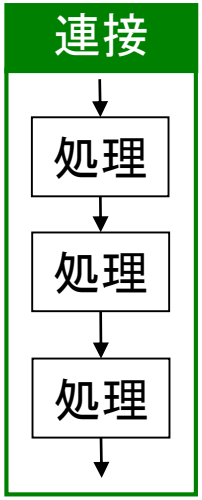
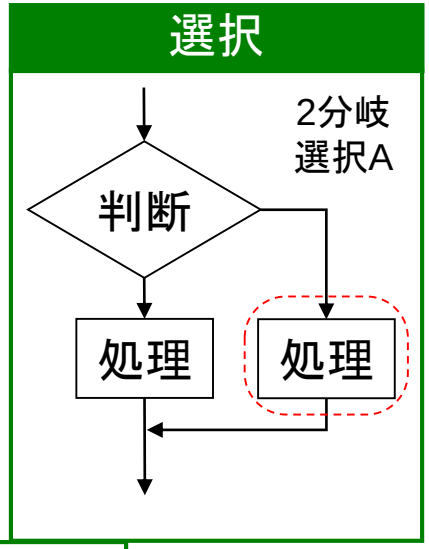
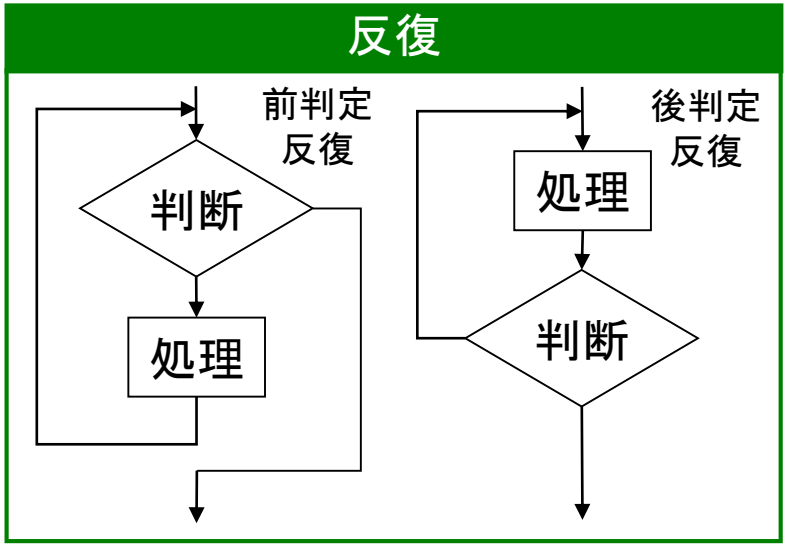
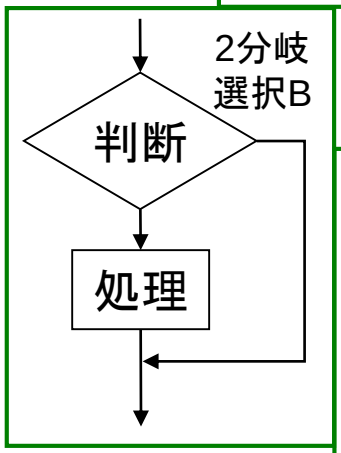
制御構造（制御フロー）：3つの基本制御構造



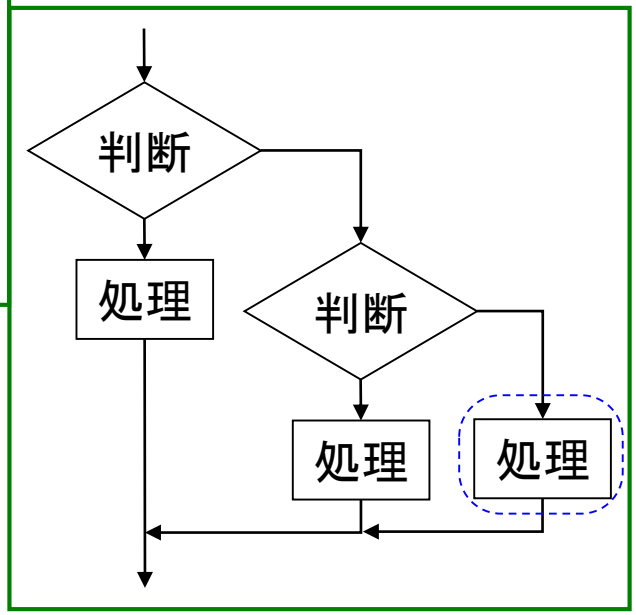






が空の場合

が2分岐選択Aの場合



について同様に・・・

	高水準言語	低水準言語
連接	文の連なり	？
選択	if文など	？
反復	while文、 for文など	？

連接／選択／判断という単位でなく、
それらの構造の要素の単位で考える

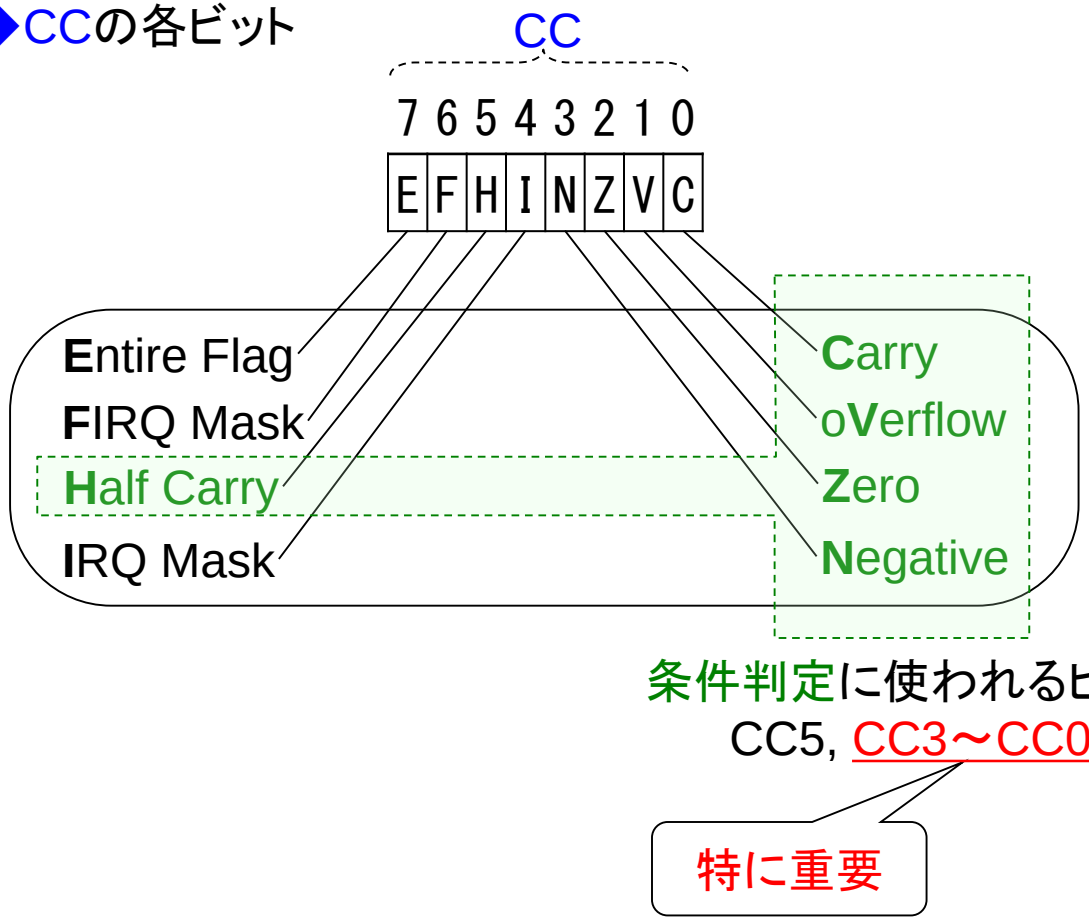
低水準言語での表現は?

制御構造の構成要素①：判断（1/4）

●条件に関する情報は
・どこにあるのか→ CC（Condition Code Reg）にある
・いつ置かれるのか→ CCを変更する命令が実行された時に置かれる

●判断処理自体はどうやって行うのか
→条件判定あり分岐の条件分岐基本命令が行う

◆CCの各ビット



意味→man6809.pdf p.1-4

条件判定に使われるビット
CC5, CC3～CC0

特に重要

「判断＝条件判定」とは、
CC5, CC3～CC0が
所望の値になっているか
どうかをチェックすること。

制御構造の構成要素①：判断（2/4）

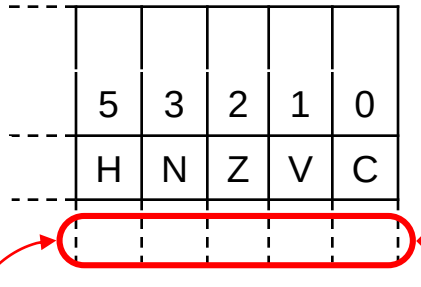
●条件に関する情報は
- どこにあるのか→ CC（Condition Code Reg）にある
- いつ置かれるのか→ CCを変更する命令が実行された時に置かれる

●判断処理自体はどうやって行うのか
→条件判定あり分岐の条件分岐基本命令が行う

◆CCを変更する命令→各命令毎、資料で調べる

- man6809.pdf　p.D-1 ～ p.D-4

各命令が、どのように
CCを変更するのか
を確認



●	変化なし
↕	0または1に変化
0	必ず0になる
1	必ず1になる
7	man6809.pdf p.D-4
8	の表下のNotes:の7～9を
9	それぞれ参照

- 6809_Insn_MainList-AL.pdf
  6809_Insn_MainList-FULL.pdf
- 6809_Insn_Branch-AL.pdf
  6809_Insn_Branch-FULL.pdf
- 6809_Insn_PshPul-AL.pdf
  6809_Insn_PshPul-FULL.pdf

＋

- 6809_Insn_Notation.pdf

- man6809.pdf
  p.A-1 ～ p.A-72

制御構造の構成要素①：判断（3/4）

●条件に関する情報は
・どこにあるのか→ CC（Condition Code Reg）にある
・いつ置かれるのか→ CCを変更する命令が実行された時に置かれる
●判断処理自体はどうやって行うのか
→条件判定あり分岐の条件分岐基本命令が行う

◆条件分岐：条件判定あり／なし

	基本命令機能	基本命令名
条件分岐（19）	**キャリービット**がセットされていなければ分岐	BCC
	キャリービットがセットされていれば分岐	BCS
	等しいならば分岐	BEQ
	より以上ならば分岐（符号付きデータ比較）	BGE
	より大ならば分岐　（符号付きデータ比較）	BGT
	より大ならば分岐　（符号無しデータ比較）	BHI
	より以上ならば分岐（符号無しデータ比較）	BHS
	より以下ならば分岐（符号付きデータ比較）	BLE
	より小ならば分岐　（符号無しデータ比較）	BLO
	より以下ならば分岐（符号無しデータ比較）	BLS
	より小ならば分岐　（符号付きデータ比較）	BLT
	負数ならば分岐	BMI
	等しくないならば分岐	BNE
	正数ならば分岐	BPL
	無条件に分岐	BRA
	分岐しない分岐	BRN
	サブルーチンへ分岐	BSR
	オーバーフロービットがセットされていなければ分岐	BVC
	オーバーフロービットがセットされていれば分岐	BVS

条件判定あり	BCC	**C**がセットされていなければ分岐
	BCS	**C**がセットされていれば分岐
	BEQ	等しいならば分岐
	BGE	より以上ならば分岐（符号付きデータ比較）
	BGT	より大ならば分岐　（符号付きデータ比較）
	BHI	より大ならば分岐　（符号無しデータ比較）
	BHS	より以上ならば分岐（符号無しデータ比較）
	BLE	より以下ならば分岐（符号付きデータ比較）
	BLO	より小ならば分岐　（符号無しデータ比較）
	BLS	より以下ならば分岐（符号無しデータ比較）
	BLT	より小ならば分岐　（符号付きデータ比較）
	BMI	負数ならば分岐
	BNE	等しくないならば分岐
	BPL	正数ならば分岐
	BVC	**V**がセットされていなければ分岐
	BVS	**V**がセットされていれば分岐
条件判定なし	BRA	無条件に分岐
	BRN	分岐しない分岐
	BSR	サブルーチンへ分岐

制御構造の構成要素①：判断（4/4）

＝

条件判定

●条件に関する情報は
・どこにあるのか→ CC（Condition Code Reg）にある
・いつ置かれるのか→ CCを変更する命令が実行された時に置かれる

●判断処理自体はどうやって行うのか
→条件判定あり分岐の条件分岐基本命令が行う

◆条件判定あり分岐の条件分岐基本命令

基本命令名	機能	判定条件
BCC	Cがセットされていなければ分岐	**C=0**
BCS	Cがセットされていれば分岐	**C=1**
BEQ	等しいならば分岐	Z=1
BGE	より以上ならば分岐（符号付きデータ比較）	(N xor V)=0
BGT	より大ならば分岐　（符号付きデータ比較）	(Z and (N xor V))=0
BHI	より大ならば分岐　（符号無しデータ比較）	(C or Z)=0
BHS	より以上ならば分岐（符号無しデータ比較）	**C=0**
BLE	より以下ならば分岐（符号付きデータ比較）	(Z or (N xor V))=0
BLO	より小ならば分岐　（符号無しデータ比較）	**C=1**
BLS	より以下ならば分岐（符号無しデータ比較）	(C or Z)=1
BLT	より小ならば分岐　（符号付きデータ比較）	(N xor V)=1
BMI	負数ならば分岐	N=1
BNE	等しくないならば分岐	Z=0
BPL	正数ならば分岐	N=0
BVC	Vがセットされていなければ分岐	V=0
BVS	Vがセットされていれば分岐	V=1

これらの基本命令は、
CCを利用はするが、
「CCを変更しない基本命令」
である。

それぞれ
基本命令名は異なるが、
判定条件が同じ。
→OPcodeが同じ。
（調べてみよう）

・6809_Insn_Branch-AL.pdf
6809_Insn_Branch-FULL.pdf

制御構造の構成要素②：処理

判断 ＝ 条件判定ありの条件分岐基本命令

【注】CCを変更する基本命令は全て「処理」である

処理 ＝ 左を除く、全ての基本命令

実行順序変更の基本命令

条件判定ありの条件分岐基本命令

BCC	Cがセットされていなければ分岐
BCS	Cがセットされていれば分岐
BEQ	等しいならば分岐
BGE	より以上ならば分岐（符号付きデータ比較）
BGT	より大ならば分岐　（符号付きデータ比較）
BHI	より大ならば分岐　（符号無しデータ比較）
BHS	より以上ならば分岐（符号無しデータ比較）
BLE	より以下ならば分岐（符号付きデータ比較）
BLO	より小ならば分岐　（符号無しデータ比較）
BLS	より以下ならば分岐（符号無しデータ比較）
BLT	より小ならば分岐　（符号付きデータ比較）
BMI	負数ならば分岐
BNE	等しくないならば分岐
BPL	正数ならば分岐
BVC	Vがセットされていなければ分岐
BVS	Vがセットされていれば分岐

（計16）

左を除く、全ての基本命令

CCを変更しない基本命令

BRA	無条件に分岐
BRN	分岐しない分岐
BSR	サブルーチンへ分岐

（計3）

JMP	ジャンプ
JSR	サブルーチンへジャンプ
RTS	サブルーチンからの復帰
SWI	SW割込み
ABX	BとXの加算
EXG	Reg間データ交換
PSH	プッシュ
PUL	プル（ポップ）
TFR	Reg間データ転送
NOP	無操作
SYNC	HW割込み待ち

（計11）

LEAS	Sに実効アドレスをロード
LEAU	Uに実効アドレスをロード

CCを変更する基本命令

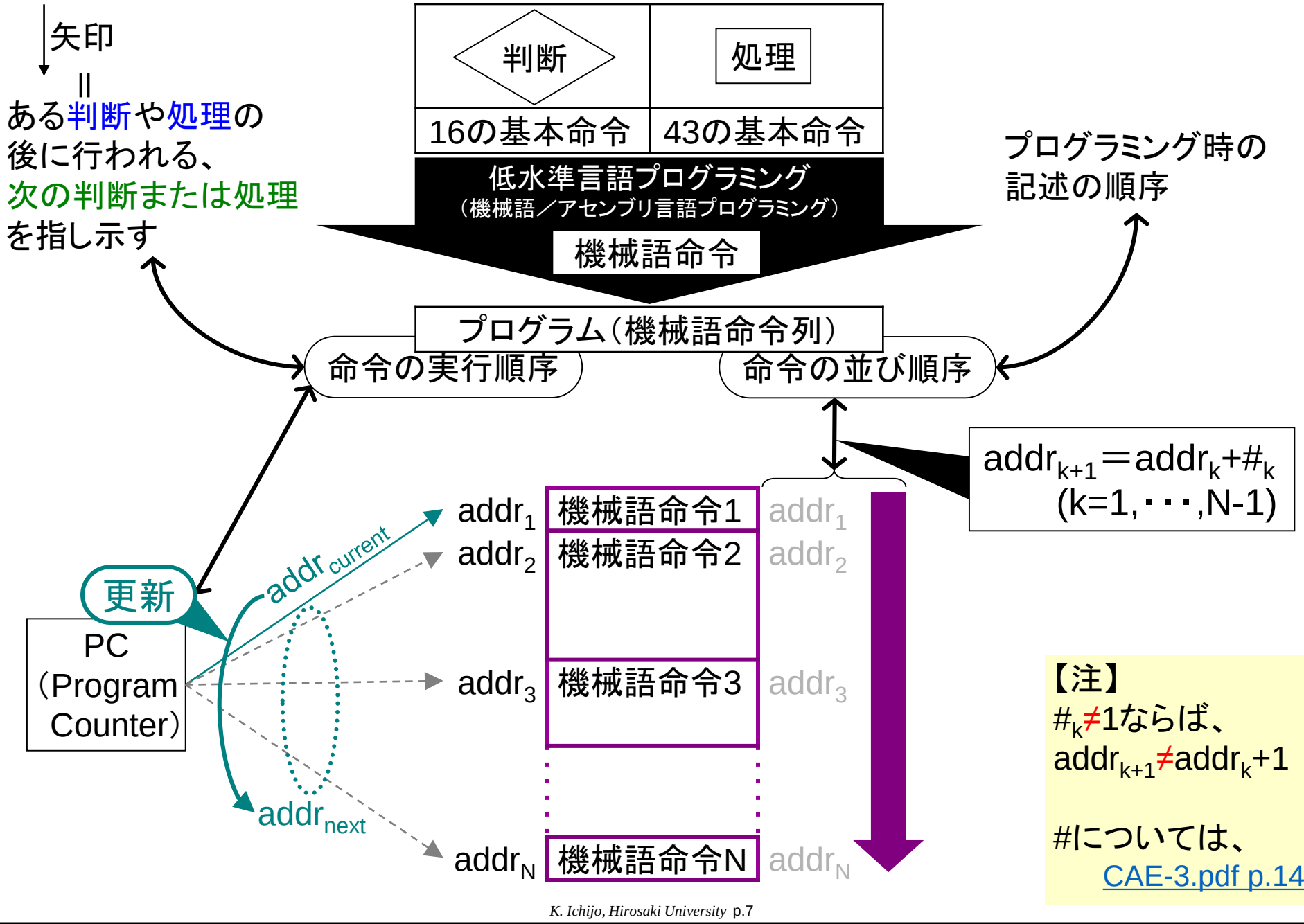
RTI	割込みルーチンからの復帰
ADC	桁上げ付き加算
ADD	加算
CLR	0代入
DAA	10進補正
DEC	1減
INC	1増
MUL	乗算
NEG	符号反転
SBC	桁上げ付き減算
SEX	符号付ビット拡張
SUB	減算
AND	論理積
COM	ビット反転
EOR	排他的論理和
OR	論理和
LD	ロード
ST	ストア
ASL	算術左シフト
ASR	算術右シフト
LSL	論理左シフト
LSR	論理右シフト
ROL	左循環シフト
ROR	右循環シフト
BIT	論理比較
CMP	算術比較
TST	テスト（0減）
CWAI	Regを退避し、HW割込み待ち

（計28）

LEAX	Xに実効アドレスをロード
LEAY	Yに実効アドレスをロード

基本命令 LEA　実効アドレスをロード

制御構造の構成要素③：矢印（1/3）



【注】
$\#_k \neq 1$ならば、
$addr_{k+1} \neq addr_k + 1$

#については、
CAE-3.pdf p.14

制御構造の構成要素③：矢印（2/3）

矢印
＝
ある判断や処理の
後に行われる、
次の判断または処理
を指し示す

判断	処理
16の基本命令	43の基本命令

低水準言語プログラミング
（機械語／アセンブリ言語プログラミング）

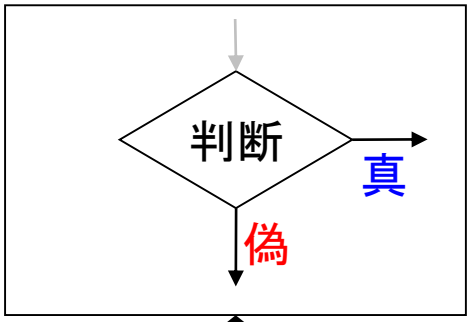
機械語命令

プログラム（機械語命令列）

命令の実行順序

PC　更新　の仕方（4種類）

判断における
2つの次への矢印と
条件判定結果との関係



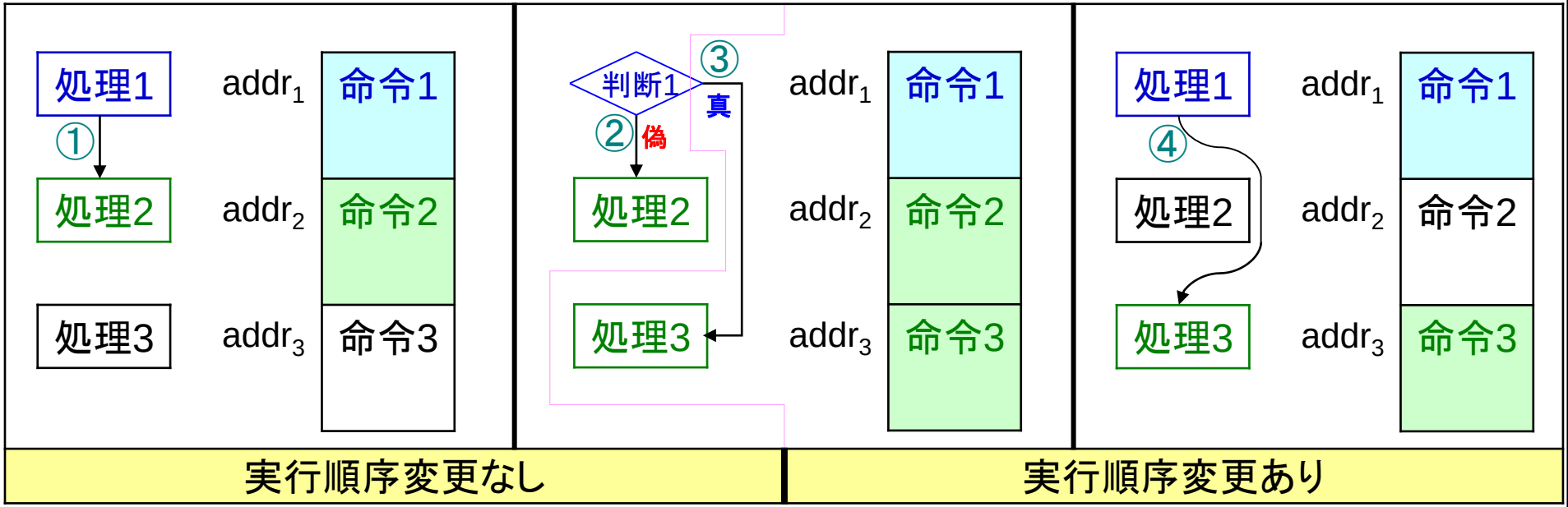
①	通常（②～④以外）の更新 $addr_{current}=addr_k \Rightarrow addr_{next}=addr_k+\#_k$
②	条件分岐の基本命令による更新・・・条件判定結果が偽のときの更新 $addr_{current}=addr_k \Rightarrow addr_{next}=addr_k+\#_k$ （①と同）
③	条件分岐の基本命令による更新・・・条件判定結果が真のときの更新 $addr_{current}=addr_k \Rightarrow addr_{next}=$ operandが示すアドレス【※】
④	実行順序変更の基本命令 ≫ による更新 $addr_{current}=addr_k \Rightarrow addr_{next}=$ operandが示すアドレス【※】 （③と同）

【※】アドレッシングモードに留意する。「operand」は機械語命令／アセンブリ言語文どちらのもので考えても同じ。

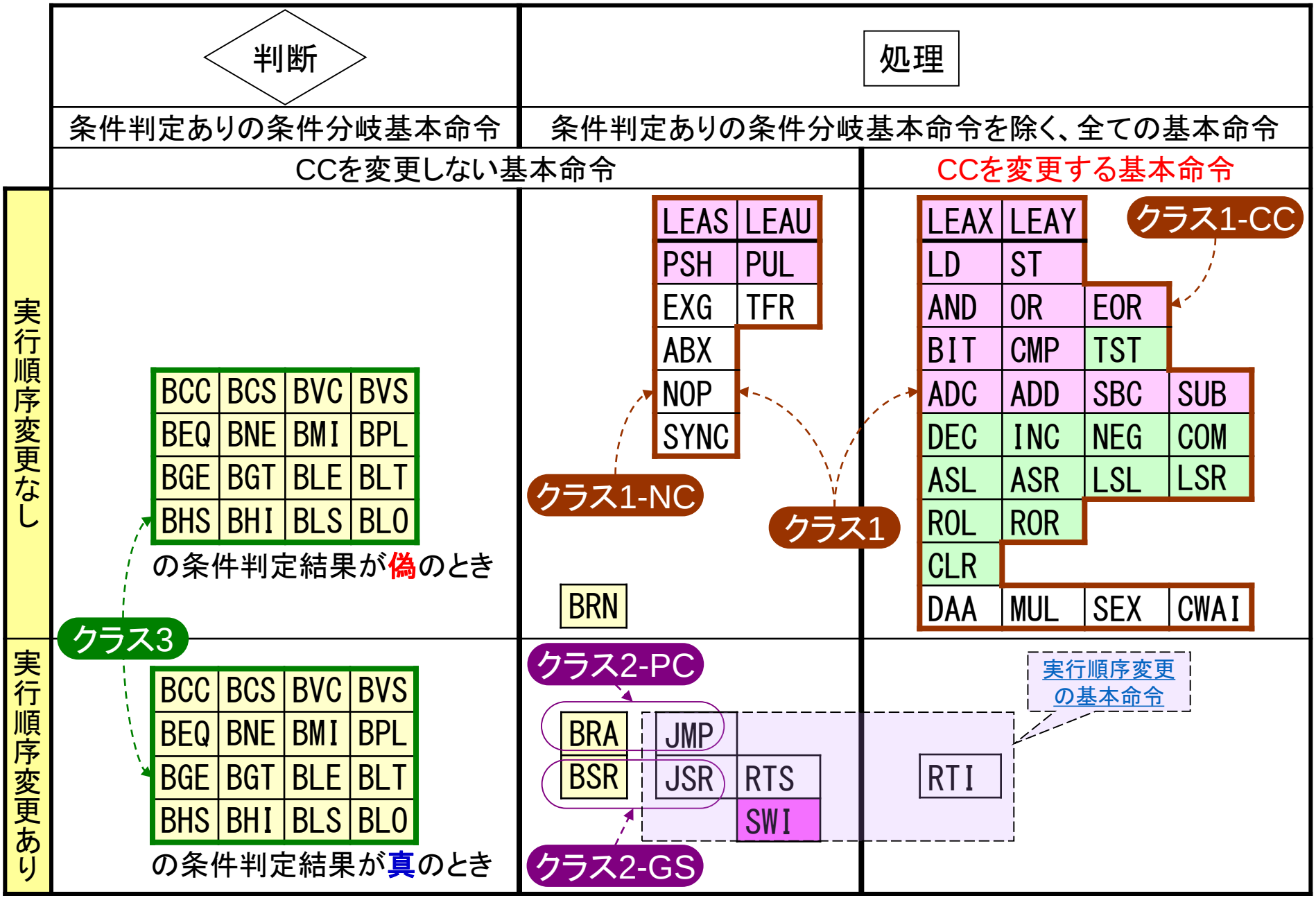
制御構造の構成要素③：矢印（3/3）

PC 更新 の仕方（4種類）

①	通常（②～④以外）の更新 $addr_{current}$ ＝$addr_k$ ⇒ $addr_{next}$＝$addr_k$+$\#_k$	実行順序変更なし
②	条件分岐の基本命令による更新・・・条件判定結果が偽のときの更新 $addr_{current}$ ＝$addr_k$ ⇒ $addr_{next}$＝$addr_k$+$\#_k$ （①と同）	
③	条件分岐の基本命令による更新・・・条件判定結果が真のときの更新 $addr_{current}$ ＝$addr_k$ ⇒ $addr_{next}$＝ operandが示すアドレス	実行順序変更あり
④	実行順序変更の基本命令 ≫ による更新 $addr_{current}$ ＝$addr_k$ ⇒ $addr_{next}$＝ operandが示すアドレス （③と同）	

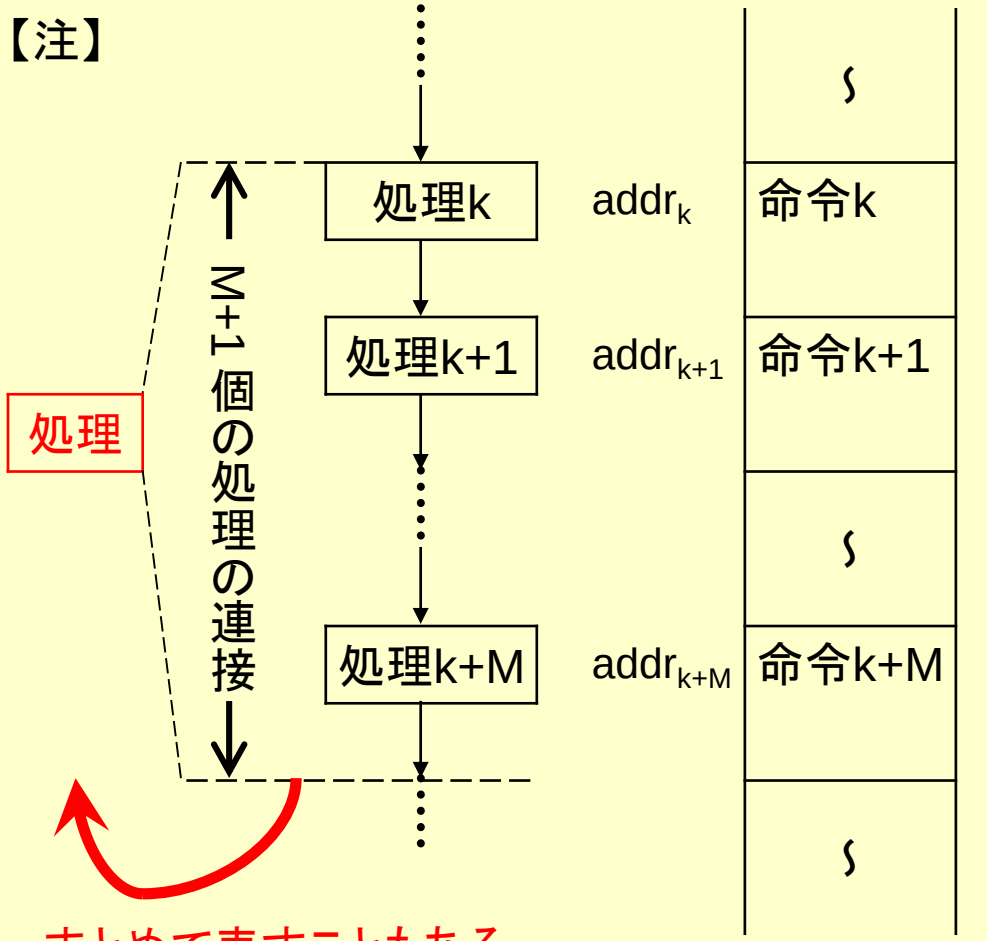


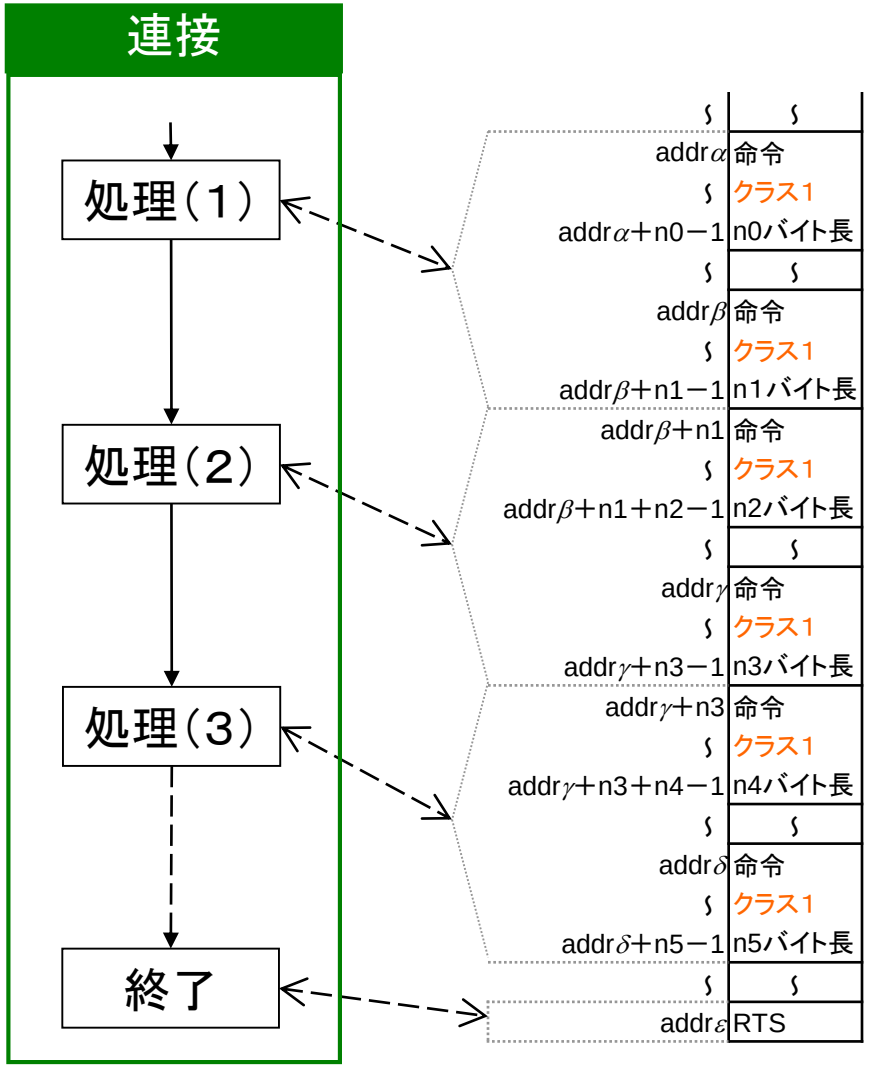
判断、処理、矢印のまとめ（1）



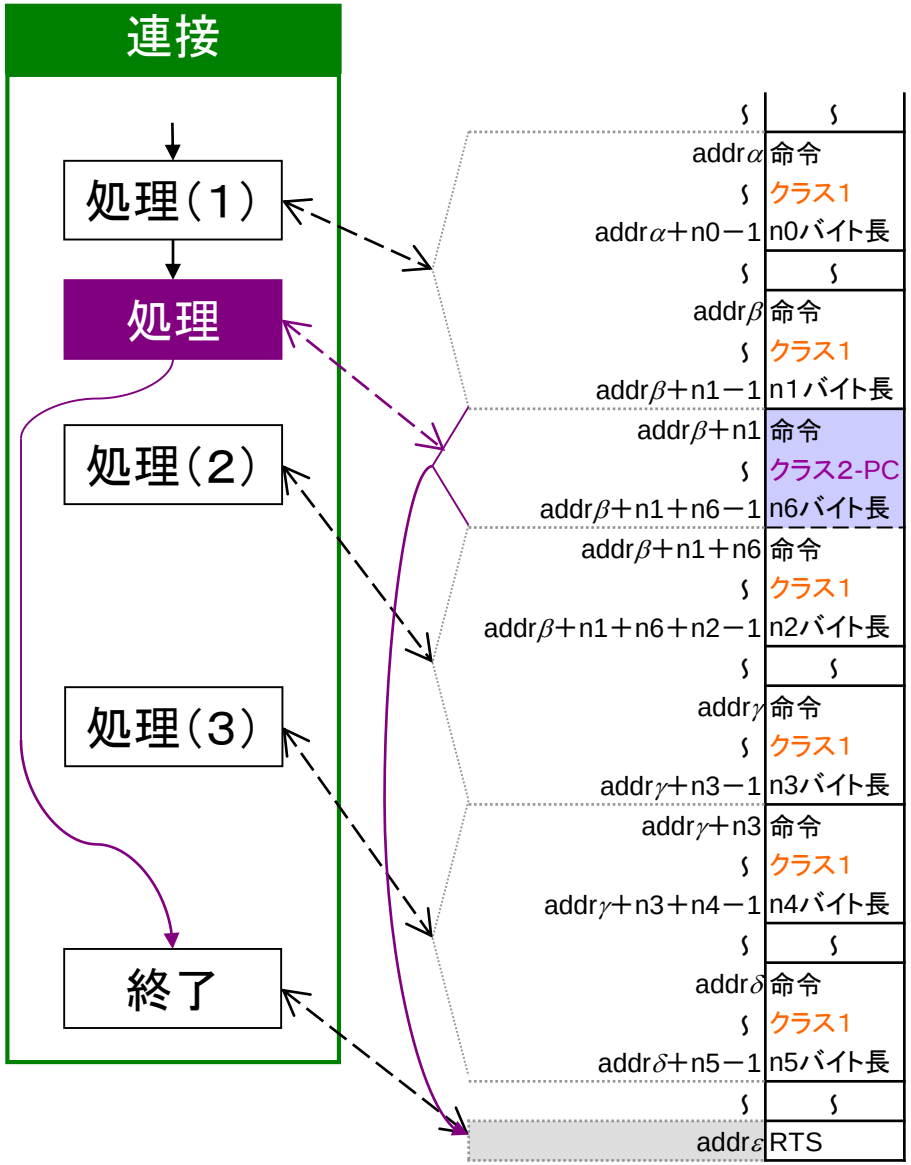
判断、処理、矢印のまとめ(2)

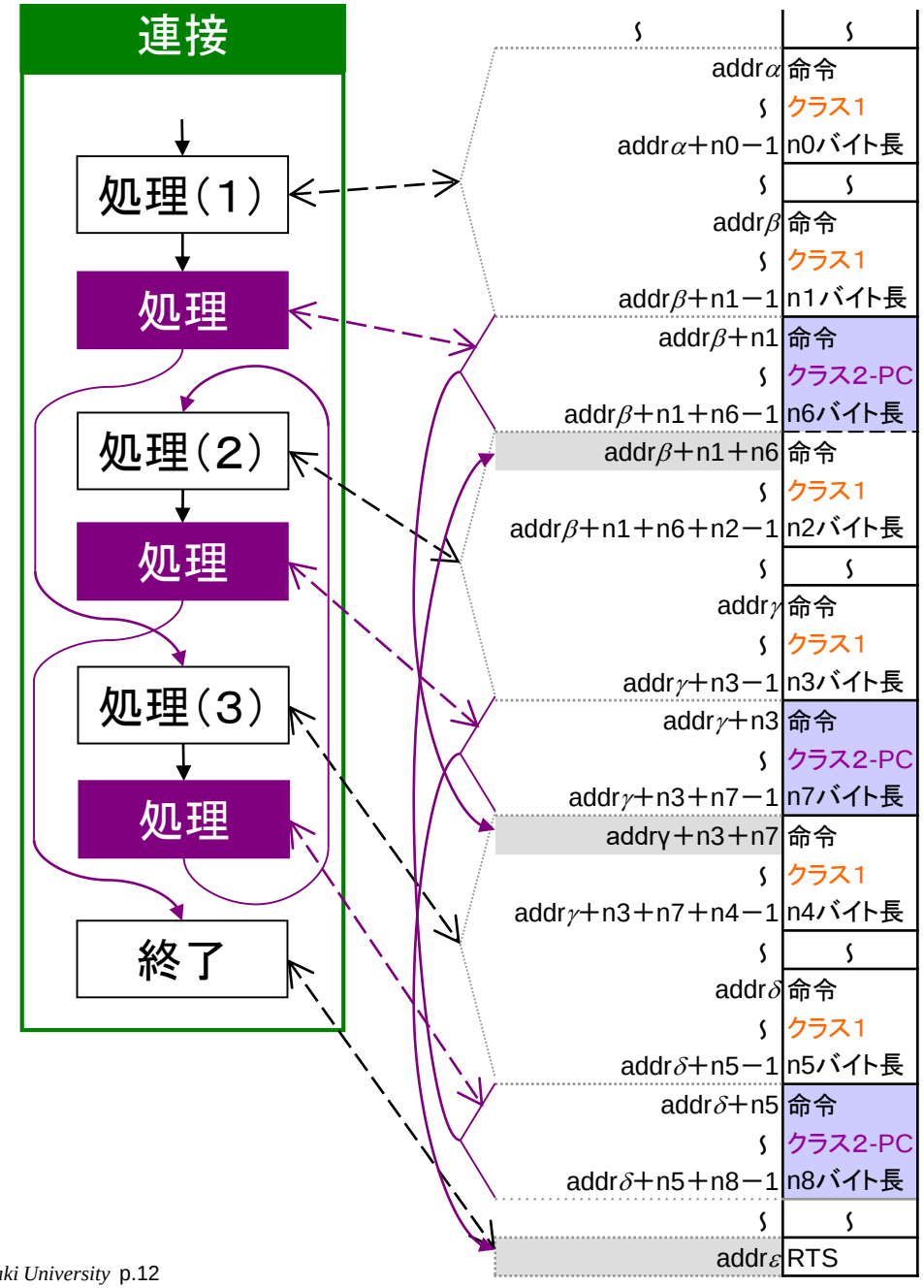
【注】



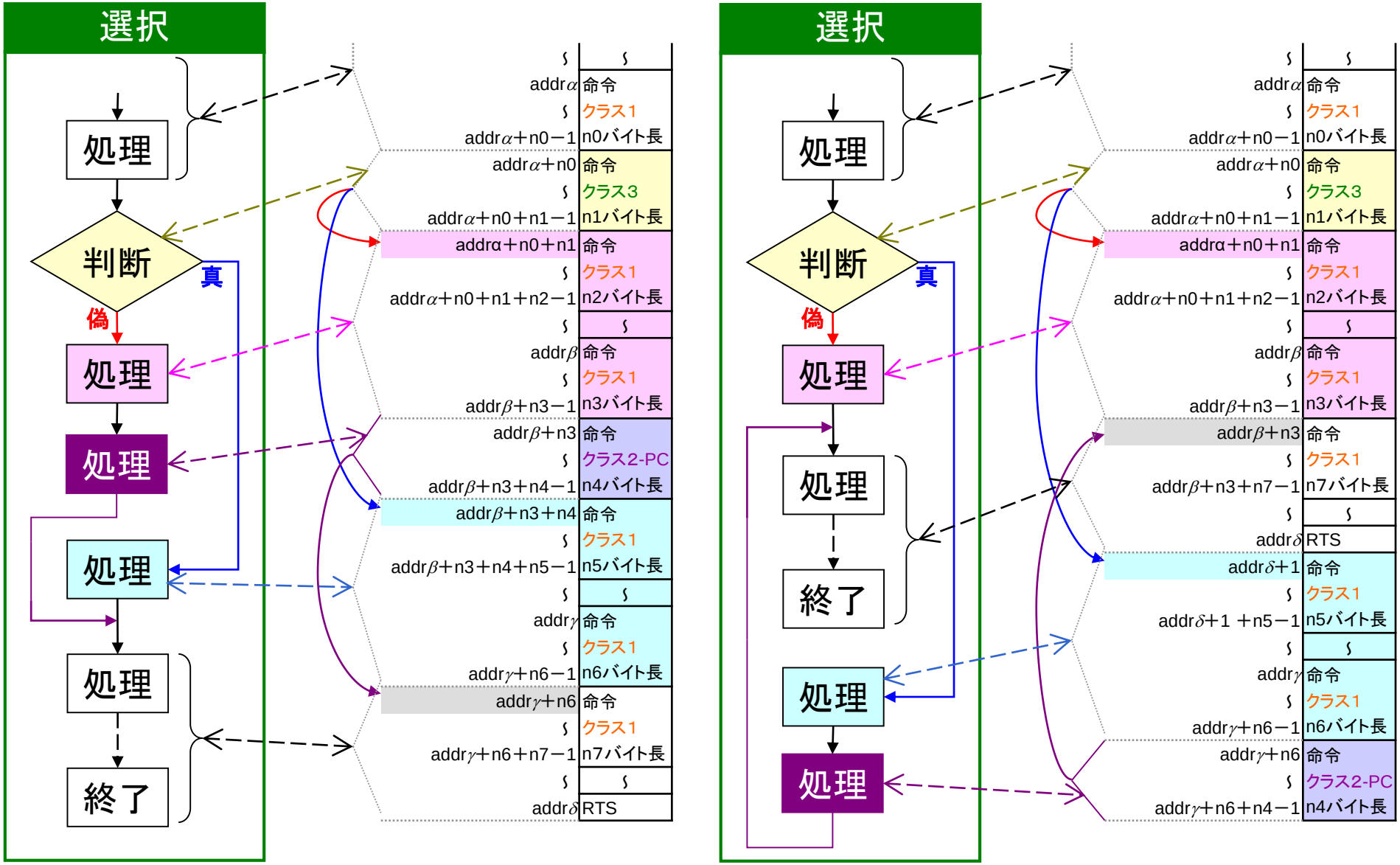


判断、処理、矢印のまとめ（3）



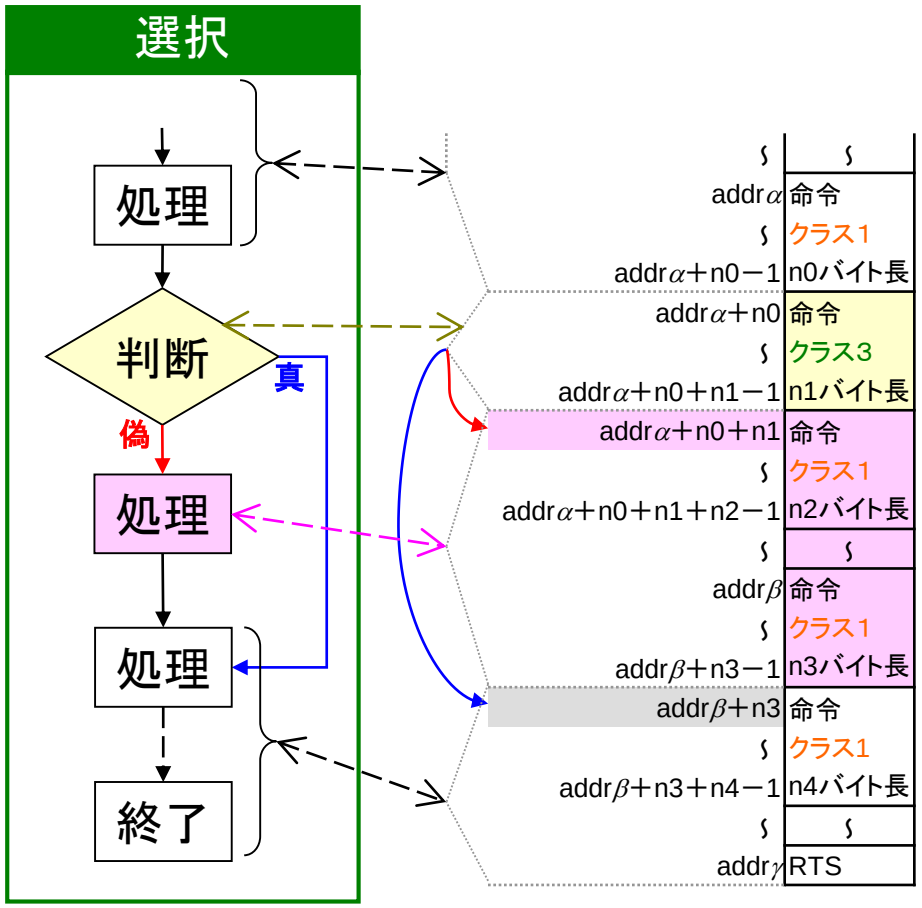


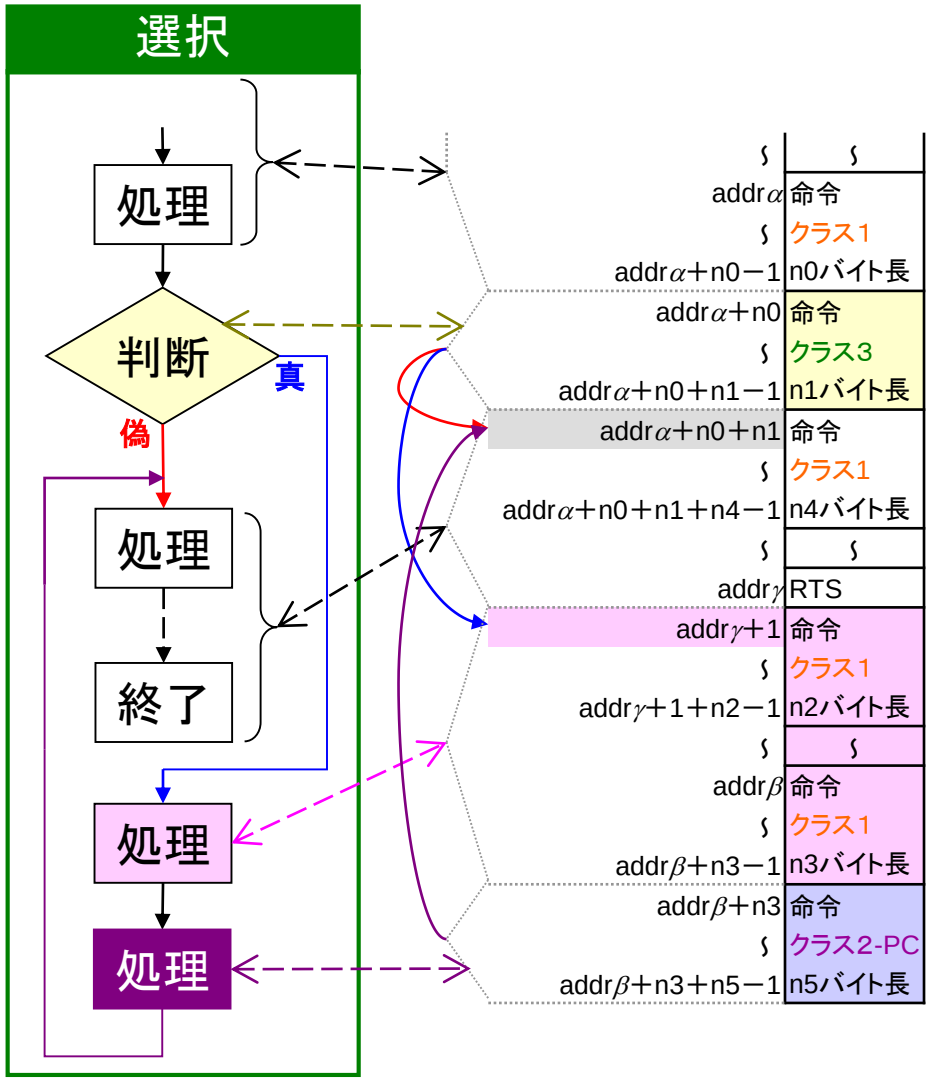
判断、処理、矢印のまとめ(4)



判断結果が真⇒ 処理　　判断結果が偽⇒ 処理

判断、処理、矢印のまとめ(5)

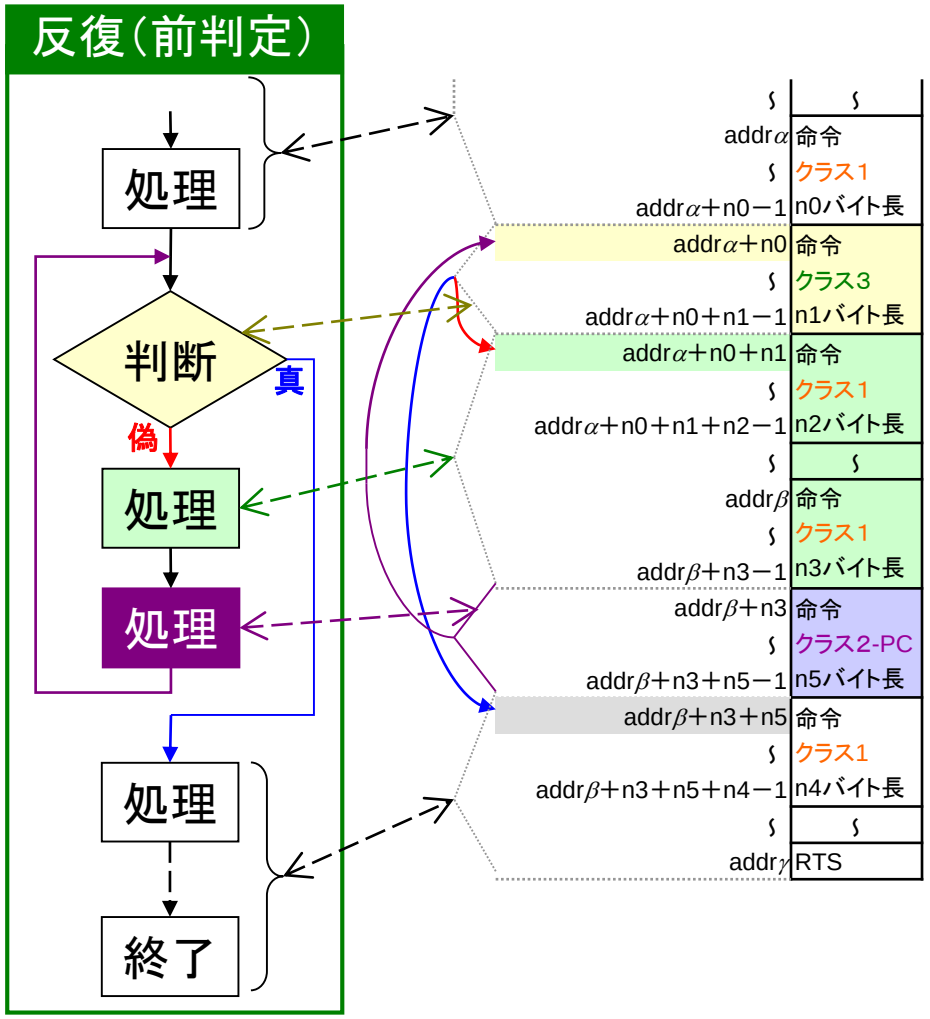




判断結果が偽⇒ 処理 も行う。
判断結果が真⇒ 処理 はスルーする。

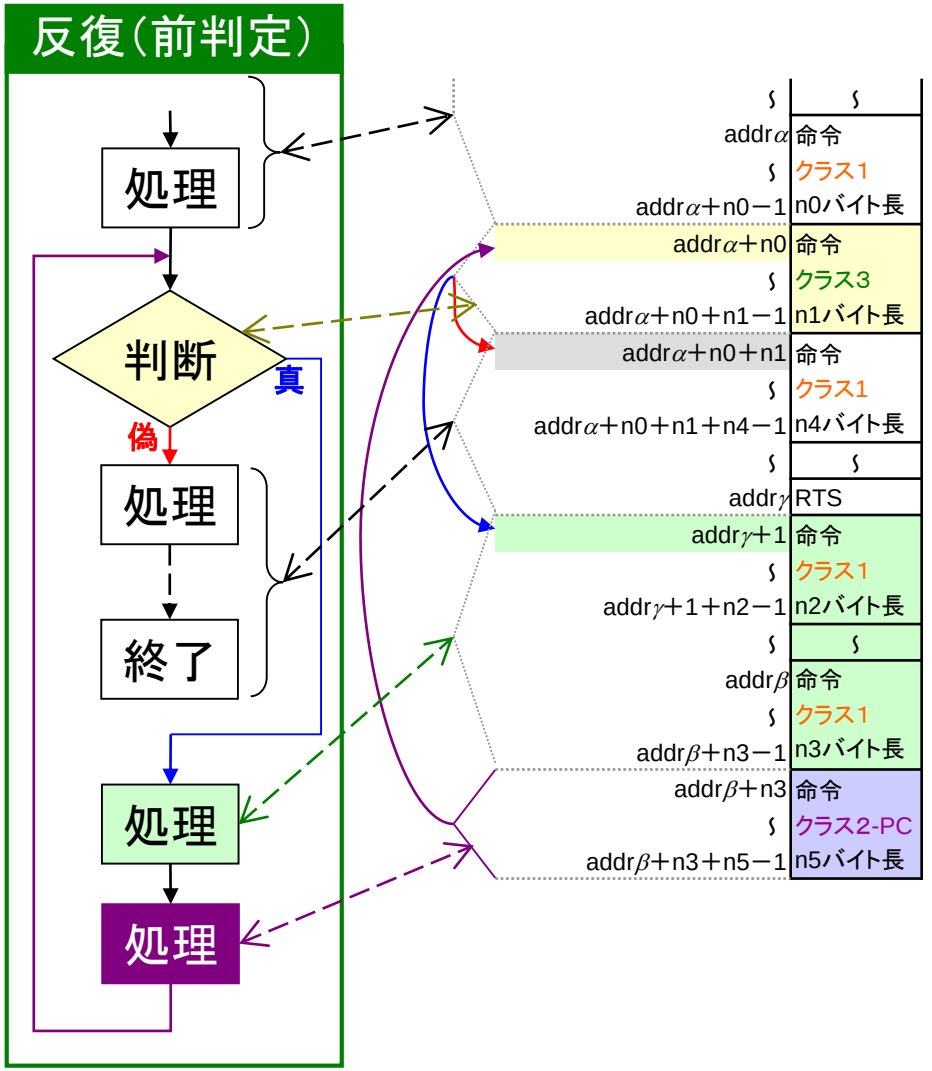
判断結果が真⇒ 処理 も行う。
判断結果が偽⇒ 処理 はスルーする。

判断、処理、矢印のまとめ(6)



判断結果が偽⇒ 処理 を反復する。

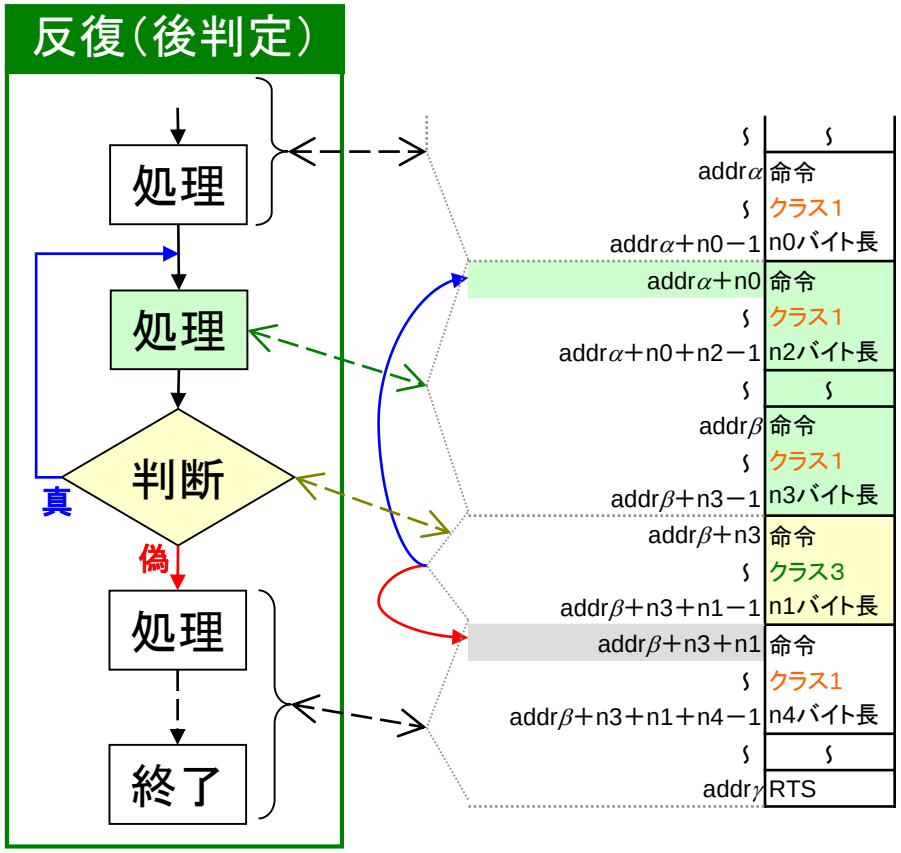
判断結果が真⇒反復をやめる。

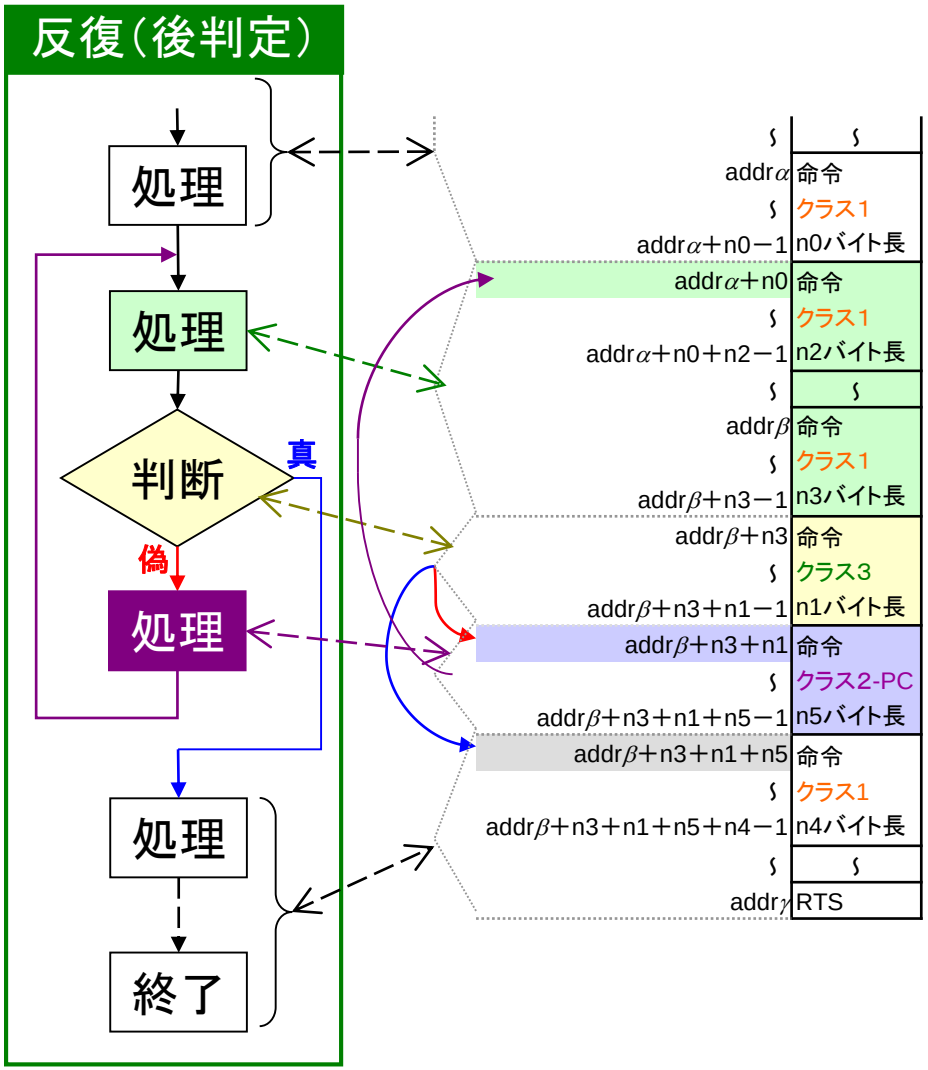


判断結果が真⇒ 処理 を反復する。

判断結果が偽⇒反復をやめる。

判断、処理、矢印のまとめ(7)





判断結果が真⇒ 処理 を反復する。

判断結果が偽⇒反復をやめる。

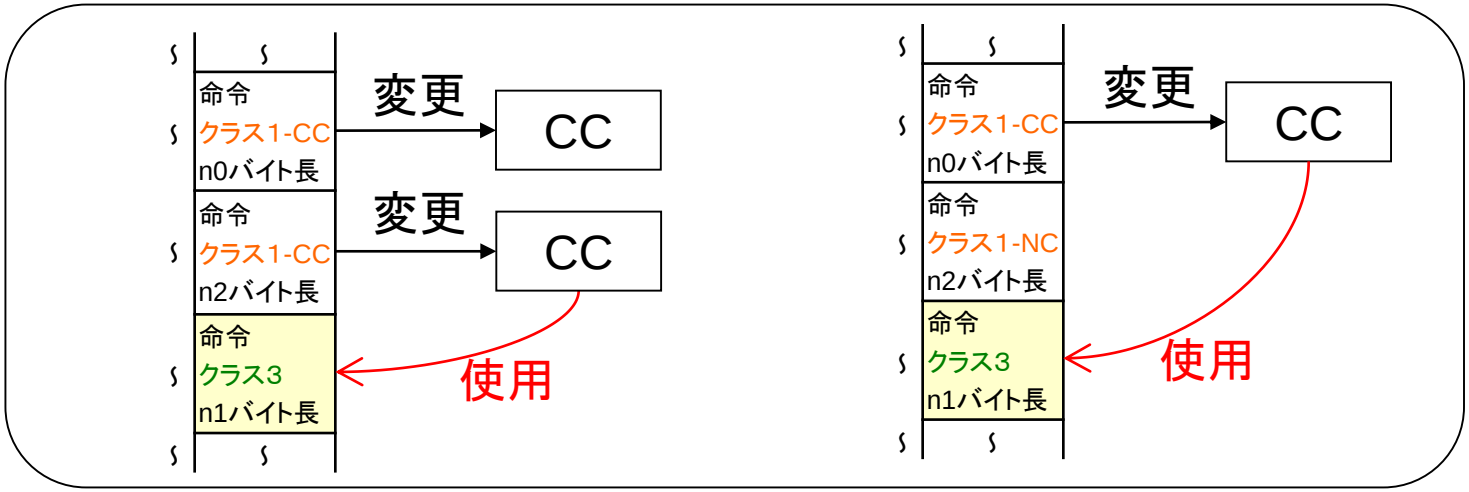
判断結果が偽⇒ 処理 を反復する。

判断結果が真⇒反復をやめる。

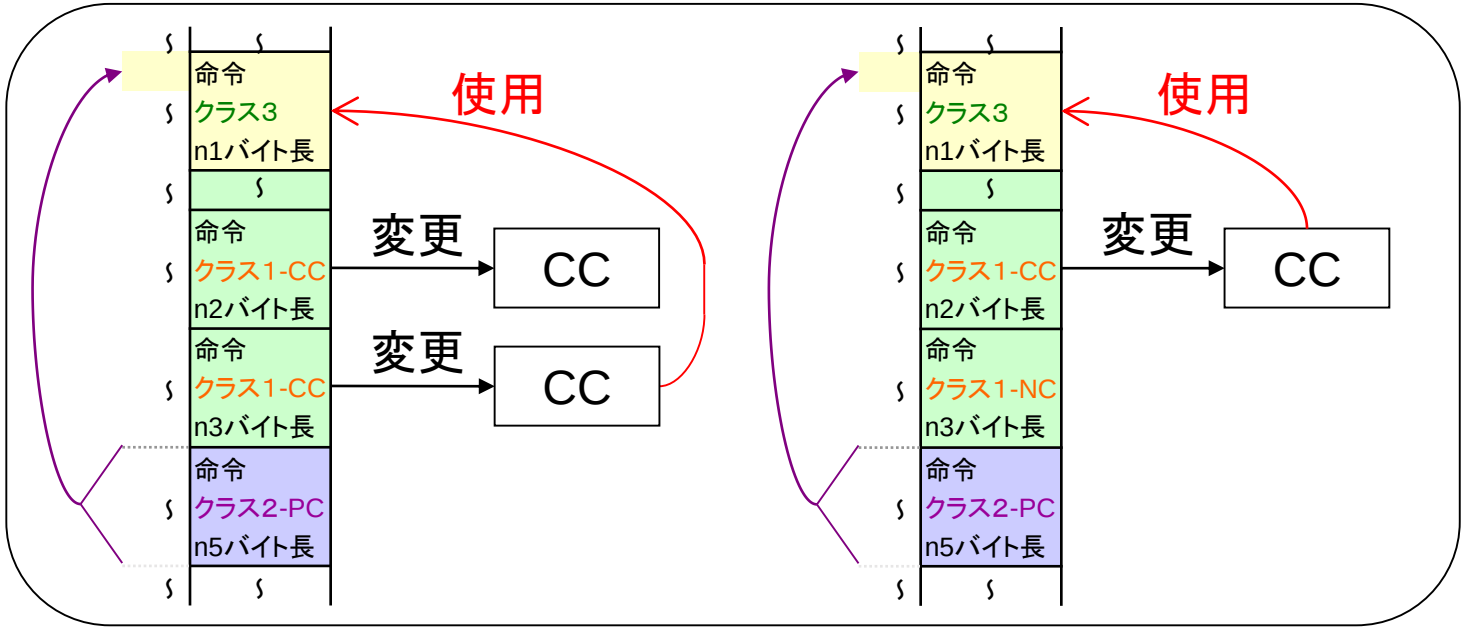
判断、処理、矢印のまとめ(8)

判断は、それが行われる以前の、最後に変更されたCCを使用する。

選択、
反復(後判定)
での例



反復(前判定)
での例



判断、処理、矢印のまとめ（9）

クラス1-CCに属する基本命令種類		左の種類に属する命令の実行後に可能な判断		判定用ビット	判定に使える条件分岐基本命令
データ転送		転送したデータの正／負による判断		N	BPL,BMI
		転送したデータの零／非零による判断		Z	BEQ,BNE
算術演算		算術演算結果による判断		演算の種類により様々	
論理演算		論理演算結果の正／負による判断		N	BPL,BMI
		論理演算結果の零／非零による判断		Z	BEQ,BNE
シフト		シフトの結果による判断		シフトの種類により様々	
比較（αはAcc、βはAcc/Ptr、γはオペランド）	算術比較 CMP$\beta\diamond\gamma$	βとγの算術比較結果（$\beta-\gamma$）による判断	$\beta-\gamma>0$（$\Leftrightarrow\beta>\gamma$）	Z,C	BHI（符号無し比較）
				N,Z,V	BGT（符号付き比較）
			$\beta-\gamma\geqq0$（$\Leftrightarrow\beta\geqq\gamma$）	C	BHS（符号無し比較）
				N,V	BGE（符号付き比較）
			$\beta-\gamma=0$（$\Leftrightarrow\beta=\gamma$）	Z	BEQ
			$\beta-\gamma\leqq0$（$\Leftrightarrow\beta\leqq\gamma$）	Z,C	BLS（符号無し比較）
				N,Z,V	BLE（符号付き比較）
			$\beta-\gamma<0$（$\Leftrightarrow\beta<\gamma$）	C	BLO（符号無し比較）
				N,V	BLT（符号付き比較）
			$\beta-\gamma\neq0$（$\Leftrightarrow\beta\neq\gamma$）	Z	BNE
	論理比較 BIT$\alpha\diamond\gamma$	αとγの論理比較結果（α AND γ）による判断	正／負	N	BPL,BMI
			零／非零	Z	BEQ,BNE
LEAX,LEAY		ロードした実効アドレスの零／非零による判断		Z	BEQ,BNE

【注】全てのクラス1-CCに属する基本命令について示していない。必要ならば 適宜別資料を参照 のこと。

課題　クラス1（「実行順序変更なし」,「処理」）に属する命令で作るプログラム

4つの1バイトデータ$07,$35,$81,$C6を考える。

これについて以下のプログラムを作成せよ。

（1-A）	Immediateアドレッシングモードを用いてデータを参照し、 $10F0～$10F3番地にデータをコピーし、終了する。
（1-C）	Immediateアドレッシングモードを用いてデータを参照し、 これらの総和を求め、その結果を$10F0番地から格納し、終了する。
（2-A）	擬似命令FCBを用いてメモリに格納してデータを参照し、 元データはそのままに、$10F0～$10F3番地にデータをコピーし、終了する。
（2-B）	擬似命令FCBを用いてメモリに格納してデータを参照し、 元データを消し（0代入し）つつ 、$10F0～$10F3番地にデータをコピーし、終了する。【移動】
（2-C）	擬似命令FCBを用いてメモリに格納してデータを参照し、 元データはそのままに、これらの総和を求め、その結果を$10F0番地から格納し、終了する。
（3-A）	mコマンドを用いて$10F0～$10F3番地以外のメモリに格納したデータを参照し、 元データはそのままに、$10F0～$10F3番地にデータをコピーし、終了する。
（3-B）	mコマンドを用いて$10F0～$10F3番地以外のメモリに格納したデータを参照し、 元データを消し（0代入し）つつ 、$10F0～$10F3番地にデータをコピーし、終了する。【移動】
（3-C）	mコマンドを用いて$10F0～$10F3番地以外のメモリに格納したデータを参照し、 元データはそのままに、これらの総和を求め、その結果を$10F0番地から格納し、終了する。

【注】・PtrであるIndex Reg（XとY）、Indexed モードのアドレッシング（特にIndexed Auto Increment）を活用せよ。
・総和結果のバイト数に留意せよ。
・『終了する』とは、ニーモニック「RTS」（機械語命令「39」）を実行すること。

課題　クラス2-PC(「実行順序変更あり」,「処理」)に属する命令も含むプログラム

4つの1バイトデータ$07,$35,$81,$C6を考える。

①まず、これについて以下の3つの作業を定める。

(4-A)	擬似命令FCBを用いてメモリに格納してデータを参照し、 元データはそのままに、$10F0～$10F3番地にデータをコピーする。
(4-B)	擬似命令FCBを用いてメモリに格納してデータを参照し、 元データはそのままに、これらの総和を求め、その結果を$10F8番地から格納する。
(4-C)	擬似命令FCBを用いてメモリに格納してデータを参照し、 元データを消し(0代入し)つつ 、$10F4～$10F7番地にデータをコピーする。「移動」

②次に、この3つの作業とRTSの並び順序を、以下のように定める。

並び順序	(4-A)，　(4-B)，　(4-C)，　RTS

このとき、必要な命令を適宜追加して、以下の実行順序に従って動作する、
7つのプログラムを作成せよ。 ただし、・②で定めた並び順序を変えてはならない。
・3つの作業のいずれの記述も削除してはならない。

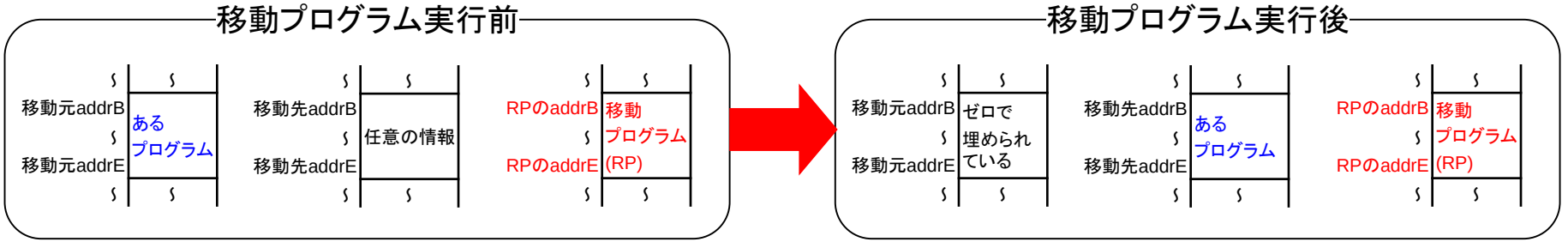
実行順序1	(4-A)，　RTS
実行順序2	(4-B)，　RTS
実行順序3	(4-C)，　RTS
実行順序4	(4-A)，　(4-B)，　RTS
実行順序5	(4-A)，　(4-C)，　RTS
実行順序6	(4-B)，　(4-C)，　RTS
実行順序7	(4-B)，　(4-A)，　(4-C)，　RTS

課題　クラス3（「判断」）に属する命令も含むプログラム

（1）
10進数の0から10までの総和を求めるプログラムを、以下に従って作成せよ。
（A）クラス1の命令群のものだけを用いて作成せよ。
（B）クラス3の命令群のものも併用して作成せよ。

（2）
N個の任意の1バイトデータを考える。（ただし、「その総和≦$(511)_{10}$」を満たすものとする）
これについて、
（A）データを適当な方法でメモリ上に配置し、その総和を求めるプログラムを作成せよ。
（B）データを適当な方法でメモリ上に配置し、配置した場所とは別の場所に、
　　データの順序を保存したまま、元を消して（0代入して）コピーする（移動する）
　　プログラムを作成せよ。

（3）
今、あるプログラムがメモリ上に置かれているとする。
これを、別の場所に移動するプログラムを作成せよ。



【注】・移動元addrBは「あるプログラム」中のORG擬似命令で指定した値。
　　・3つのメモリ領域が重なり合ってはいけない。（当たり前）
　　・移動元addrB、移動先addrB、RPのaddrB、の大小関係は問わない。