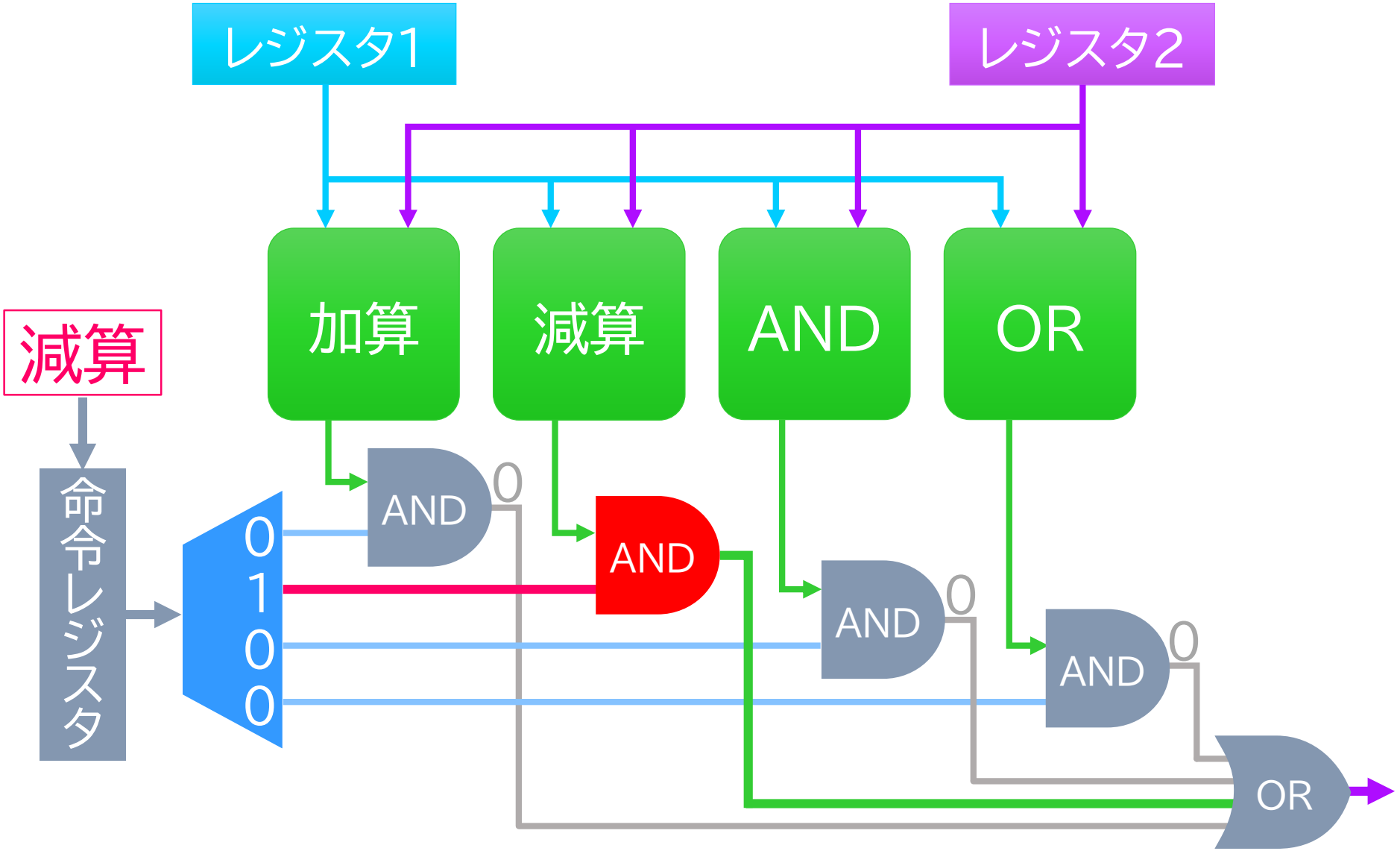
✣ その他装置

略称・名称	働き
ALU	算術演算や論理演算を行う
命令デコーダ	命令番号を解読し、各装置を制御する
コントロールユニット	コントロールバスに制御信号を送る

命令デコーダとALU



減算を行う場合



命令サイクル

重要

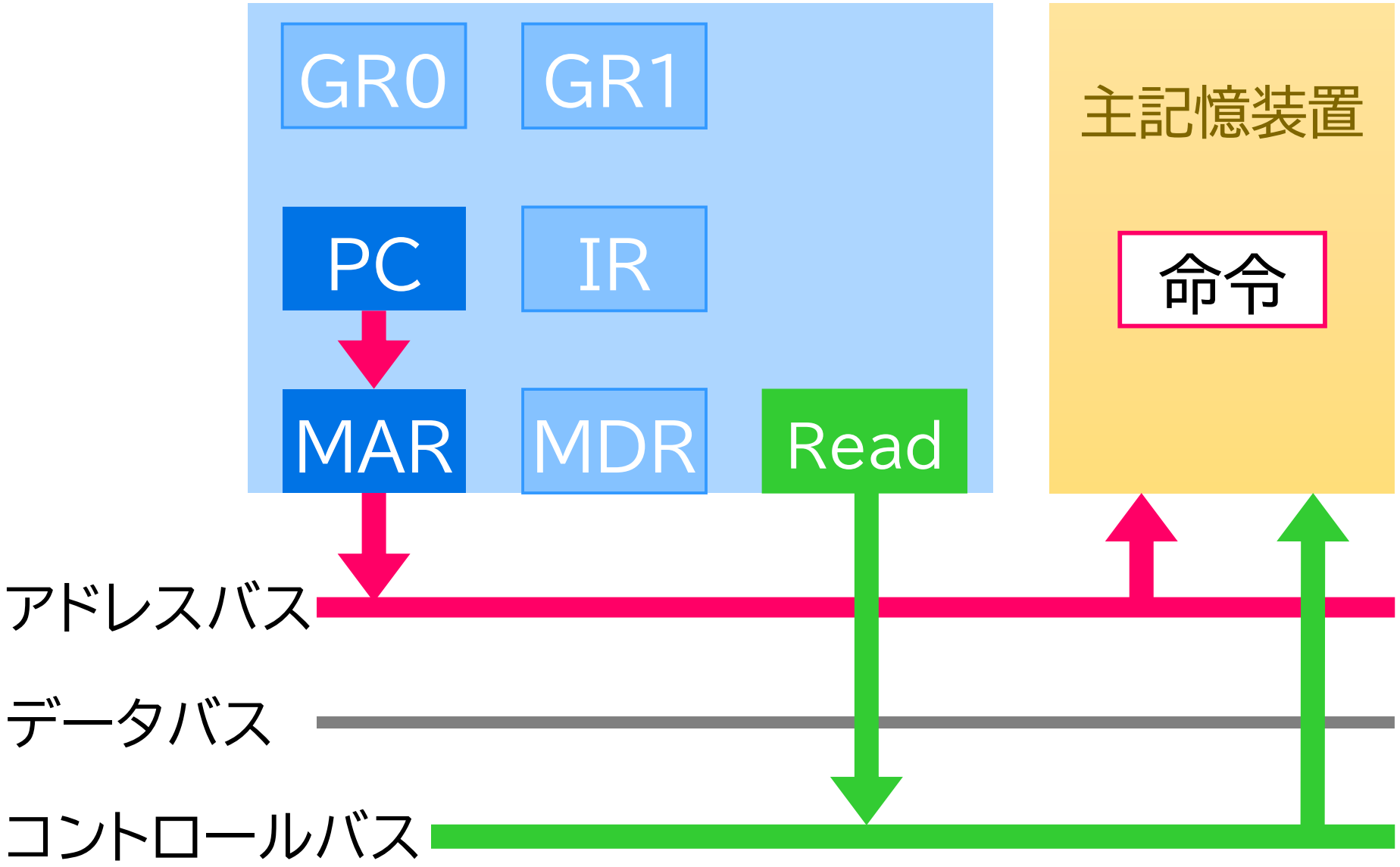
1つの命令を実行する手順

❶ 命令読み出し 過程(fetch)
❷ 命令解読 過程(decode)
❸ 命令実行 過程(execute)
❹ 結果書き込み 過程(writeback)

CPUの違いによって部分的に異なる場合があるので注意が必要。

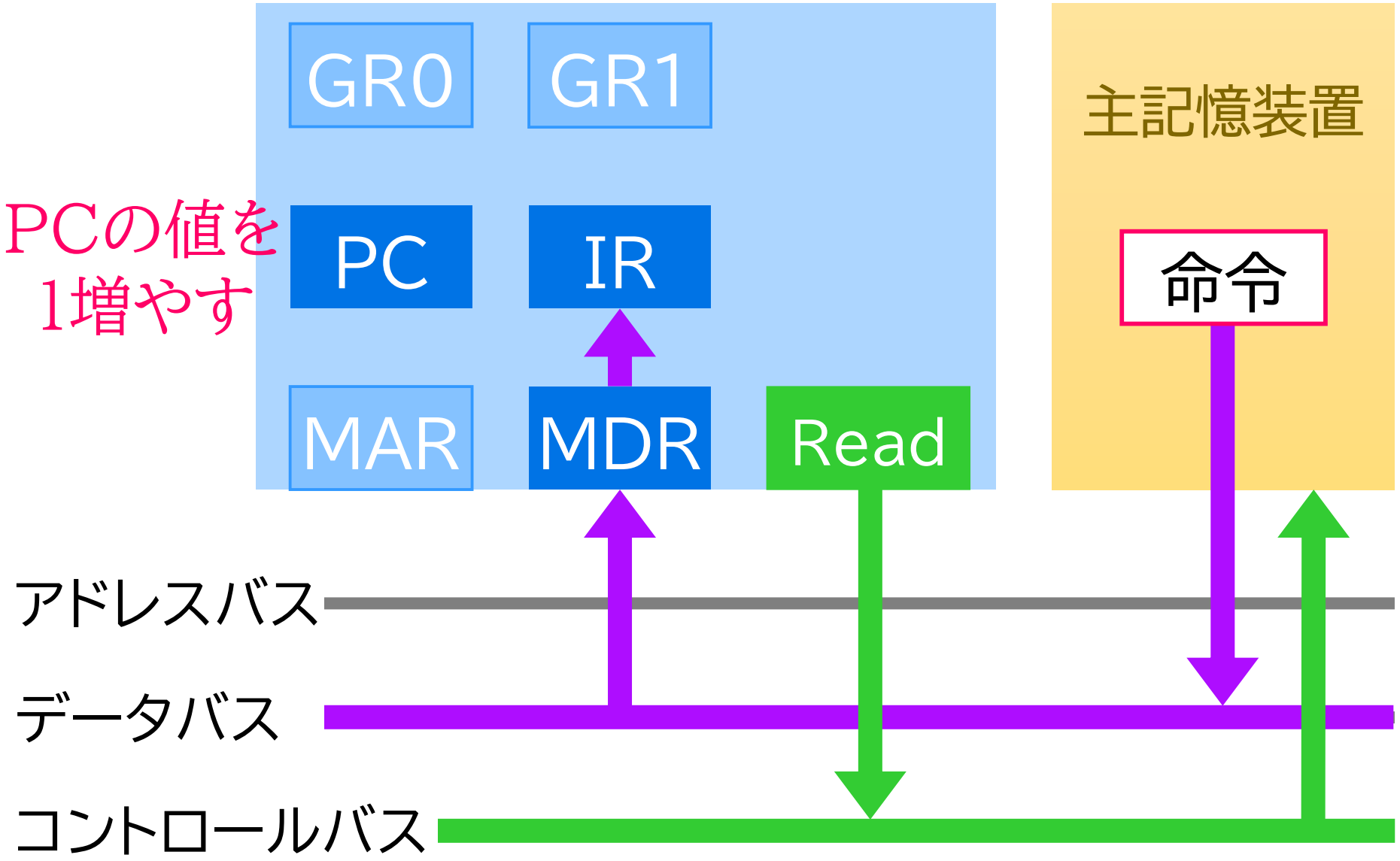
命令読み出し

重要



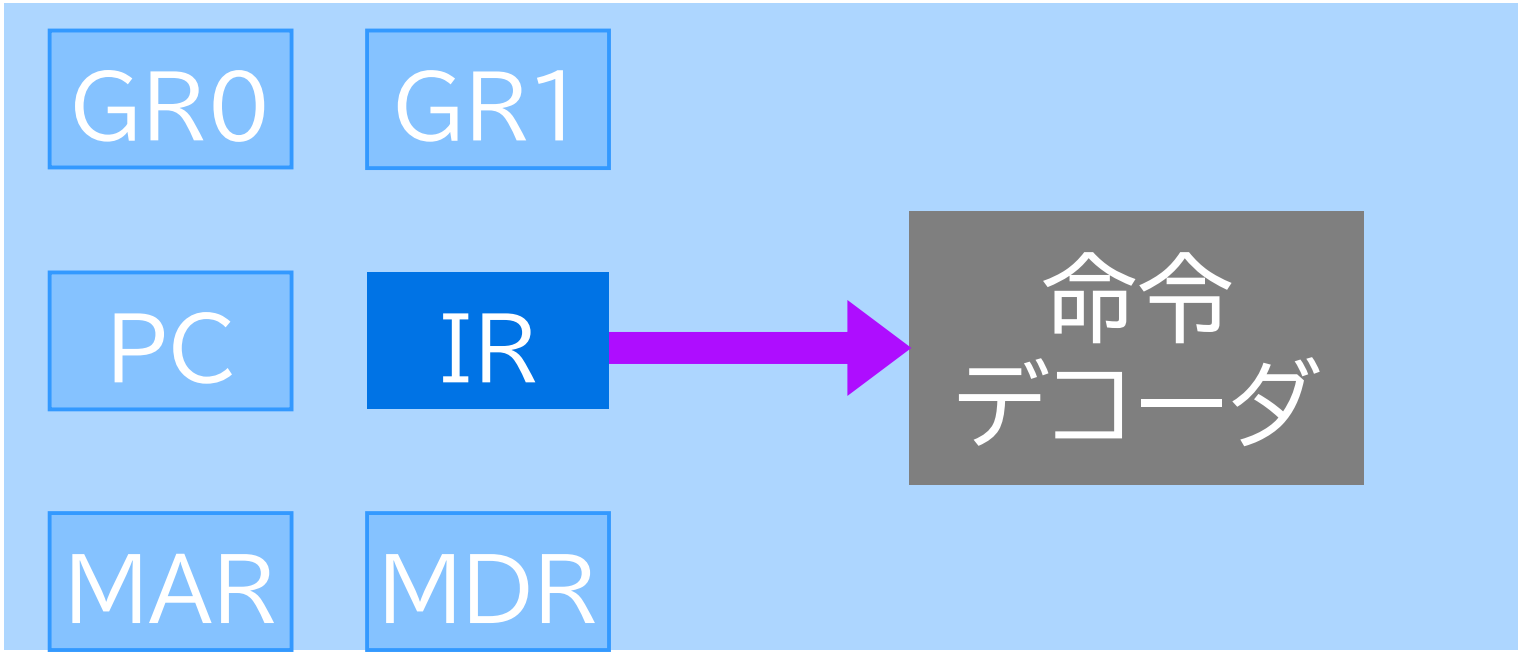
命令読み出し

重要


GR0
GR1
主記憶装置
PCの値を
1増やす
PC
IR
命令
MAR
MDR
Read
アドレスバス
データバス
コントロールバス

命令解読

重要



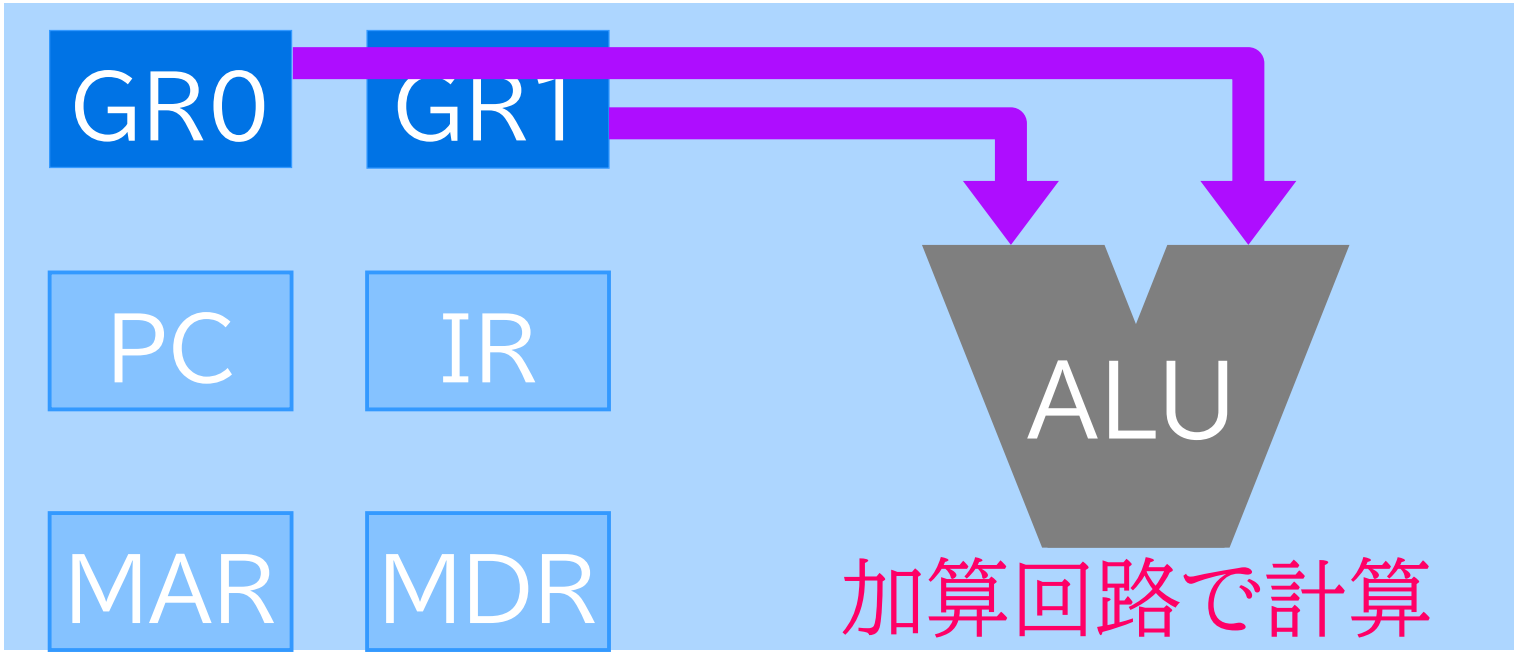
アドレスバス

データバス

コントロールバス

命令実行

重要



GR0＋GR1 を計算する場合

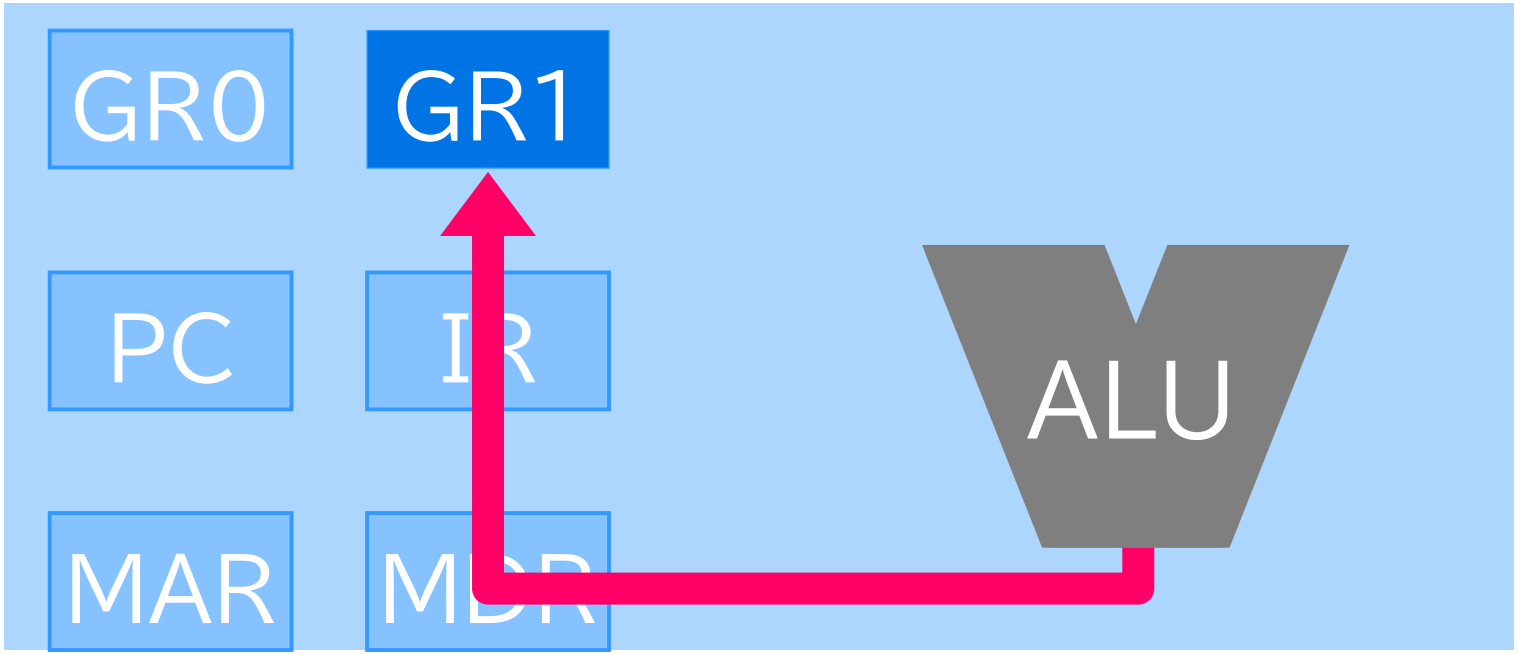
アドレスバス

データバス

コントロールバス

結果書き込み

重要



計算結果をGR1に書き込む場合

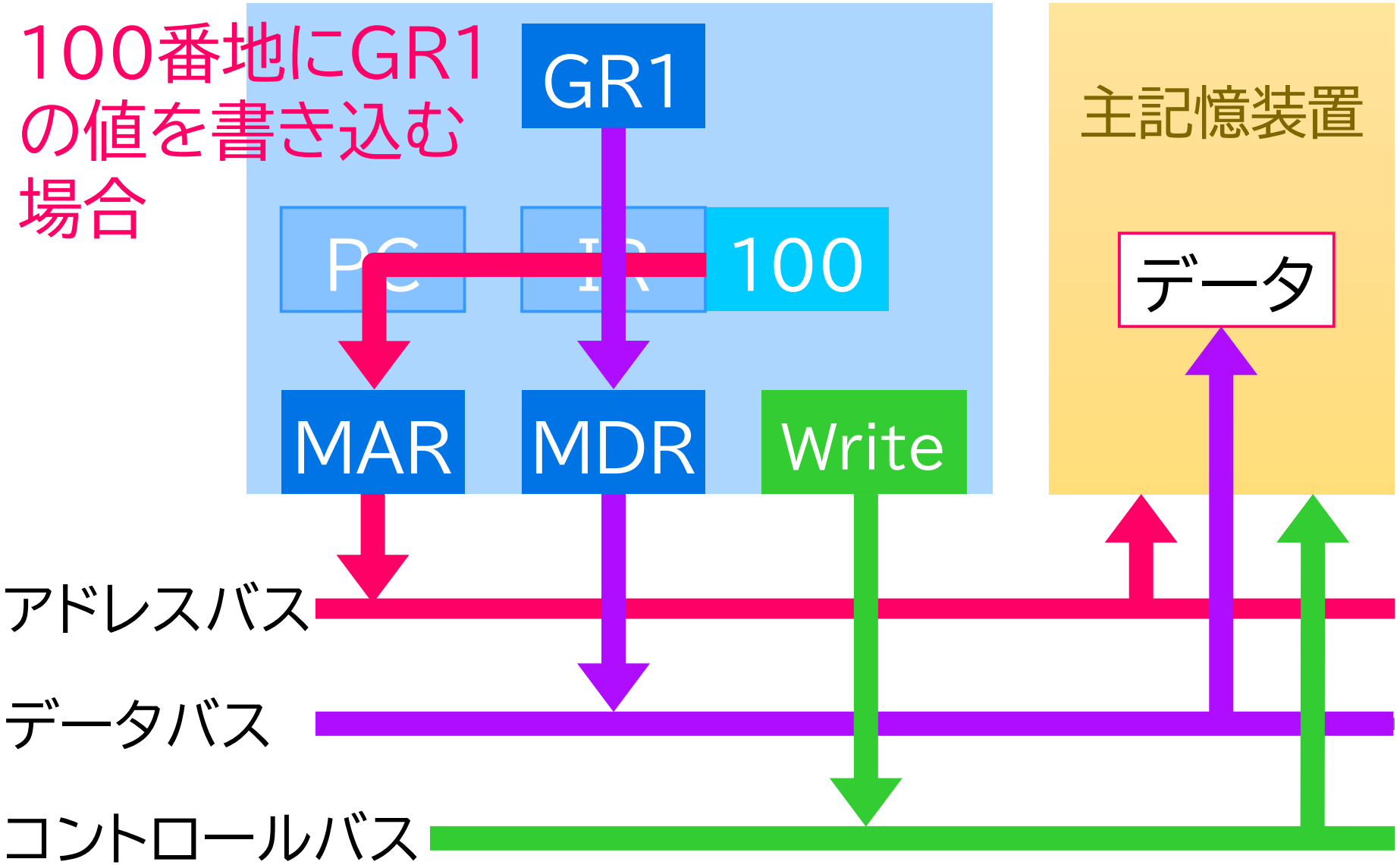
アドレスバス

データバス

コントロールバス

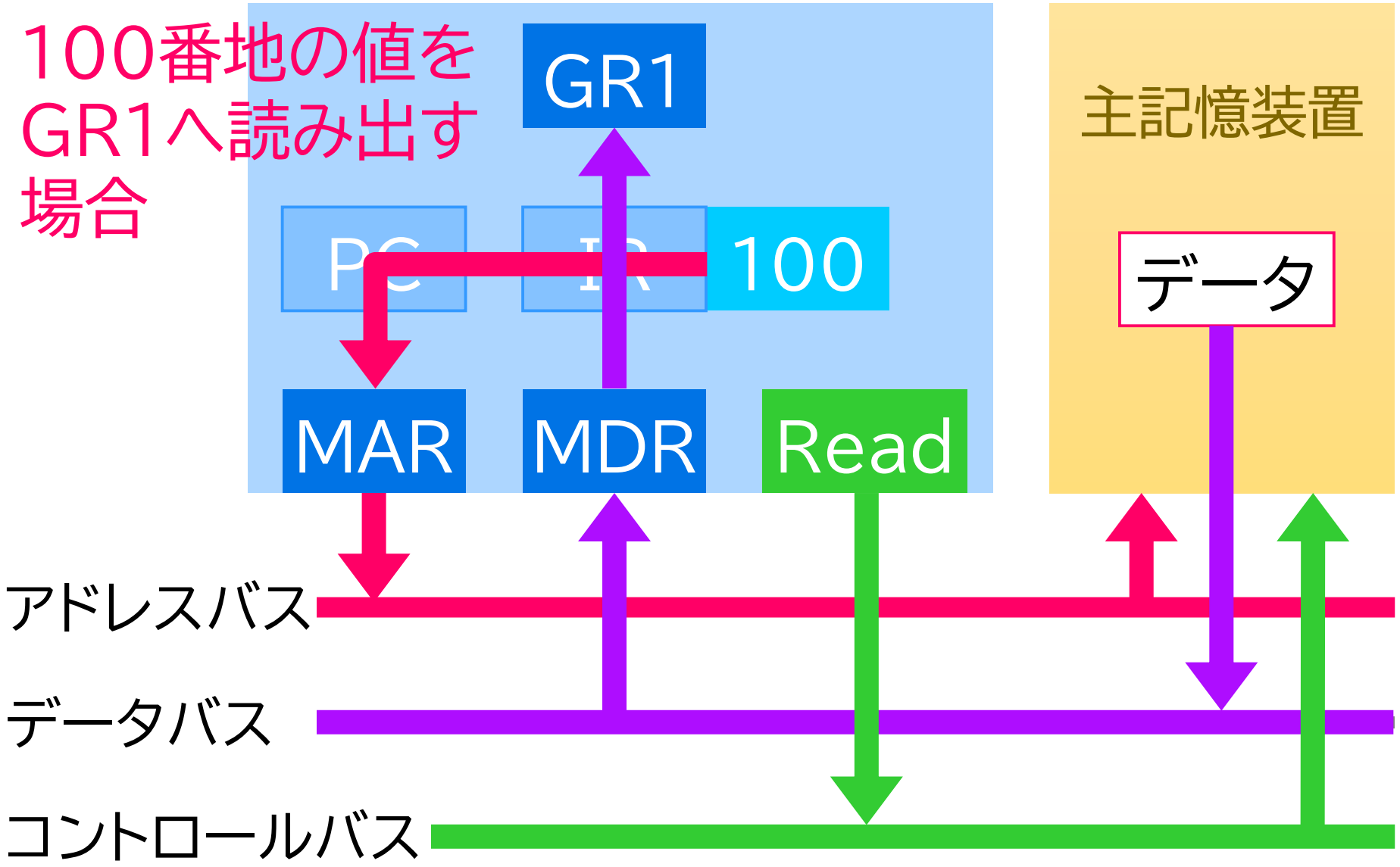
主記憶装置への書き込み

重要



主記憶装置からの読み出し

重要



命令読出し過程の手順

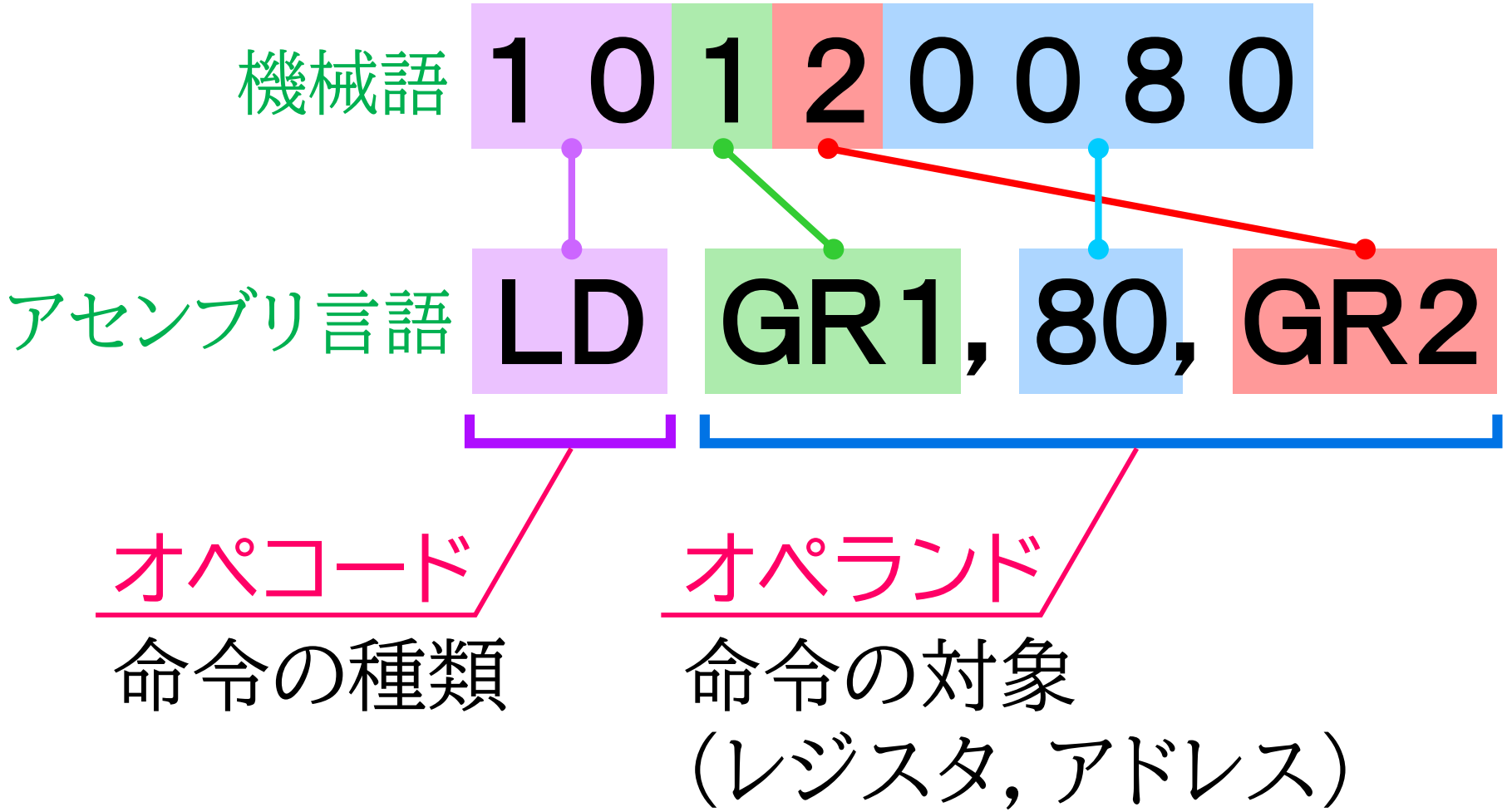
レジスタ PC に格納されている番地をレジスタ MAR に移す。続いて、その値を アドレス バスを経由して主記憶装置に送る。 主記憶装置から読み出した命令は、 データ バスを経由して、レジスタ MDR に送り、その後、レジスタ IR に格納する。 この処理の間にレジスタ PC の値に 1 を加算する。

アセンブリ言語

CASLⅡを取り扱う。

- データ転送命令
- 算術演算命令(加算、減算)
- 分岐命令
- 算術比較命令

命令の記述


機械語
1 0 1 2 0 0 8 0
アセンブリ言語
LD GR1, 80, GR2
オペコード
命令の種類
オペランド
命令の対象
(レジスタ, アドレス)

LD（ロード）

重要

レジスタ、または、主記憶装置内の値を、指定したレジスタに読み出す。

LD　r1，r2

レジスタr1 ← レジスタr2の値

LD　r1，adr

レジスタr1 ← adr番地の値

ST（ストア）

重要

レジスタの値を、主記憶装置内の指定したアドレスに書き込む。

ST　r1，adr

adr番地 ← レジスタr1の値

LAD（ロード アドレス）

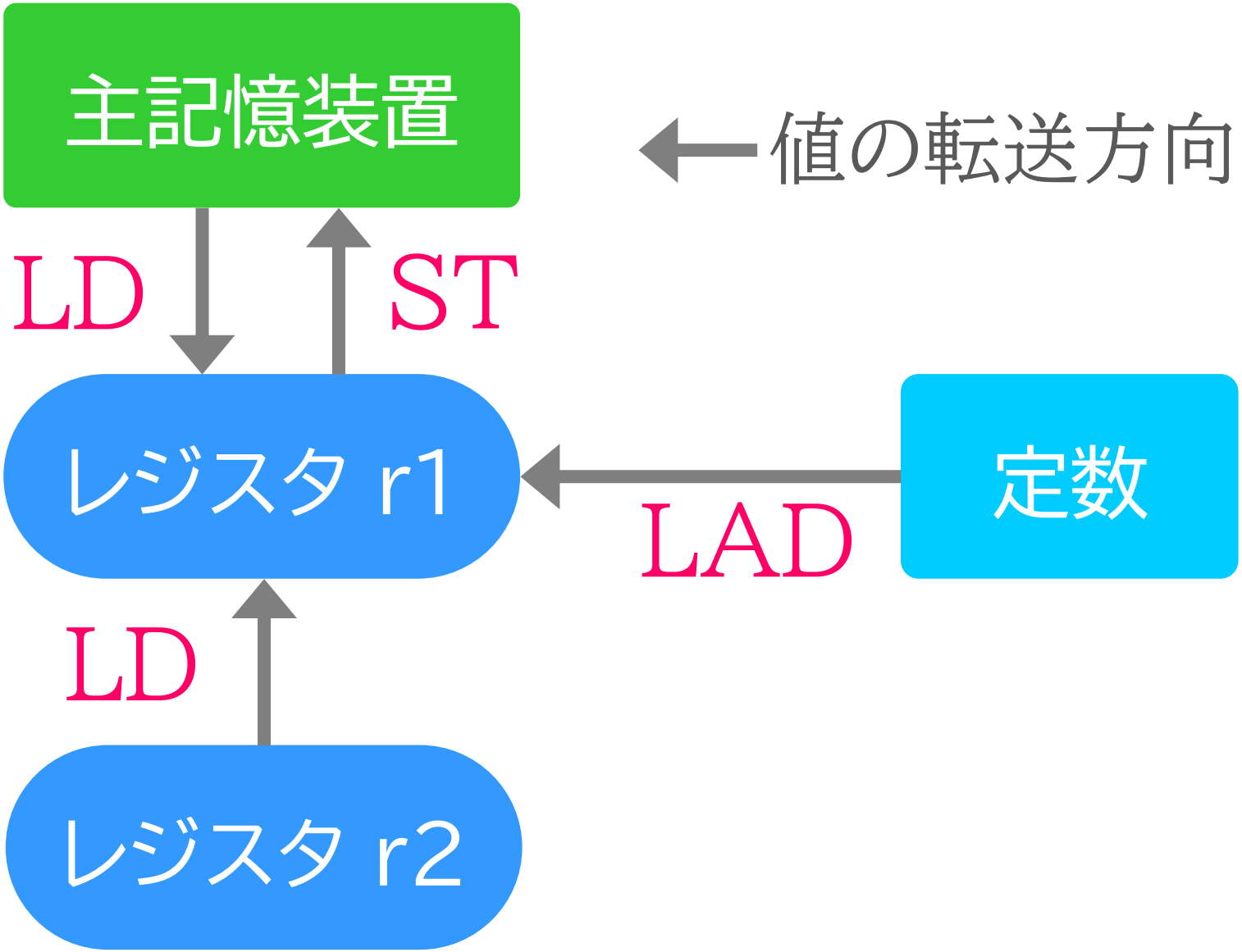
重要

レジスタに定数(番地)を代入する。

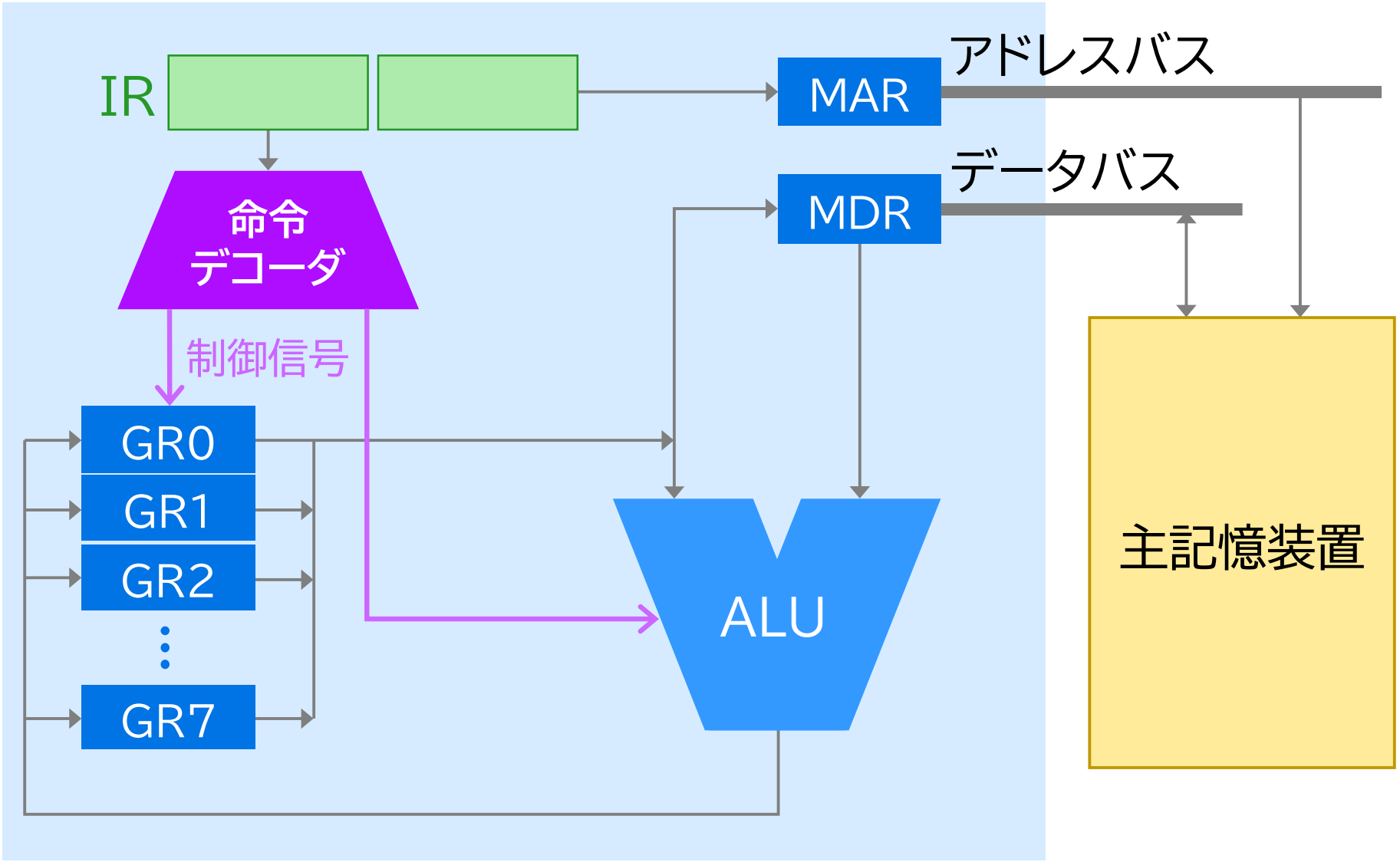
LAD　r1，adr

レジスタr1 ← adr

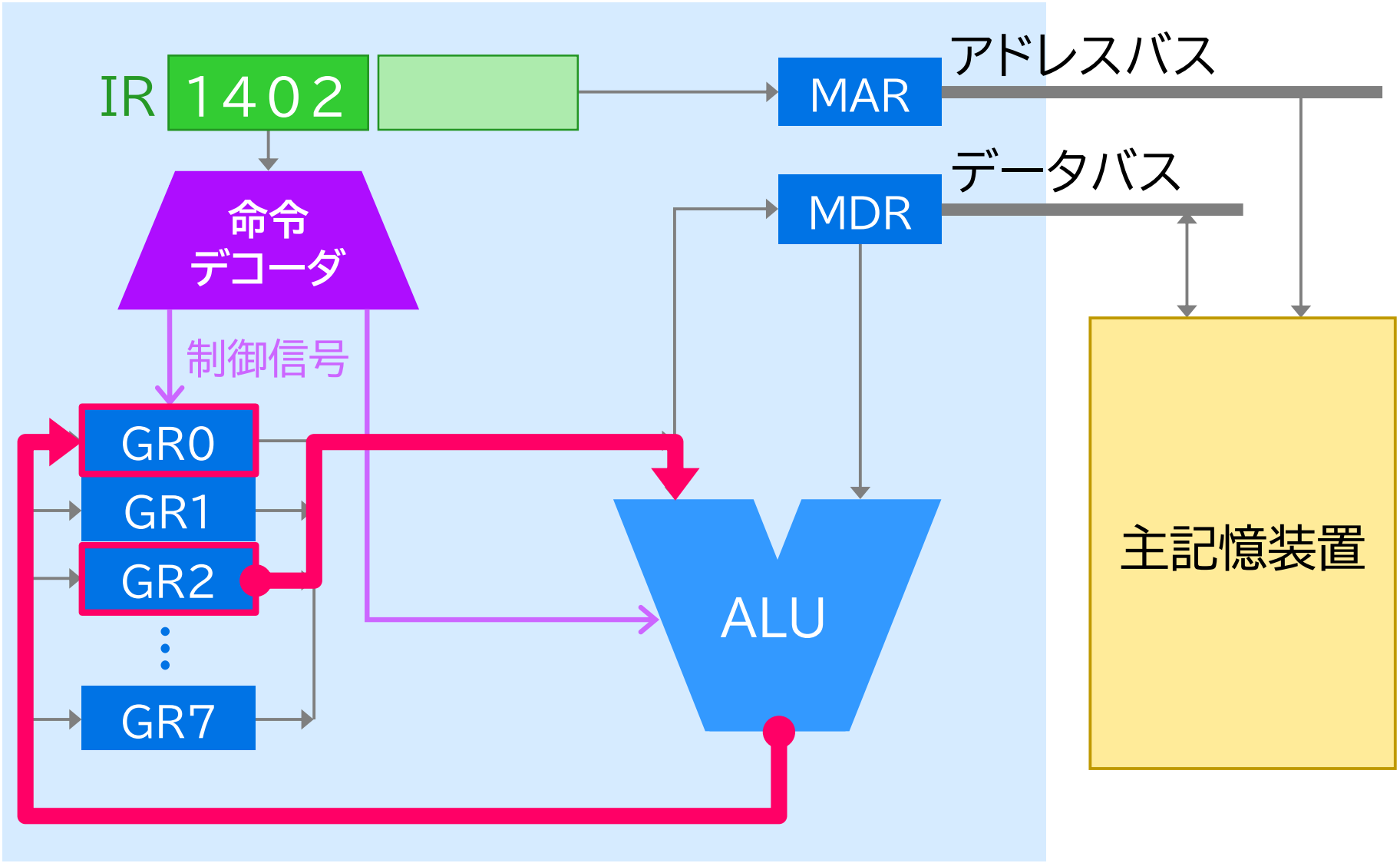
データ転送命令のまとめ



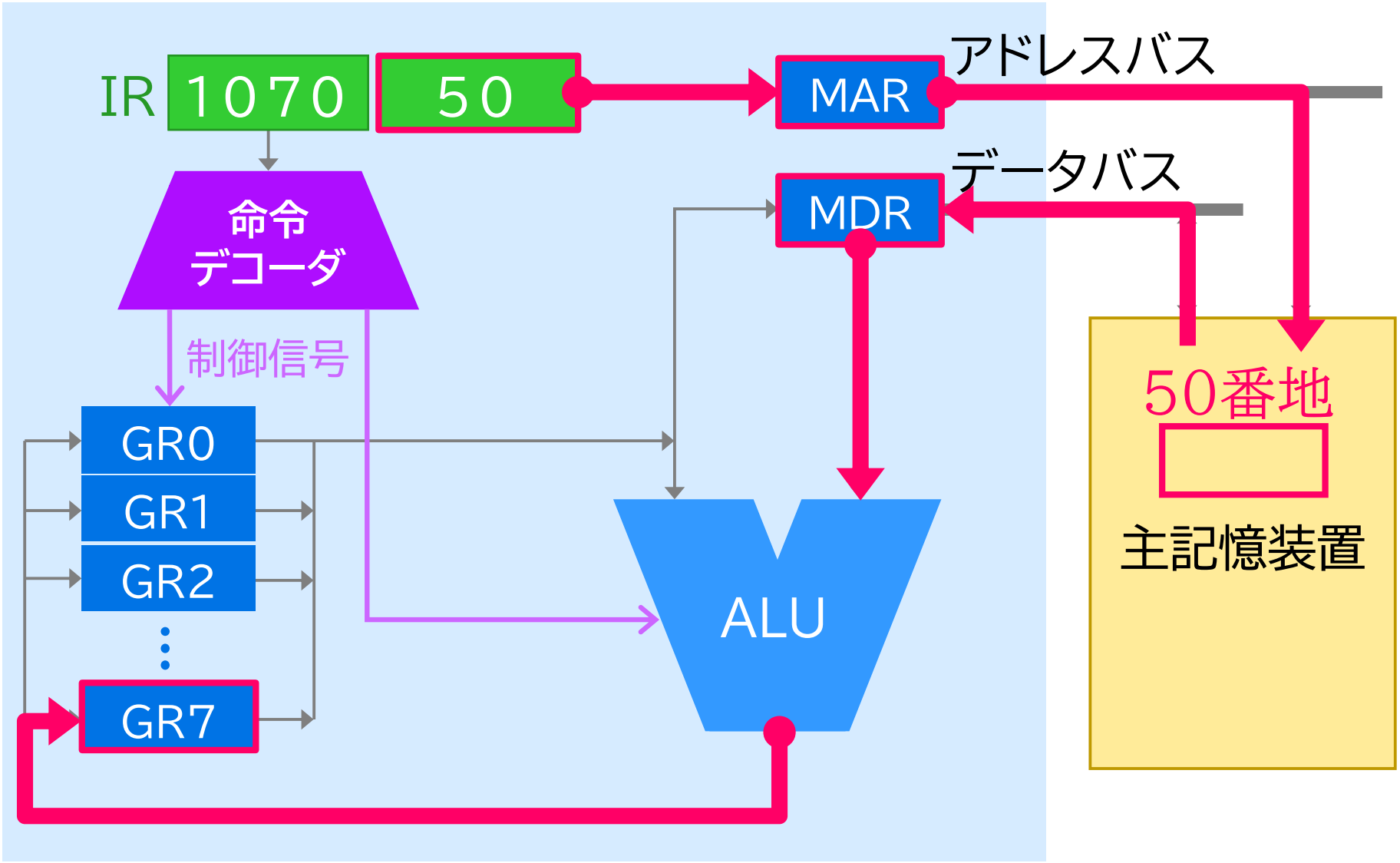
CPUの概略図



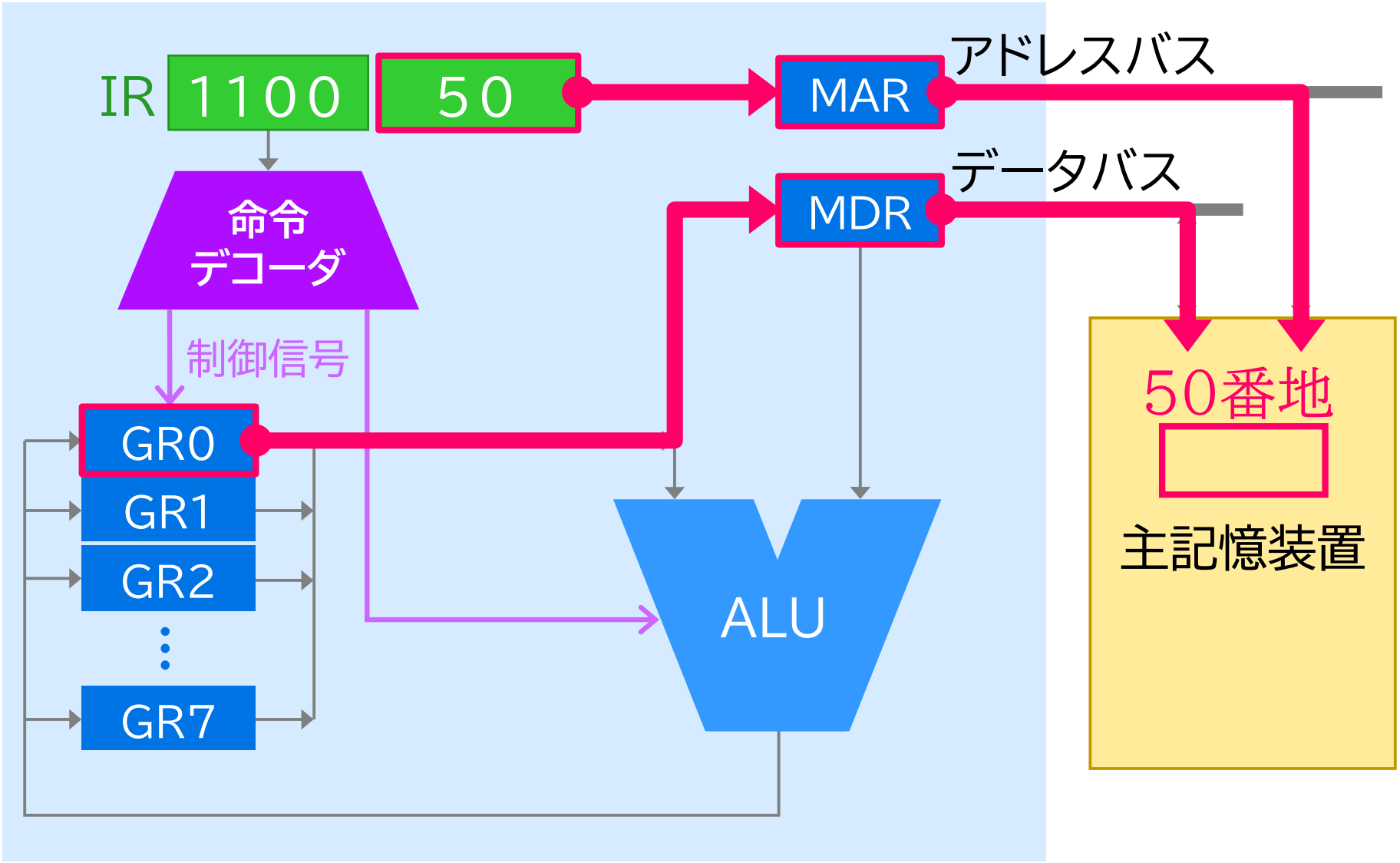
LD GR0,GR2 の実行



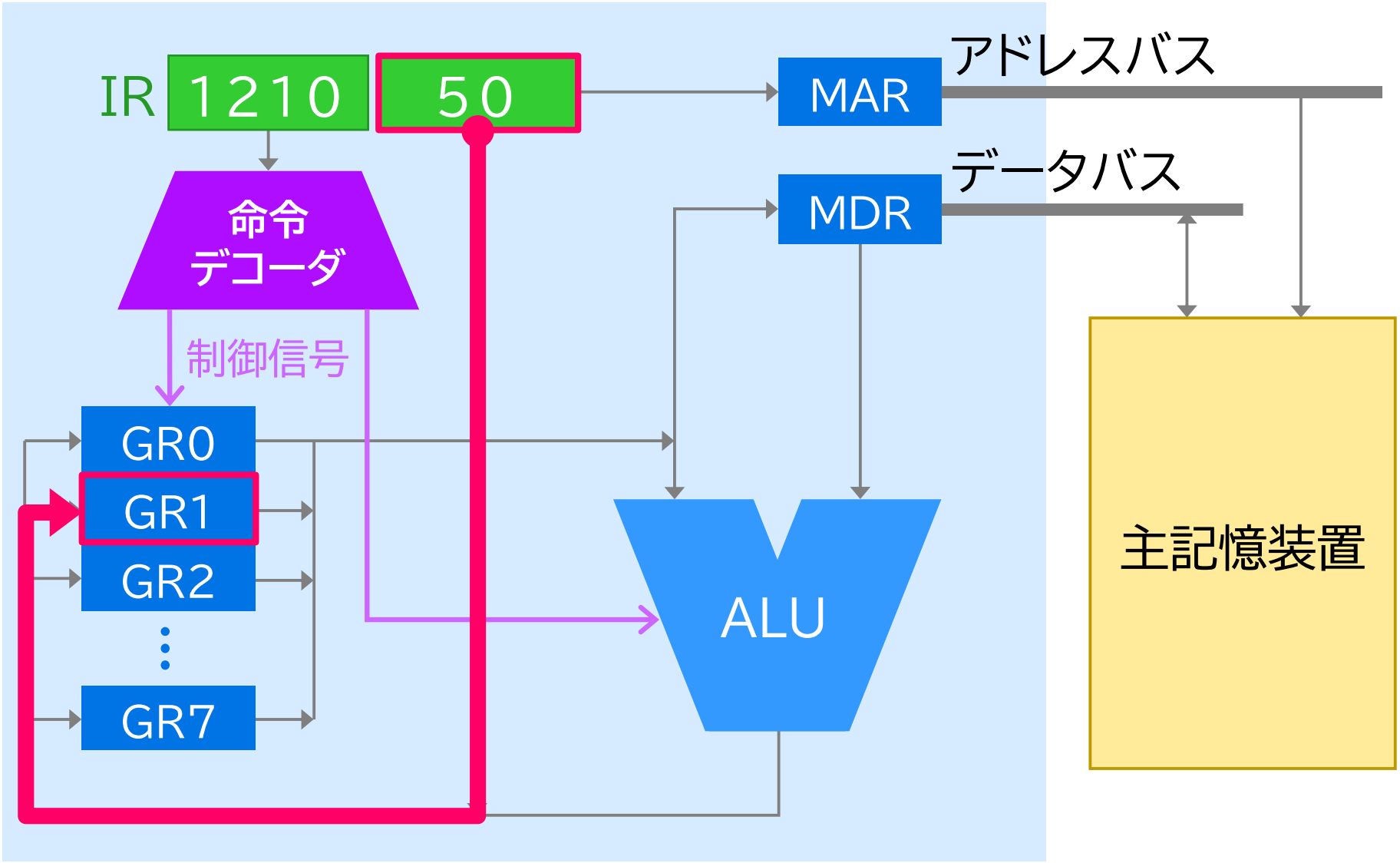
LD GR7,50 の実行


IR
1070
50
MAR
アドレスバス
データバス
MDR
命令
デコーダ
制御信号
GR0
GR1
GR2
GR7
ALU
50番地
主記憶装置

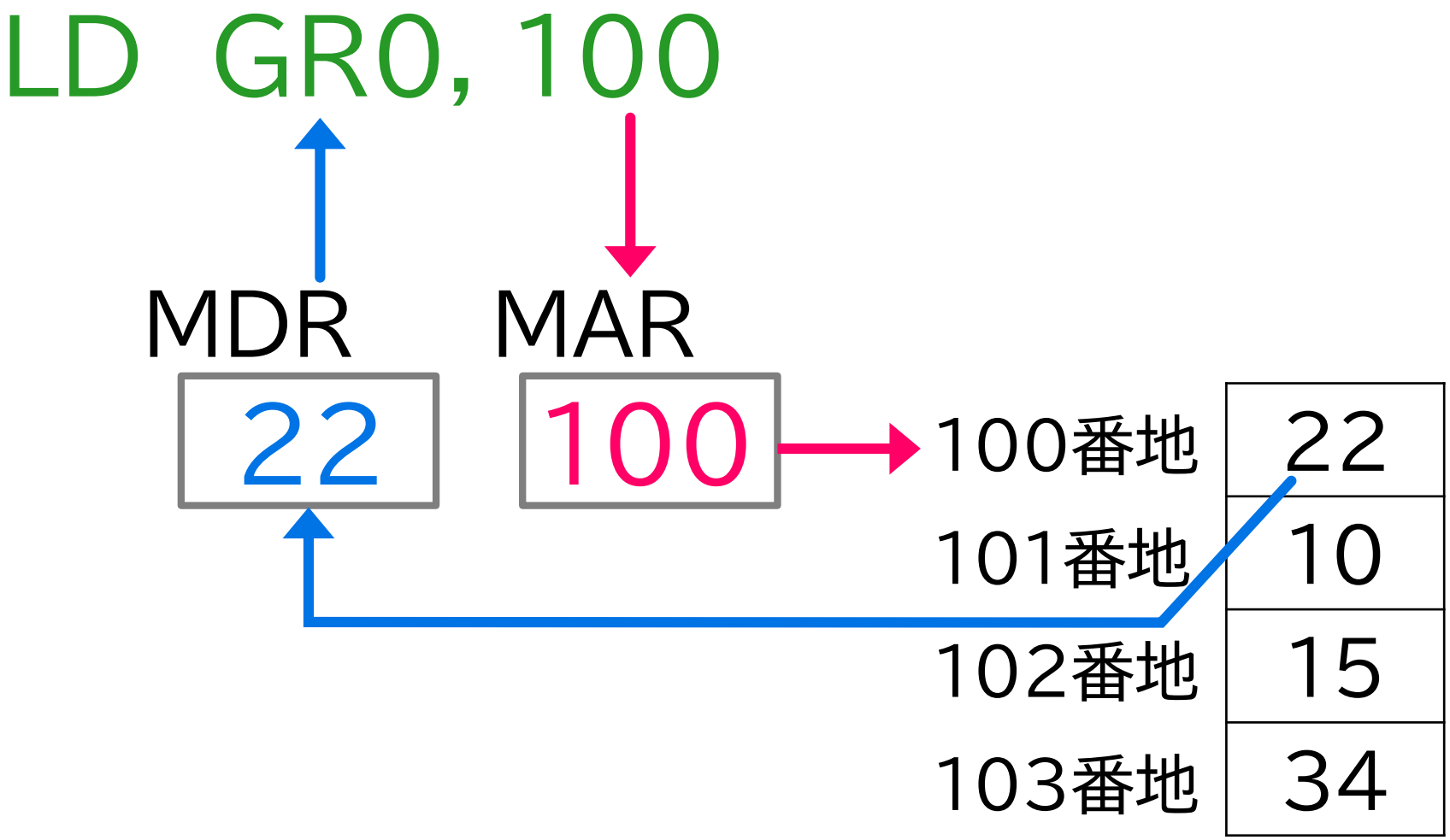
ST GR0,50 の実行


IR
1100
50
MAR
アドレスバス
データバス
MDR
命令
デコーダ
制御信号
GR0
GR1
GR2
GR7
ALU
50番地
主記憶装置

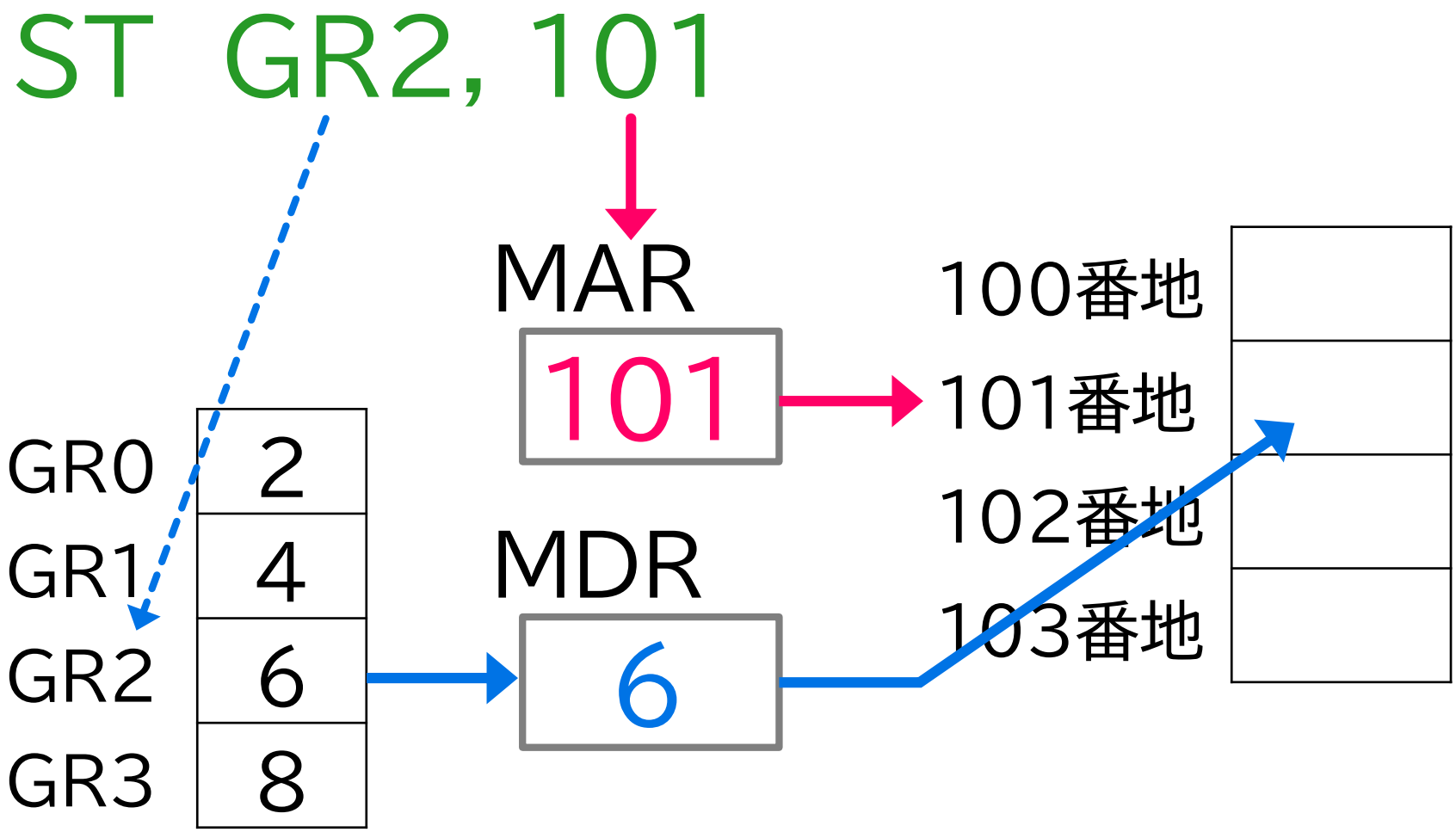
LAD GR1,50 の実行



MARとMDRの値(LDの場合)



MARとMDRの値（STの場合）



ADDA（算術加算）

重要

2つの値の加算結果をレジスタに書き込む。

ADDA　r1，r2

レジスタr1 ← r1の値 ＋ r2の値

ADDA　r1，adr

レジスタr1 ← r1の値 ＋ adr番地の値

SUBA（算術減算）

重要

2つの値の減算結果をレジスタに書き込む。

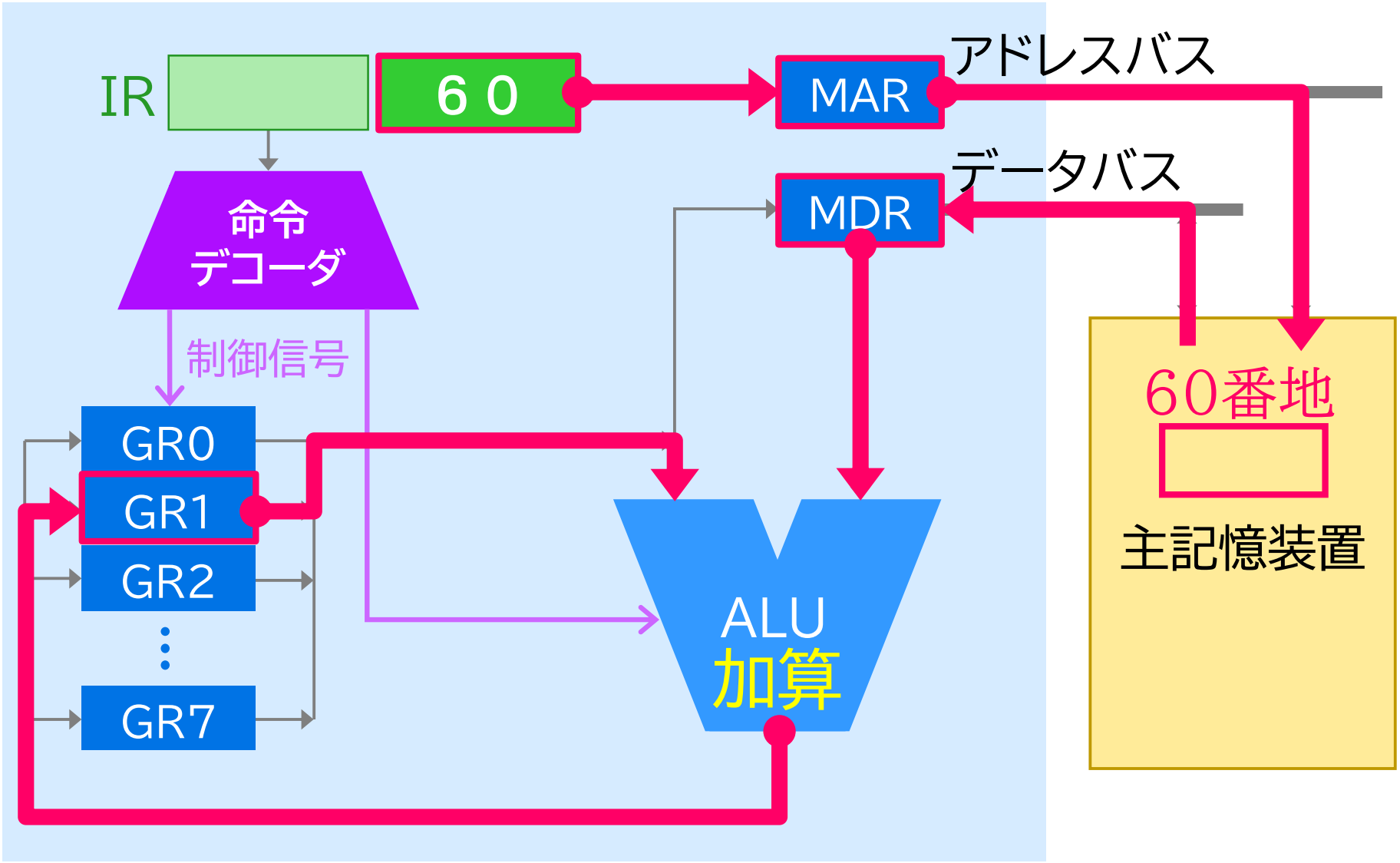
SUBA　r1，r2

レジスタr1 ← r1の値 － r2の値

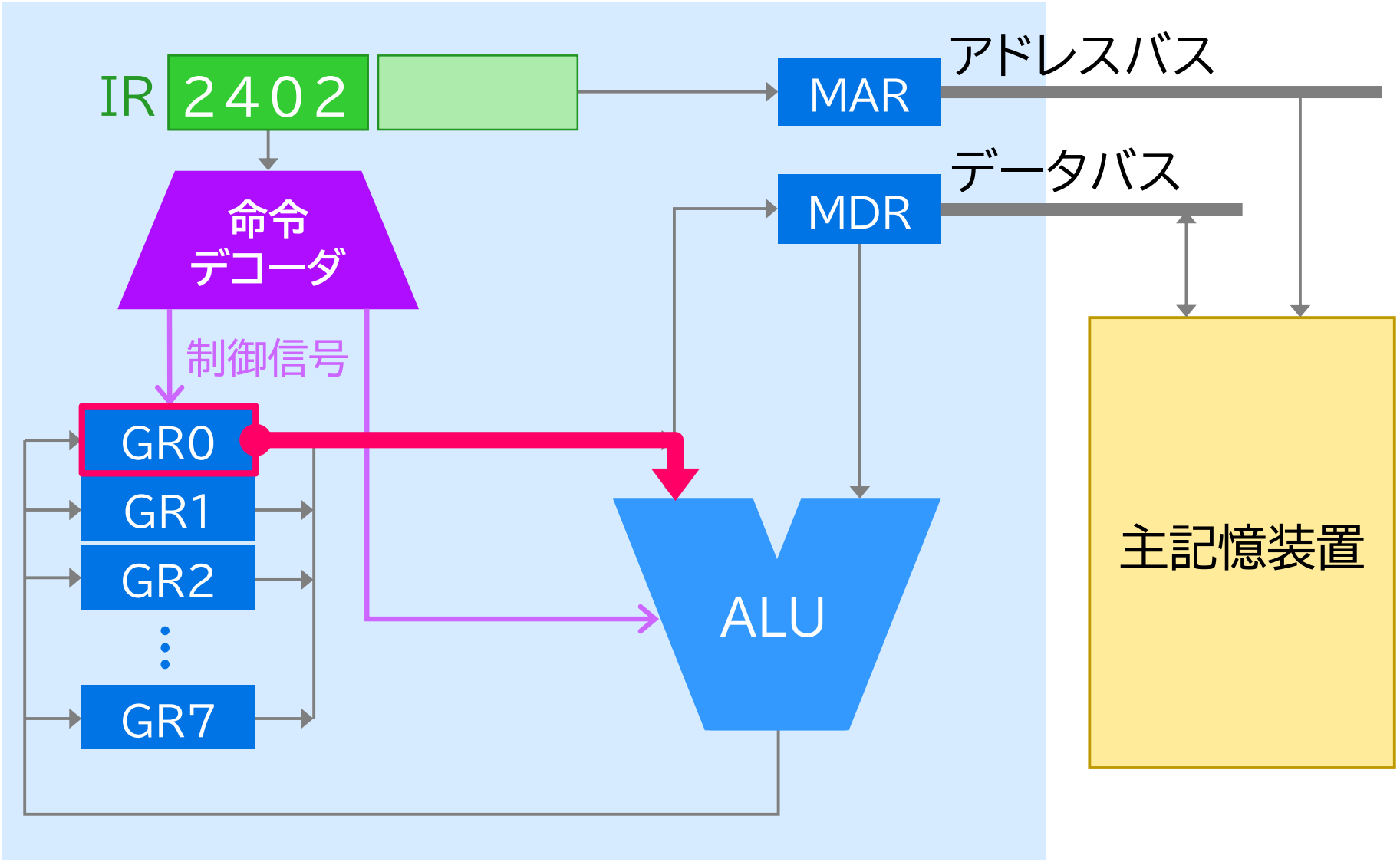
SUBA　r1，adr

レジスタr1 ← r1の値 － adr番地の値

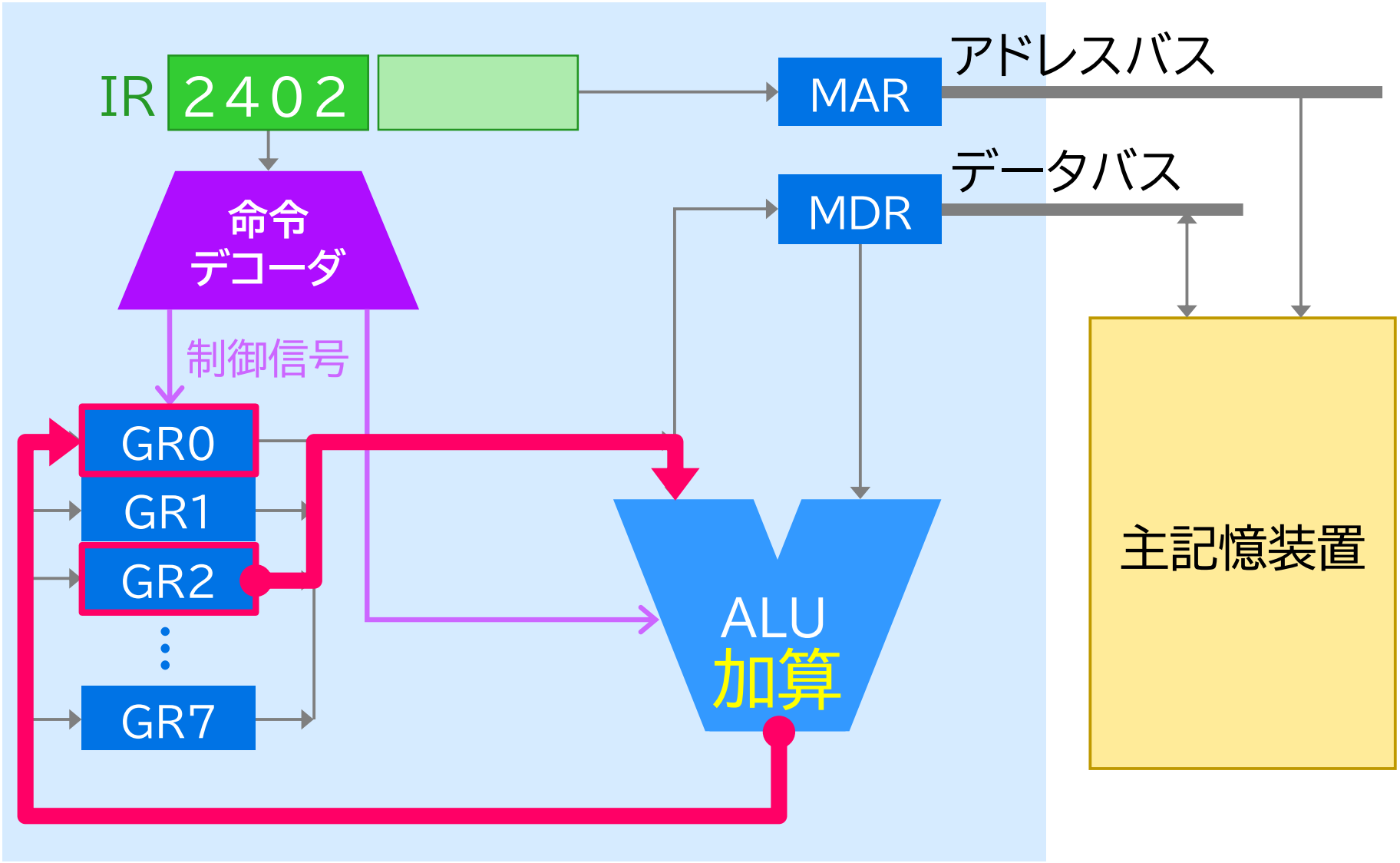
ADDA GR1,60 の実行



ADDA GR0,GR2 の実行



ADDA GR0,GR2 の実行



x=a+bを計算するプログラム

主記憶装置

番地	変数
100	a
101	b
102	x

CPU

GR0	

❶ GR0に変数ａの値を移す。
❷ GR0に変数ｂの値を加算。
❸ GR0の値を変数ｘに移す。

アセンブリ言語プログラム

```
❶ LD    GR0, 100
❷ ADDA  GR0, 101
❸ ST    GR0, 102
```

x=a+b−cを計算するプログラム

主記憶装置

番地	変数
100	a
101	b
102	c
103	x

CPU

GR0 [ ]

❶ GR0に変数ａの値を移す。
❷ GR0 に変数ｂの値を加算。
❸ GR0 から変数ｃの値を減算。
❹ GR0 の値を変数ｘに移す。

アセンブリ言語プログラム

```
❶ LD    GR0, 100
❷ ADDA  GR0, 101
❸ SUBA  GR0, 102
❹ ST    GR0, 103
```

実効アドレスと指標レジスタ

重要

LD　r1，adr

レジスタr1 ← adr番地の値

実効アドレス

処理対象のアドレス

LD　r1，adr，x

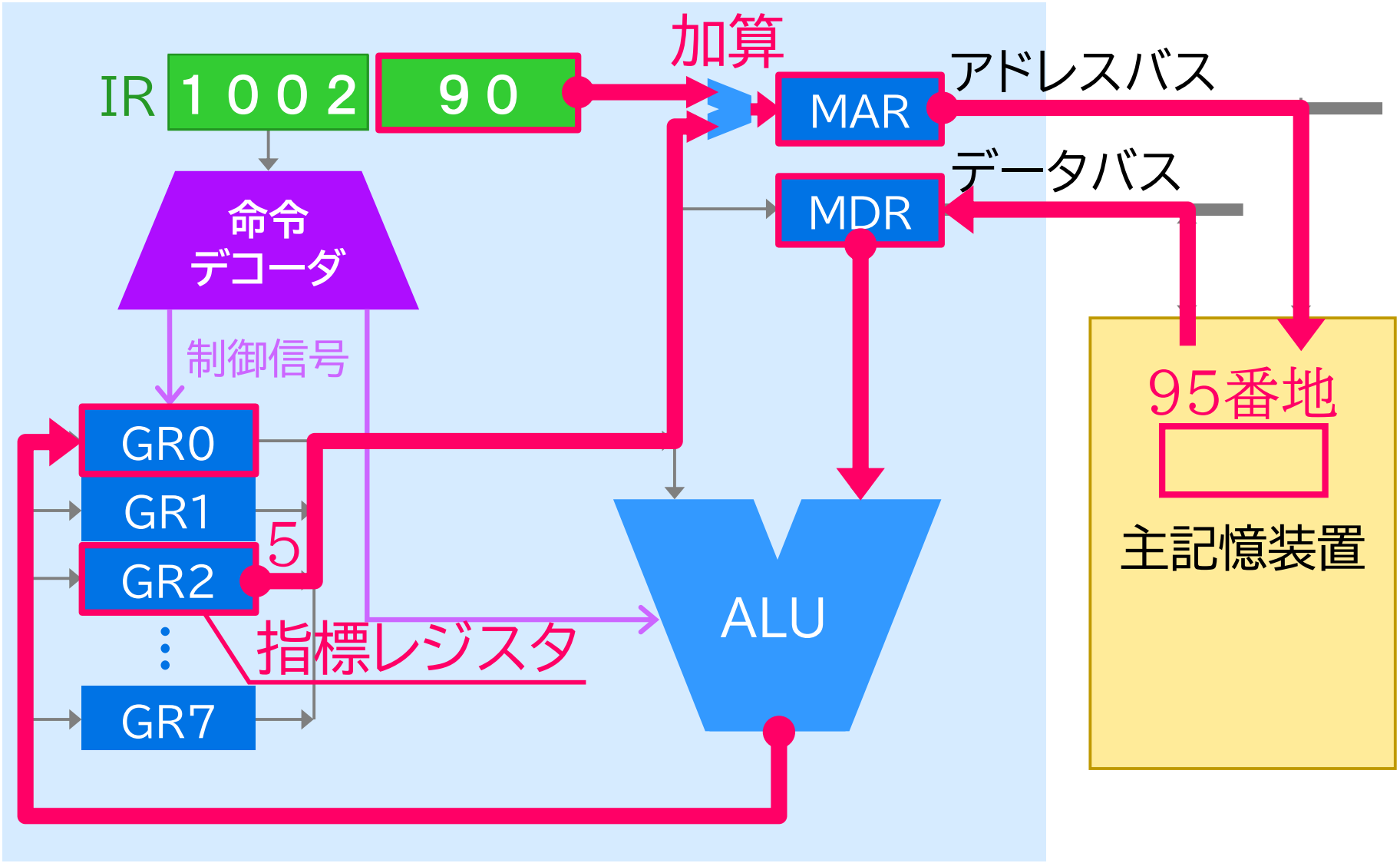
指標レジスタ

GR1～GR7のどれか

レジスタr1 ← (adr＋レジスタx)番地の値

実効アドレス

LD GR0,90,GR2 の実行



指標レジスタを用いた命令記述 重要

LD r1, adr, x

レジスタr1 ← (adr + レジスタx)番地の値

LAD r1, adr, x

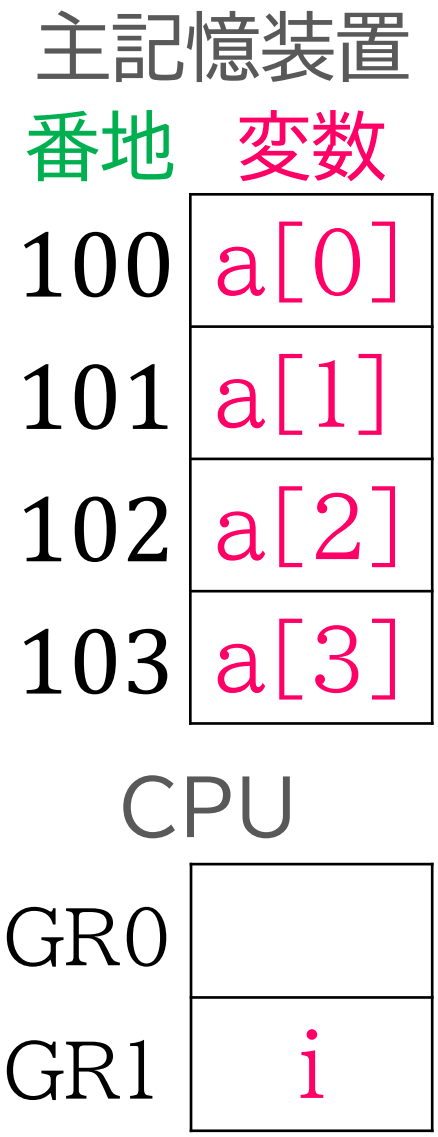
レジスタr1 ← (adr + レジスタx)

ST r1, adr, x

(adr + レジスタx)番地 ← レジスタr1の値

配列の読み書きへの応用

参考



配列ａの先頭番地が100であるとき、a[i]が記憶されている番地は、(100+i)番地になる。

※ 配列要素の大きさは1byteとする。

iの値がGR1に格納されているとするとき、a[i]の値をGR0へ読み出す命令は、

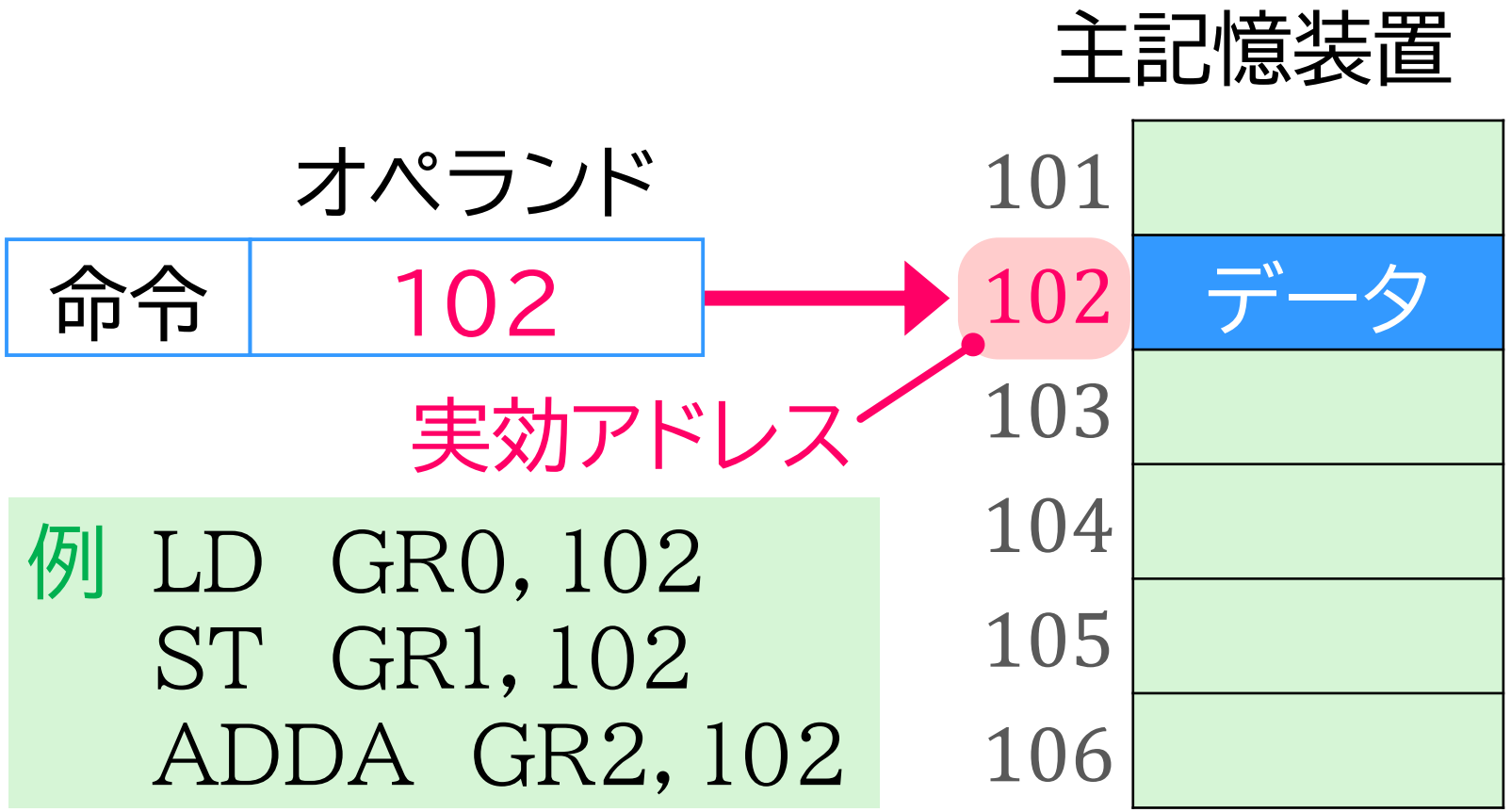
```
LD  GR0, 100, GR1
```

アドレス指定方式

- 直接アドレス指定
- 間接アドレス指定
- 相対アドレス指定
- 指標アドレス指定
- 即値アドレス指定

直接アドレス指定

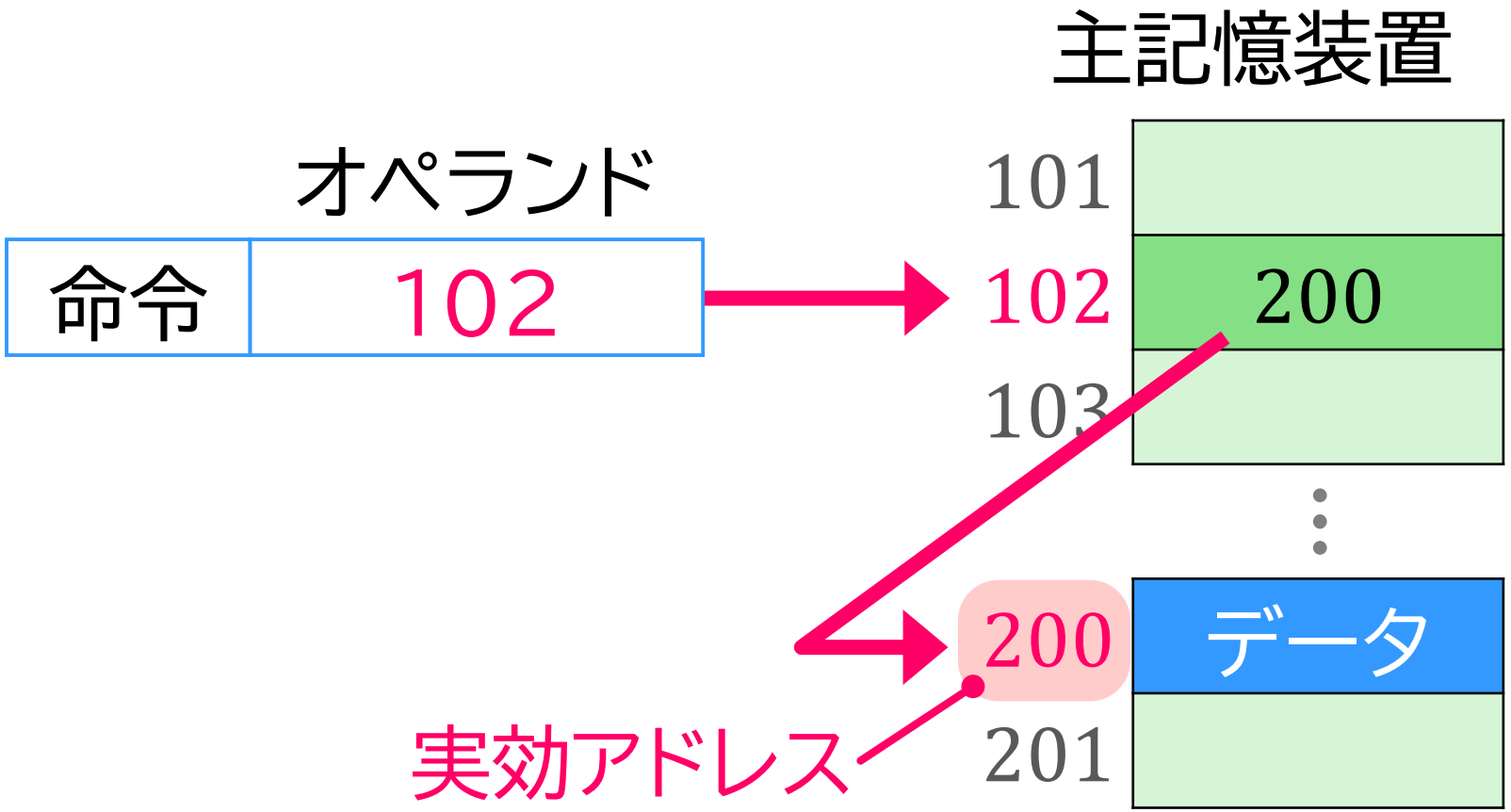
オペランドのアドレス部の値を実効アドレスとする。



例
```
LD   GR0, 102
ST   GR1, 102
ADDA GR2, 102
```

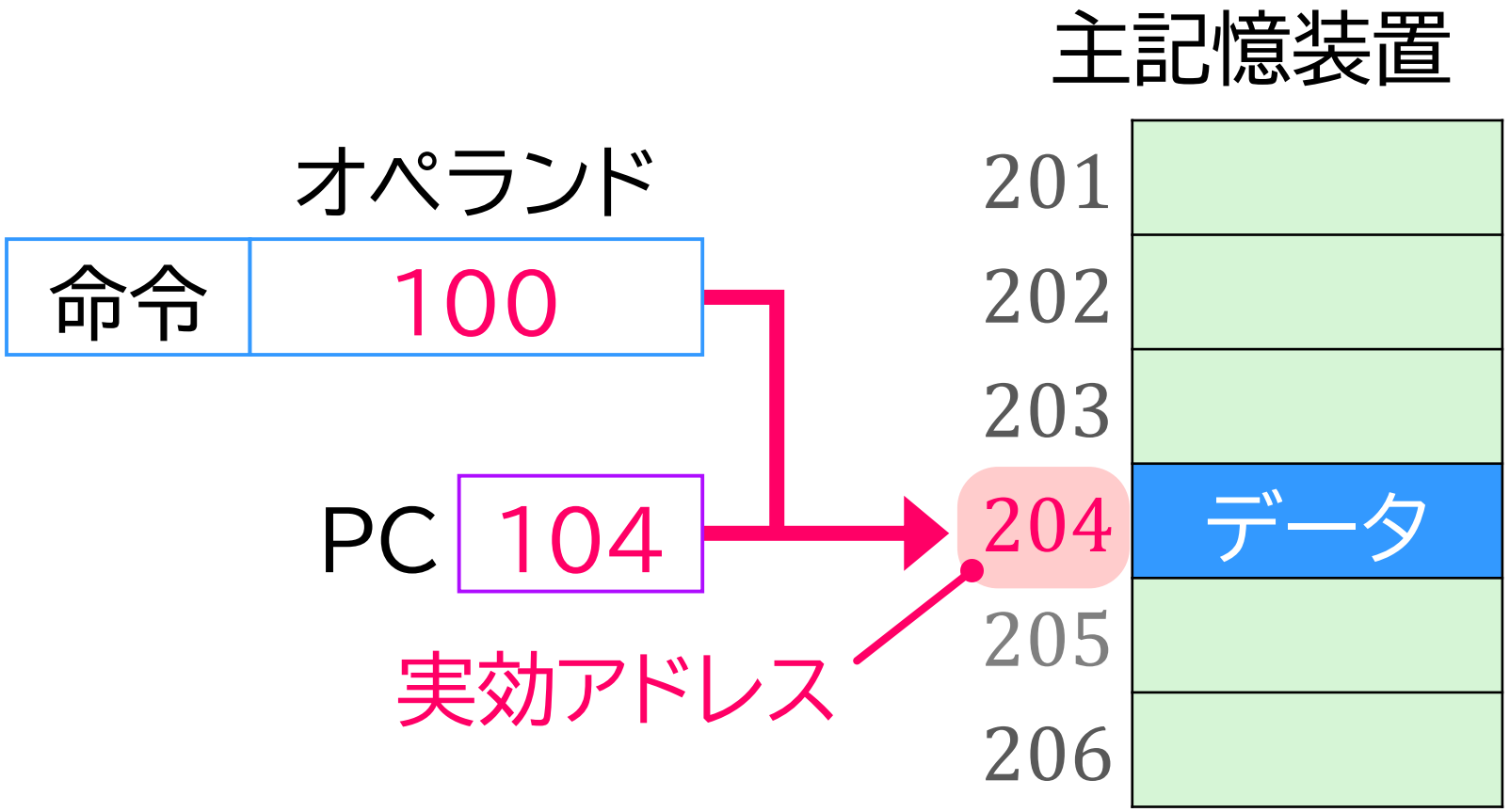
間接アドレス指定

オペランドのアドレス部に、実効アドレスを格納しているアドレスを格納する。



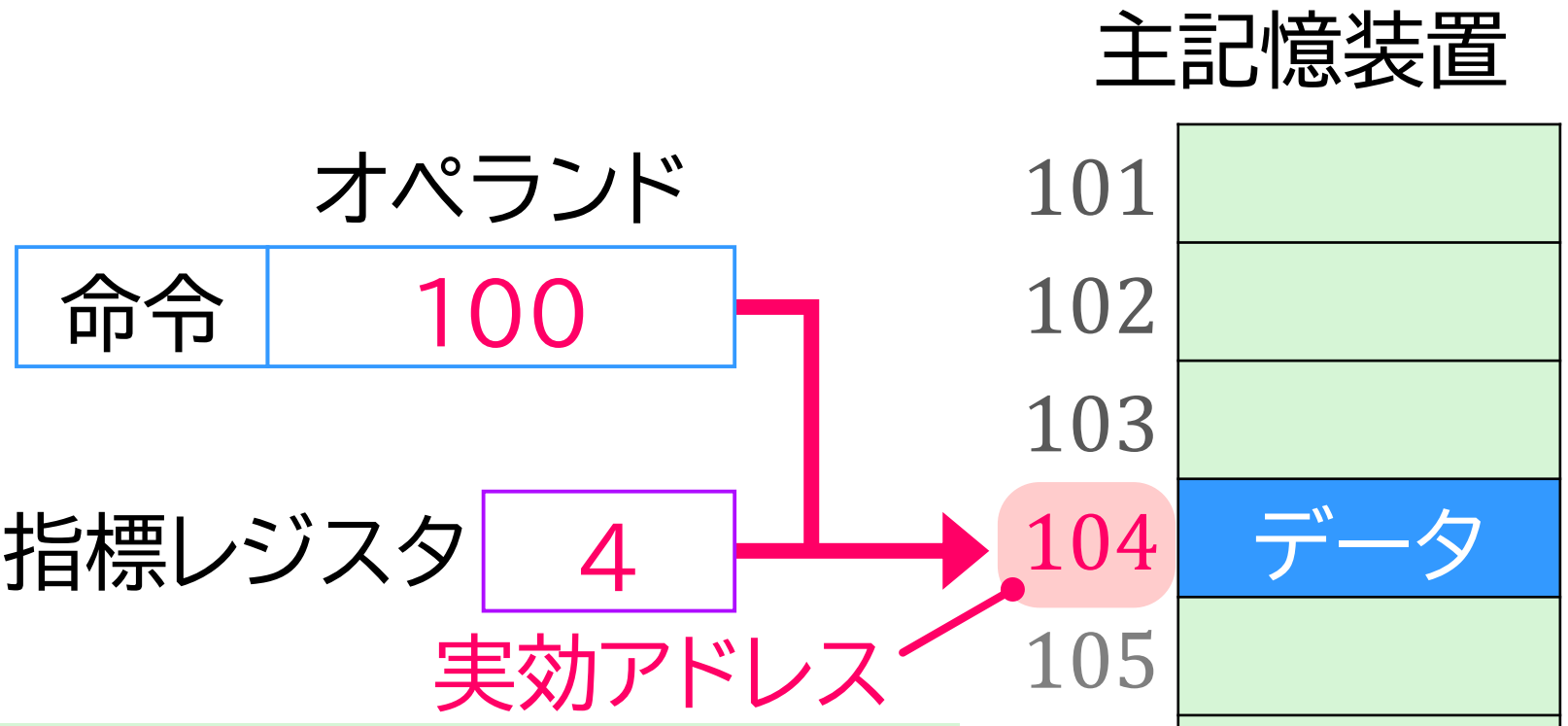
相対アドレス指定

オペランドのアドレス部の値に、PCの値を加算したものを実効アドレスとする。



指標アドレス指定

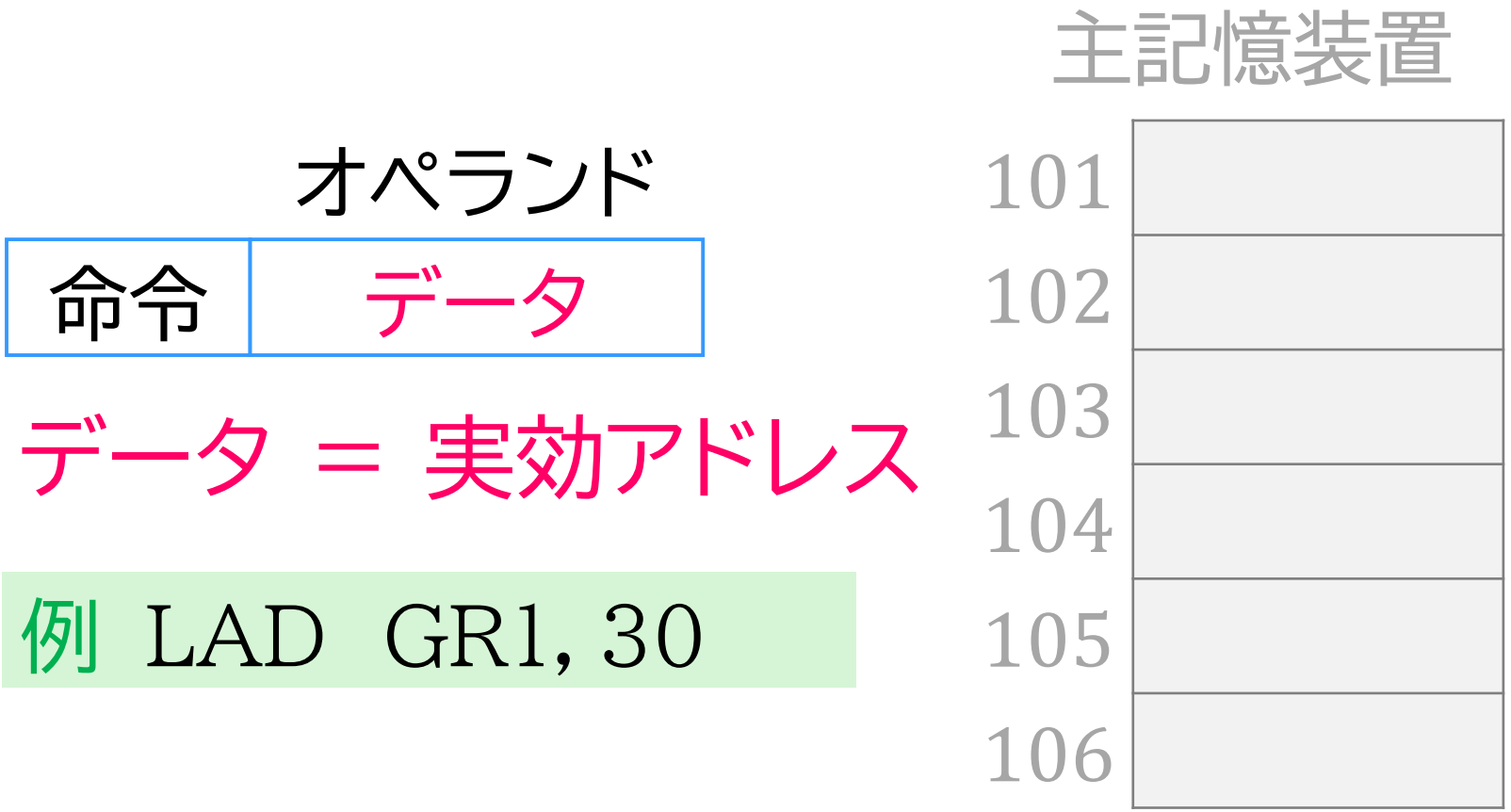
オペランドのアドレス部の値に、指標レジスタの値を加算したものを実効アドレスとする。



例　LD　GR0, 100, GR1

即値アドレス指定

オペランドのアドレス部にデータそのものを格納する。



データ ＝ 実効アドレス

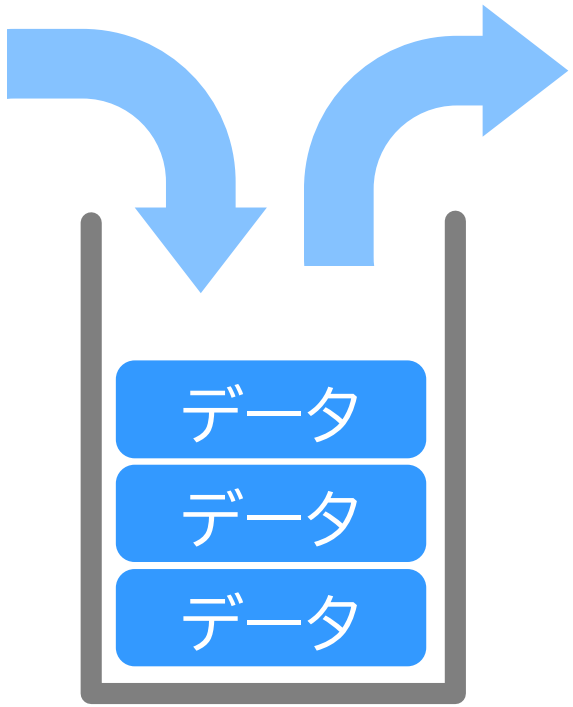
例 LAD GR1, 30

スタック

データを一時的に蓄えておくための主記憶装置上の記憶場所。
後入れ先出し方式(LIFO)

Push
データを入れる

Pop
データを取り出す



PUSH／POP

PUSH　adr

adr(定数)をスタックへ入れる。

PUSH　adr, x

(adr+レジスタx)をスタックへ入れる。

POP　r1

スタックから取り出した値をレジスタr1へ書き込む。

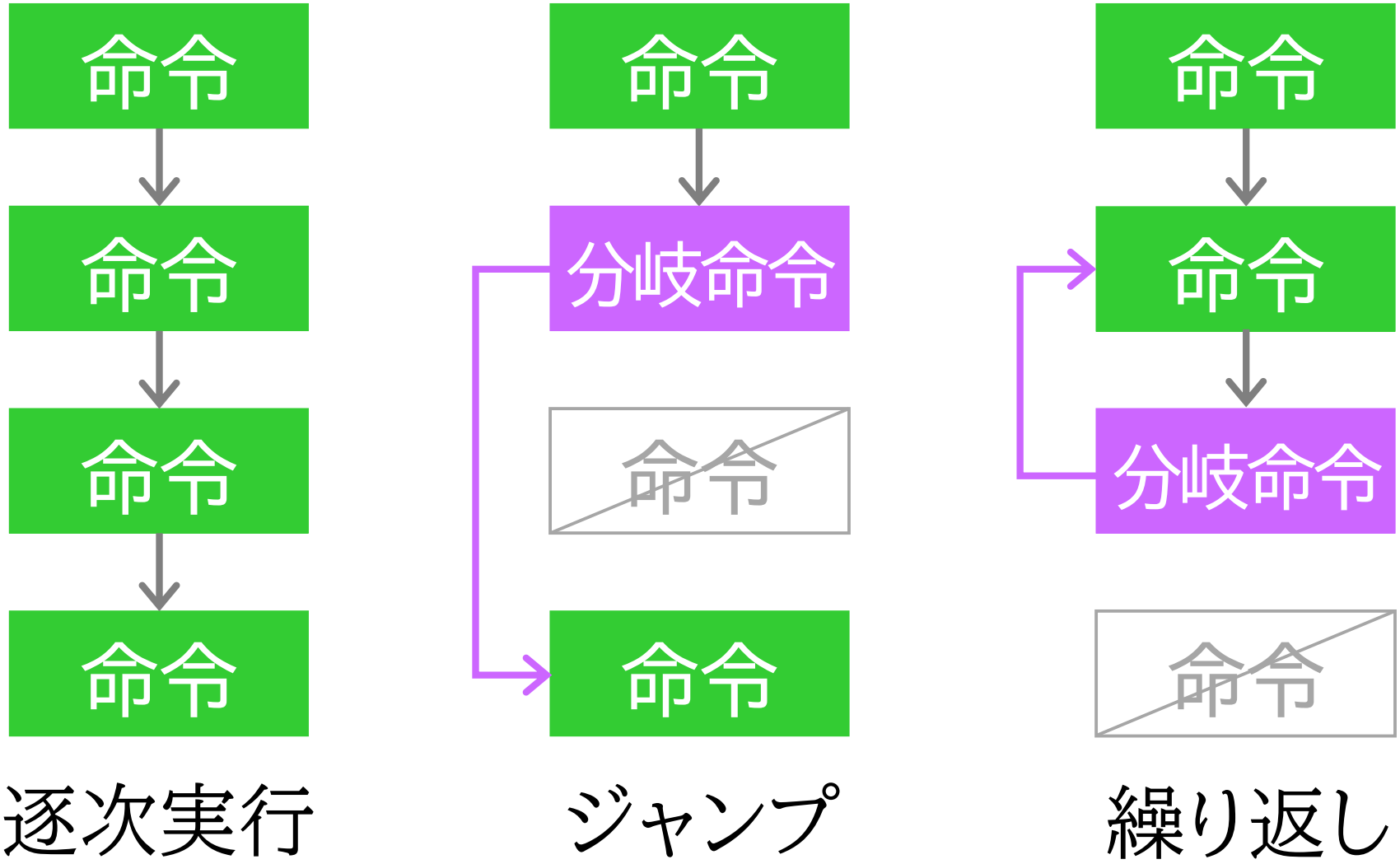
キュー

データを一時的に蓄えておく方式の一つ。
先入れ先出し方式(FIFO)

Enqueue
データを入れる

Dequeue
データを取り出す

プログラムの流れ（逐次実行と分岐）



分岐命令の種類

重要

- 無条件分岐
- 条件分岐
  - 負分岐
  - 正分岐
  - 零分岐
  - 非零分岐

無条件分岐

重要

命令を読み出すアドレスを強制的に変更して、プログラムの流れを変える。

JUMP　adr

PC(プログラムカウンタ) ← adr

次に実行する命令は、adr番地から読み出される。

ラベル

アドレスの代わりとして使う識別記号。

プログラムの作成時にアドレスの値を考える必要がなくなる。プログラムが読みやすくなる。

100	LD GR0, 200
102	JUMP 110
110	ST GR0, 201
200	DC 5
201	DS 1

	LD GR0, *A*
	JUMP *X*
X	ST GR0, *B*
A	DC 5
B	DS 1

フラグレジスタ(FR)

重要

演算の結果の値や、データ転送した値の状態を示すレジスタ。
3つのフラグ(各1bit)から構成される。

FR

フラグ

重要

SF(サインフラグ)

値が
- 負の数のとき → 1
- 0または正の数のとき → 0

ZF(ゼロフラグ)

値が
- 0のとき → 1
- 0以外のとき → 0

OF(オーバーフローフラグ)

値が
- 有効bitを越えたとき → 1
- 有効bitを越えないとき → 0

フラグの値

重要

	演算結果の値		
	正の数	0	負の数
SF (サインフラグ)	0	0	1
ZF (ゼロフラグ)	0	1	0

算術比較命令

重要

2つの値の減算結果からFRを設定する。
減算結果の値は残さない。

CPA r1，r2

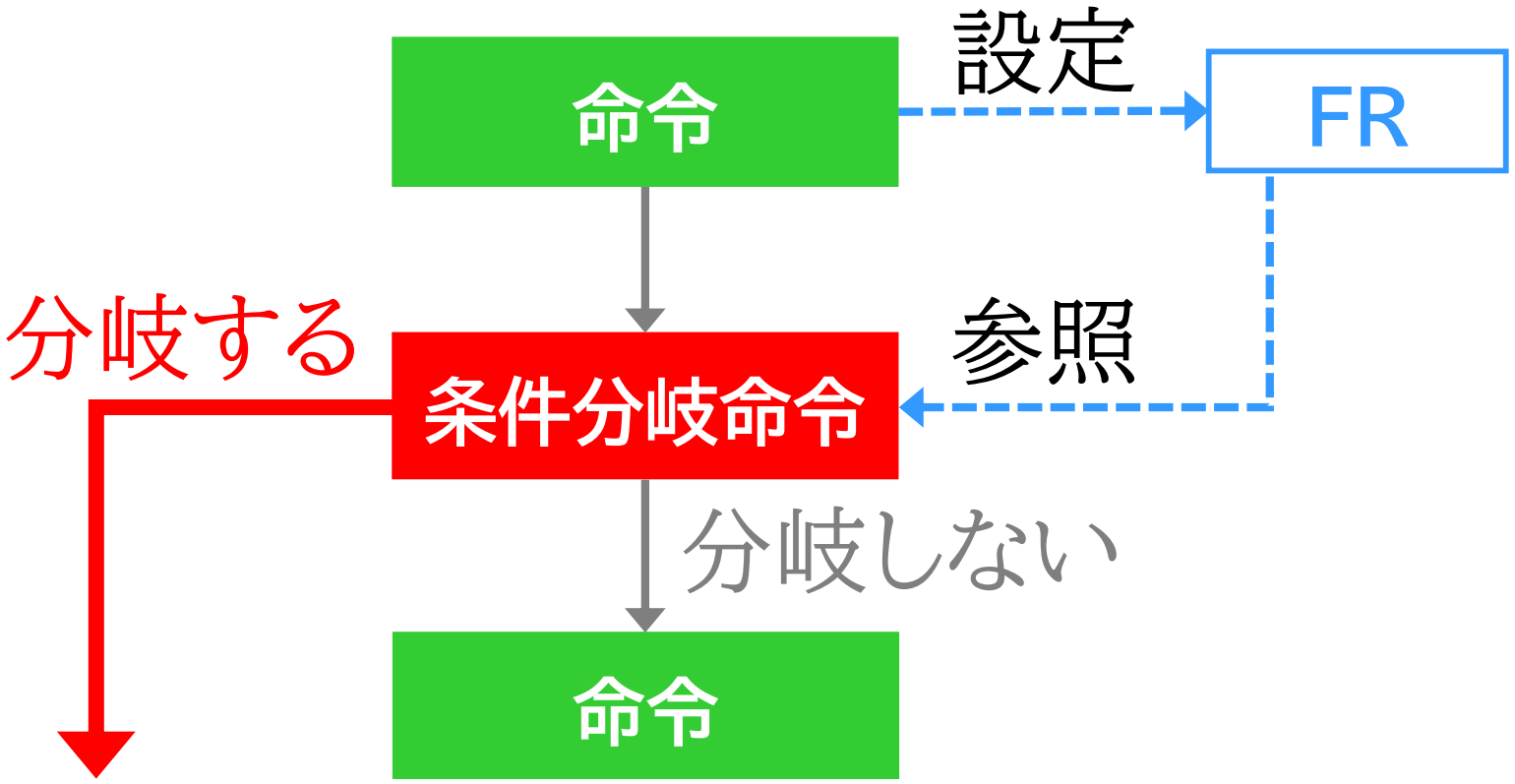
r1－r2の結果からFRを設定する。

CPA r1，adr

r1－adr番地の値の結果からFRを設定する。

条件分岐命令

FRの値を参照して、分岐するか、しないかを決定する。



条件分岐命令

重要

JMI adr　負分岐

負(SF＝1)のとき、PC ← adr

JPL adr　正分岐

正(SF＝0, ZF＝0)のとき、PC ← adr

JZE adr　零分岐

0(ZF＝1)のとき、PC ← adr

JNZ adr　非零分岐

0でない(ZF＝0)のとき、PC ← adr

算術比較と条件分岐の使用

GR0＜GR1 のとき、X番地に分岐

↓

GR0－GR1＜0 のとき、X番地に分岐

↓

GR0－GR1 を計算
計算結果が負の値のとき、X番地に分岐

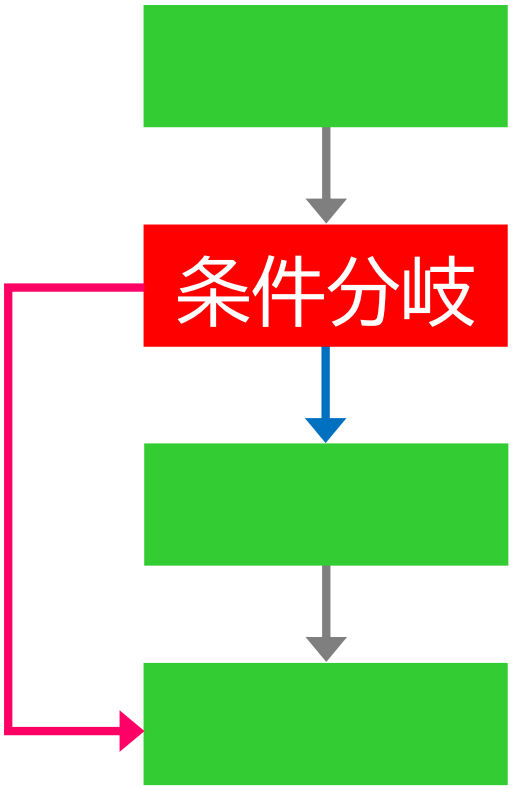
↓

```
CPA  GR0, GR1
JMI  X
```

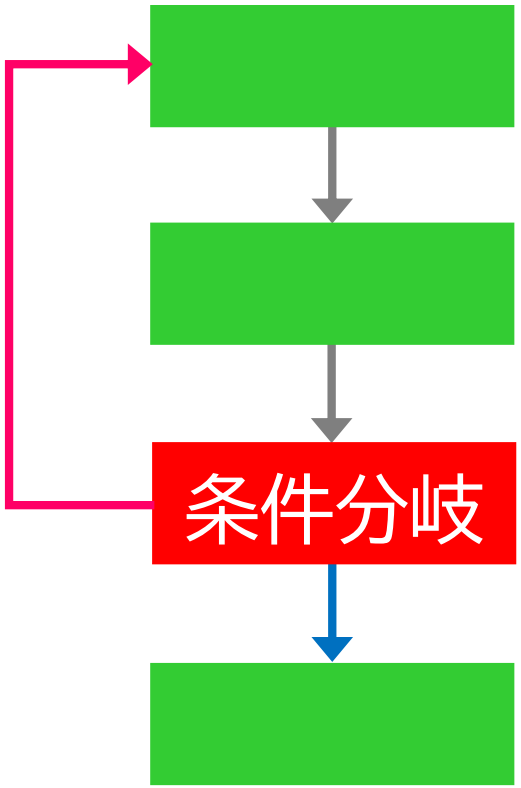
条件分岐の使用

参考

if文型



do-while文型

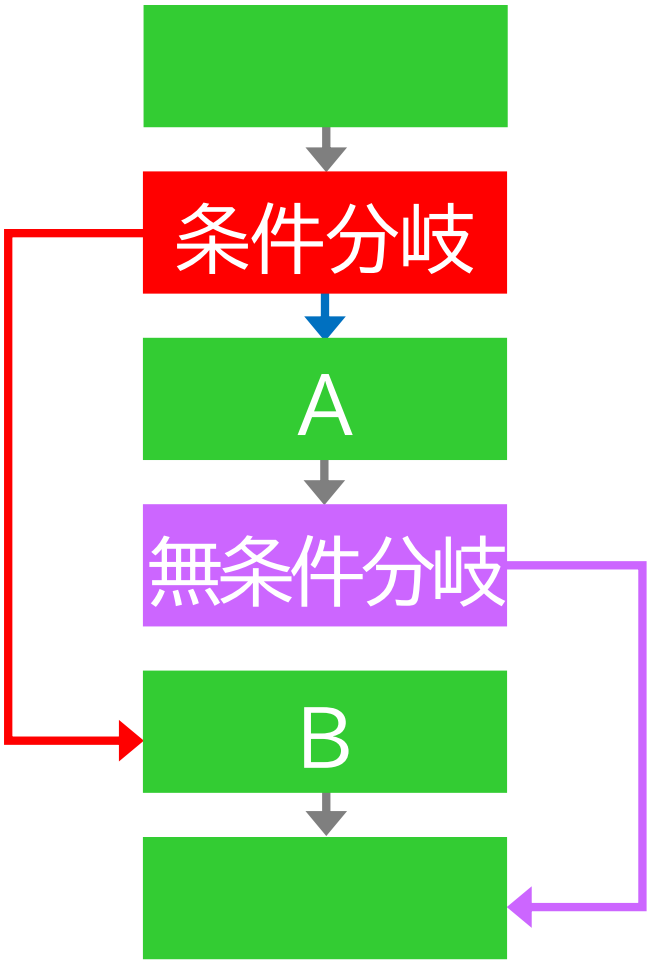


if文型は、C言語などとは異なり、条件が成立するときにジャンプすることに注意。

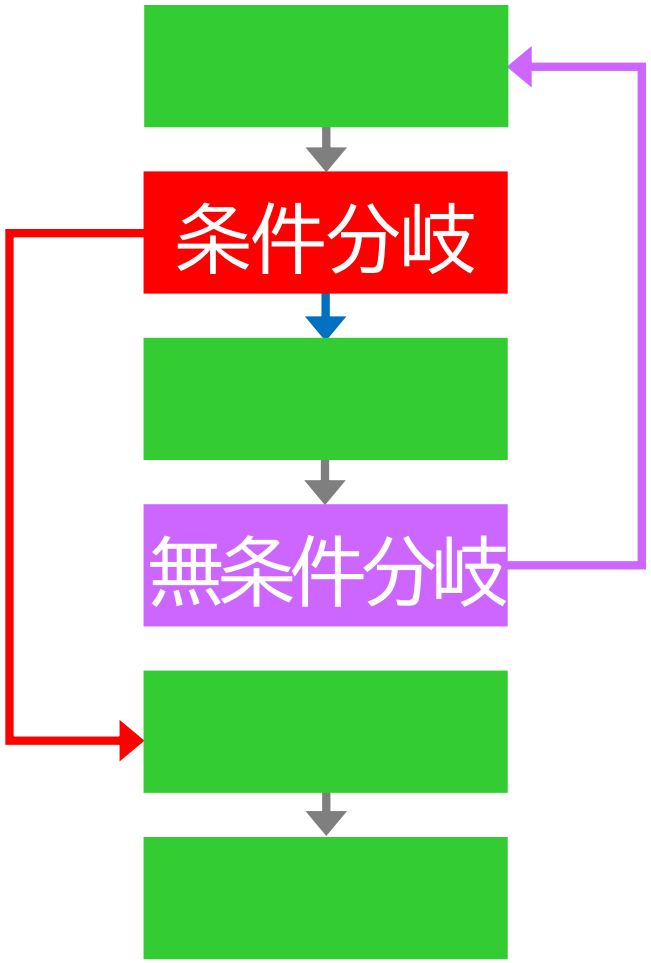
条件分岐と無条件分岐の使用

参考

if-else文型



while文型



CPUの性能評価

重要

CPUの処理能力を表す指標

- 平均命令実行時間
- MIPS
- FLOPS

平均命令実行時間

重要

1つの命令の実行に要する時間の平均値。

$$\text{平均命令実行時間} = \frac{\text{CPUの処理時間}}{\text{実行命令数}}$$

20万個の命令の実行に、0.5秒かかったとするとき、

$$\text{平均命令実行時間} = \frac{5 \times 10^{-1}}{2 \times 10^{5}} = 2.5 \times 10^{-6} \text{ 秒}$$

$$= 2.5 \text{ マイクロ秒}$$

MIPS

重要

1秒間に実行できる命令数(百万単位)。

$$\mathrm{MIPS} = \frac{\text{実行命令数}}{\text{CPUの処理時間}} \times 10^{-6}$$

$$= \frac{1}{\text{平均命令実行時間} \times 10^{6}}$$

平均命令実行時間が2.5マイクロ秒のとき、

$$\mathrm{MIPS} = \frac{1}{2.5 \times 10^{-6} \times 10^{6}} = 0.4\ \mathrm{MIPS}$$

FLOPS

重要

1秒間に実行できる浮動小数点数演算の数。

コンピュータの数値計算能力を表す指標として使われる。

パソコン用CPU
100～500 GFLOPS

GPU(グラフィックス プロセッサ)
1～100 TFLOPS

P	ペタ	10^{15}	千兆
T	テラ	10^{12}	一兆
G	ギガ	10^{9}	十億
M	メガ	10^{6}	百万
K	キロ	10^{3}	千

CPUアーキテクチャ

- CISC　複雑命令セットコンピュータ
  1つの命令で複雑な処理を実行できる。
  組み込まれている命令数は多い。
- RISC　縮小命令セットコンピュータ
  1つの命令は単純な処理だけを行う。
  組み込まれている命令数は少ない。

現在は、両者の長所を合わせ持つCPUが開発されてきている。

CISCとRISCの特徴

	CISC	RISC
利点	小さいプログラムで、複雑な処理が行える。	命令の実行時間が短く、どれも同じ長さである。 CPUの回路構造が簡単になる。
欠点	命令の実行時間が長く、命令ごとに時間が違う。 CPUの回路構造が複雑になる。	複雑な処理をしたいとき、プログラムが大きくなる。
用途	パソコン用CPUなど	組み込み機器用小型マイコンなど

CPUの高速化法

基本的に、CPUの動作クロック周波数に比例して、CPUの処理速度は速くなる。

他の高速化技術

- パイプライン処理
- マルチプロセッサ, マルチコア
- SIMD演算

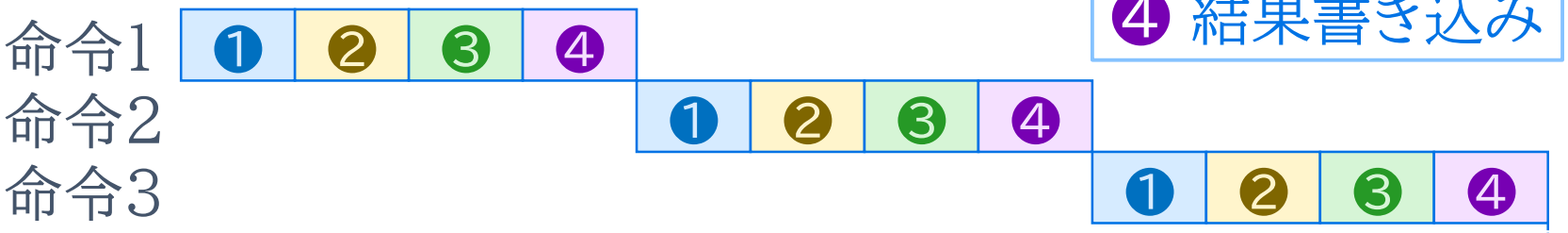
パイプライン処理

重要

複数の命令を並行して実行することで、処理速度を上げる。

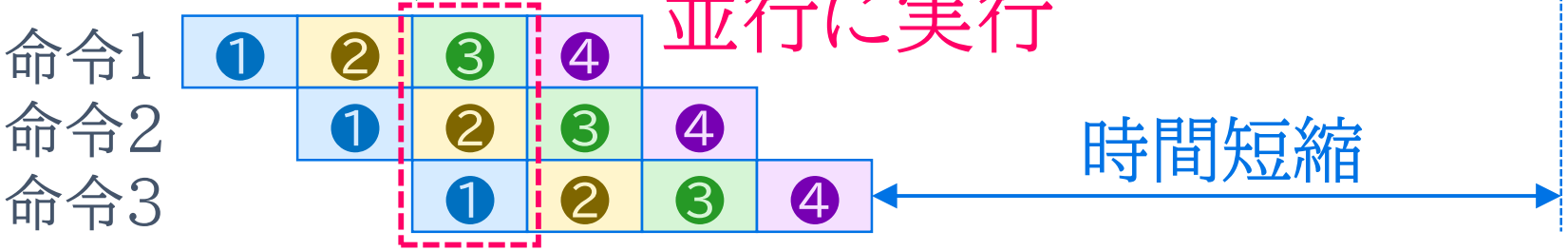
❶ 命令読み出し
❷ 命令解読
❸ 命令実行
❹ 結果書き込み

逐次処理



パイプライン処理

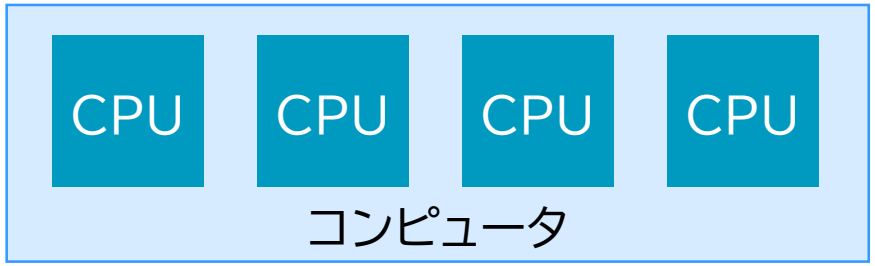
命令サイクルの過程を並行に実行



マルチプロセッサ／マルチコア

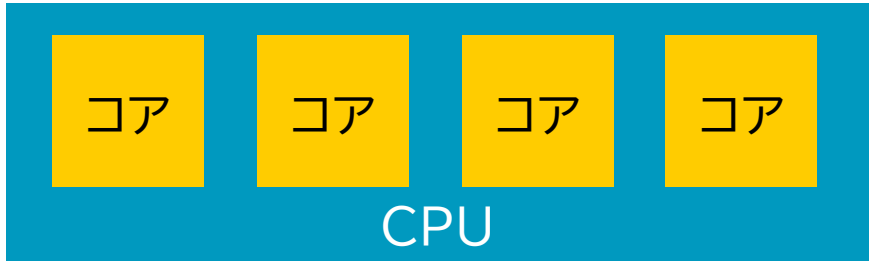
✤ マルチプロセッサ

1台のコンピュータの中に複数のCPUを搭載する。



✤ マルチコア

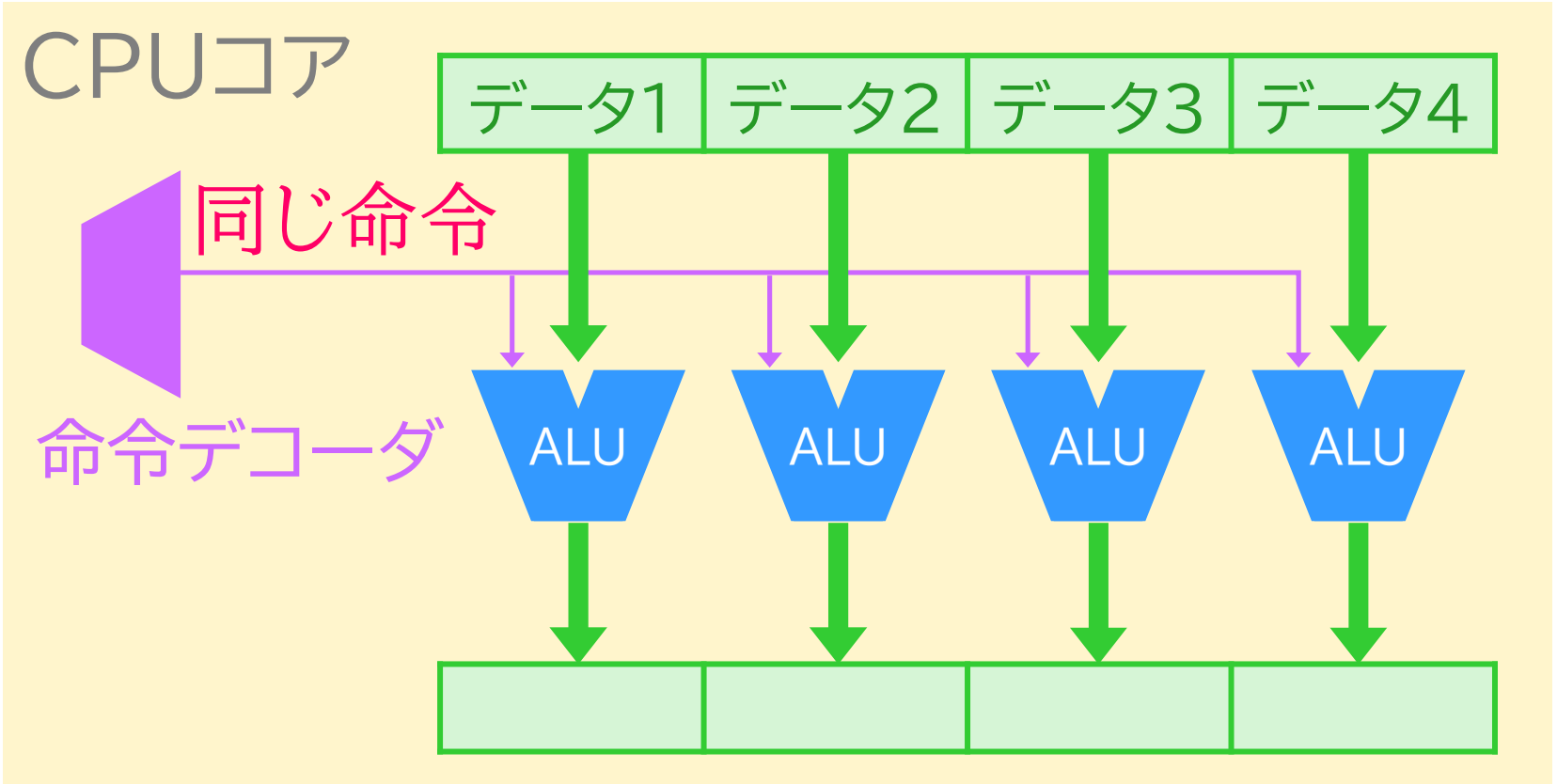
1個のCPUの中に複数のCPUコアを搭載する。



SIMD方式

重要

1つの命令で複数のデータを処理することにより、処理速度を上げる。



スーパーコンピュータ

Summit（アメリカ）

スパコンランキングTOP500で、現在１位。

計算性能 200PFLOPS（2×10^{17} FLOPS）

9216 CPUs (IBM Power9, 22 Cores)

27648 GPUs (NVIDIA Tesla V100)

その他のスーパーコンピュータ

京（日本） 10PFLOPS（2019年に運用終了）

神威・太湖之光（中国） 93PFLOPS

天河二号（中国） 33PFLOPS

定期試験

日時　2020年2月6日　9時10分

出題範囲

- CPUの構成要素と働き。命令サイクル
- アセンブリ言語の記述。実効アドレス
- フラグと分岐処理
- CPUの高速化、CPUの性能評価
- 中間試験範囲から基礎的問題

昨年度の試験問題は、mylogの授業資料からダウンロードできます。