

プログラミング演習1

課題4: アセンブリ言語

第2回 2進数の印刷など

2017年7月4日

- 座席は自由です
- 講義資料ページを開いておく
- WebClassで出席を登録する準備

課題 (宿題) のチェック

- 例題の説明後，今週の課題に入ってから確認
- 先週分の宿題ができていない人はこの時間内，もしくは来週までには必ず取り組んでくること
- 最終週 (7/11) までに課題が終わっていない人は別途個別課題を出します

課題がすべてクリアできた状態でないと
レポート課題に取り組むのはほぼ無理です

例題4:2進数の印刷

```
;-----  
; 例4 2進数の印刷  
;-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE   ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2   ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3  ;GR3 ← -1 + GR3  
      CPA GR3,ZERO    ;GR3と0を算術比較  
      JPL LOOP       ;GR3 > 0 ならLOOPへ  
      JZE LOOP       ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```

例題4:2進数の印刷

- 文字列の出力: OUT 命令
 - OUT DATA02, OUTLEN ではダメなのか??

- OUT命令は文字列しか出力できない



DATA02に入っている値が0か1かを判定
⇒ 文字列に変換

例題の補足ページ:

<http://www.ced.is.utsunomiya-u.ac.jp/lecture/2017/prog/p1/kadai4/hosoku.htm>

例題4:2進数の印刷

□処理の流れ

1. ループカウンタを設定
2. 最下位1ビットが1か0かの判定を行う (マスク処理)
3. 1 or 0の文字コードを計算
4. 最下位ビットの表示位置に計算した文字コード格納
5. DATA02を1ビット右にシフト, 表示位置を一つ左に
6. 1. の処理に戻る

例題の補足ページ:

<http://www.ced.is.utsunomiya-u.ac.jp/lecture/2017/prog/p1/kadai4/hosoku.htm>

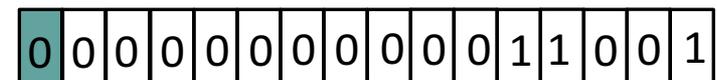
例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
LAD GR3,15 ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
LD GR1,DATA02 ;GR1 ← (DATA02)  
LOOP LAD GR2,0,GR1 ;GR1をGR2に退避  
AND GR1,HEX01 ;GR1と#0001との論理積を取る  
ADDL GR1,ZONE ;'0'との論理加算により文字コードを得る  
ST GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
LAD GR1,0,GR2 ;GR1を復元  
SRL GR1,1 ;GR1を1ビット論理右シフト  
LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
CPA GR3,ZERO ;GR3と0を算術比較  
JPL LOOP ;GR3 > 0 ならLOOPへ  
JZE LOOP ;GR3 = 0 ならLOOPへ  
OUT OUTBUF,OUTLEN ;文字を出力  
RET  
DATA02 DC 25 ;←適当に変えてみる  
HEX01 DC #0001  
ZONE DC '0'  
ZERO DC 0  
OUTBUF DS 16 ;16個の領域を確保  
OUTLEN DC 16 ;出力する文字列の長さ16  
END
```

符号ビット (負:1, 非負:0)

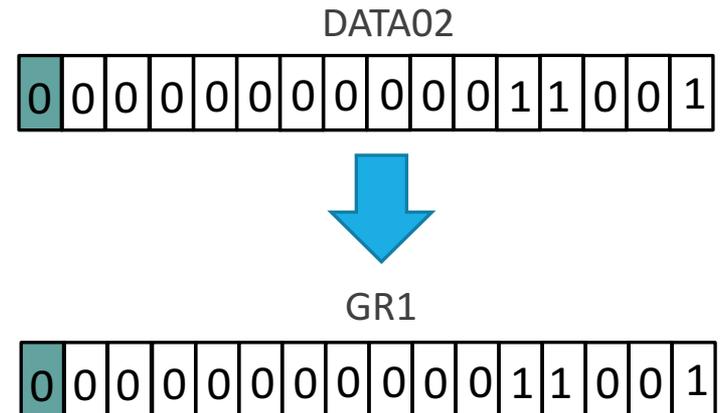


DATA02



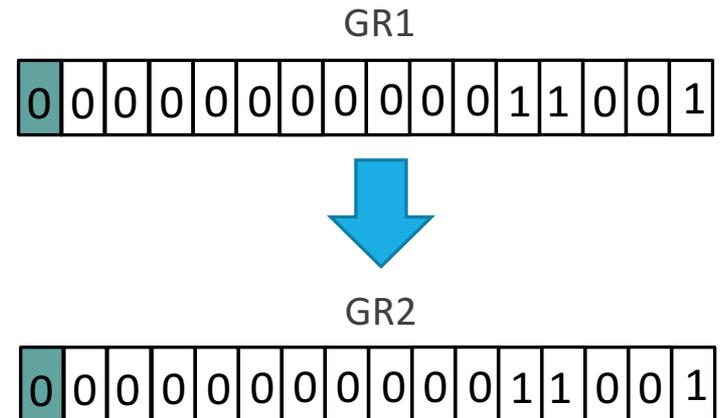
例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE   ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2   ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16     ;16個の領域を確保  
OUTLEN DC 16     ;出力する文字列の長さ16  
END
```



例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避 → GR1は後の処理で論理積を取る (マスク処理) ため  
  AND  GR1,HEX01  ;GR1と#0001との論理積を取る  
  ADDL GR1,ZONE   ;'0'との論理加算により文字コードを得る  
  ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
  LAD GR1,0,GR2   ;GR1を復元  
  SRL GR1,1      ;GR1を1ビット論理右シフト  
  LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
  CPA GR3,ZERO   ;GR3と0を算術比較  
  JPL LOOP      ;GR3 > 0 ならLOOPへ  
  JZE LOOP      ;GR3 = 0 ならLOOPへ  
  OUT OUTBUF,OUTLEN ;文字を出力  
  RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16     ;16個の領域を確保  
OUTLEN DC 16     ;出力する文字列の長さ16  
END
```



例題4:2進数の印刷

```
;-----  
; 例4 2進数の印刷  
;-----
```

```
EX4 START
```

```
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))
```

```
  LD  GR1,DATA02  ;GR1 ← (DATA02)
```

```
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避
```

```
  AND  GR1,HEX01  ;GR1と#0001との論理積を取る → マスク処理 (最下位ビットが 0 or 1)
```

```
  ADDL GR1,ZONE   ;'0'との論理加算により文字コードを得る
```

```
  ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア
```

```
  LAD GR1,0,GR2   ;GR1を復元
```

```
  SRL GR1,1       ;GR1を1ビット論理右シフト
```

```
  LAD GR3,-1,GR3  ;GR3 ← -1 + GR3
```

```
  CPA GR3,ZERO    ;GR3と0を算術比較
```

```
  JPL LOOP        ;GR3 > 0 ならLOOPへ
```

```
  JZE LOOP        ;GR3 = 0 ならLOOPへ
```

```
  OUT OUTBUF,OUTLEN ;文字を出力
```

```
  RET
```

```
DATA02 DC 25      ;←適当に変えてみる
```

```
HEX01  DC #0001
```

```
ZONE   DC '0'
```

```
ZERO   DC 0
```

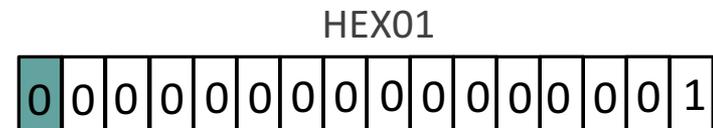
```
OUTBUF DS 16      ;16個の領域を確保
```

```
OUTLEN DC 16      ;出力する文字列の長さ16
```

```
  END
```



+ AND (論理積)



||



例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16     ;16個の領域を確保  
OUTLEN DC 16     ;出力する文字列の長さ16  
END
```

文字列変換



例題4:2進数の印刷

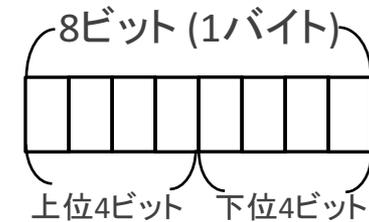
□ 文字コード

上位4ビット →

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0				0	@	P	'	p				ー	タ	ミ		
1			!	1	A	Q	a	q			。	ア	チ	ム		
2			"	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4			\$	4	D	T	d	t			。	エ	ト	ヤ		
5			%	5	E	U	e	u			・	オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			'	7	G	W	g	w			ァ	キ	ヌ	ラ		
8			(8	H	X	h	x			ィ	ク	ネ	リ		
9)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
B			+	;	K	[k	[ォ	サ	ヒ	ロ		
C			.	<	L	¥	l				ャ	シ	フ	ワ		
D			-	=	M]	m	}			ュ	ス	ヘ	ン		
E			.	>	N	^	n	—			ョ	セ	ホ	。		
F			/	?	O	_	o				ッ	ソ	マ			

下位4ビット ↓

● 1文字のビット構成



文字列 '0'



文字コード一覧 (講義ページ)

<http://www.ced.is.utsunomiya-u.ac.jp/lecture/2017/prog/p1/kadai4/characterCode.htm>

例題4:2進数の印刷

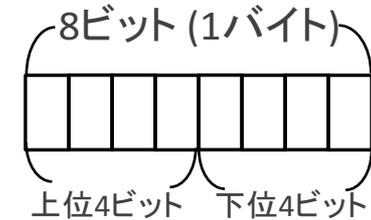
□ 文字コード

上位4ビット →

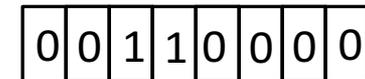
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0				0	@	P	'	p				ー	タ	ミ		
1			!	1	A	Q	a	q			。	ア	チ	ム		
2			"	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4			\$	4	D	T	d	t			。	エ	ト	ヤ		
5			%	5	E	U	e	u			・	オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			'	7	G	W	g	w			ァ	キ	ヌ	ラ		
8			(8	H	X	h	x			ィ	ク	ネ	リ		
9)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
B			+	;	K	[k	[ォ	サ	ヒ	ロ		
C			.	<	L	¥	l				ャ	シ	フ	ワ		
D			-	=	M]	m	}			ュ	ス	ヘ	ン		
E			.	>	N	^	n	_			ョ	セ	ホ	°		
F			/	?	O	_	o				ッ	ソ	マ	°		

下位4ビット ↓

● 1文字のビット構成



文字列 '0'



文字列 '1'



文字コード一覧 (講義ページ)

<http://www.ced.is.utsunomiya-u.ac.jp/lecture/2017/prog/p1/kadai4/characterCode.htm>

例題4:2進数の印刷

□ 文字コードと数値

10進数の7: $(7)_{10} = (07)_{16} = (0000\ 0111)_2$

文字列'7': '7' = $(37)_{16} = (0011\ 0111)_2$

↓
文字コード

↓
 $(0011\ 0111)_2 = (55)_{10}$

□ 命令によって区別される

□ ADDA命令: 2進数の数値のみ扱う

⇒ $(0011\ 0111)_2$ は10進数の55として処理

□ OUT命令: 文字列のみ扱う

⇒ $(0011\ 0111)_2$ は文字列'7'として処理

例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```

文字列変換



例題4:2進数の印刷

□ ADDL (ADD Logical; 論理加算) 命令

[ラベル] ADDL r1, r2

⇒ r1で指定したレジスタの内容とr2で指定したレジスタを
符号のない数値とみなして加算, 結果をr1に入れる

[ラベル] ADDL r, adr[,x]

⇒ rで指定したレジスタの内容とadr[,x]で指定した主記憶の
内容を符号のない数値とみなして加算, 結果をrに入れる

例題4:2進数の印刷

□ 文字列への変換

数値 $(1)_{10}$

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

+ ADDL
(ADD Logical; 論理加算)

文字列 '0'

0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0

||

文字列 '1'

0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 1

数値 $(0)_{10}$

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

+ ADDL
(ADD Logical; 論理加算)

文字列 '0'

0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0

||

文字列 '0'

0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0

例題4:2進数の印刷

```
;-----  
; 例4 2進数の印刷  
;-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```

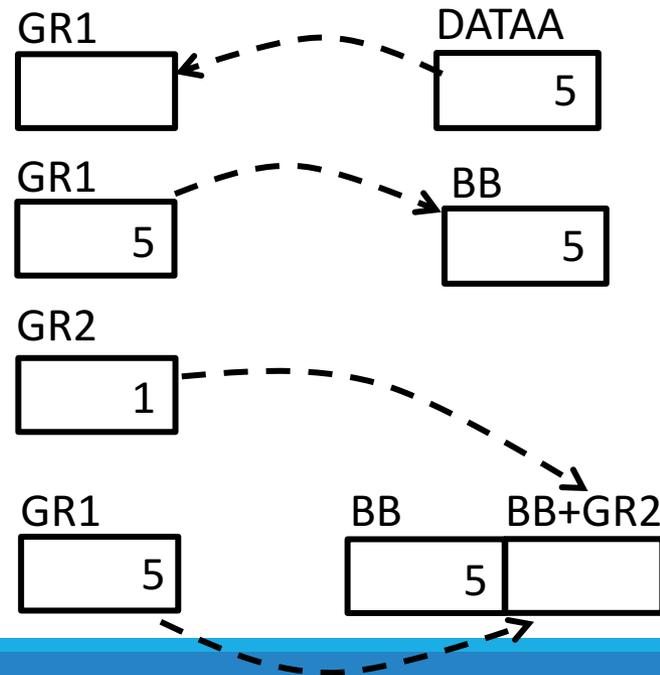
例題4:2進数の印刷

□ ST (Store; ストア) 命令

[ラベル] ST r, adr[,x]

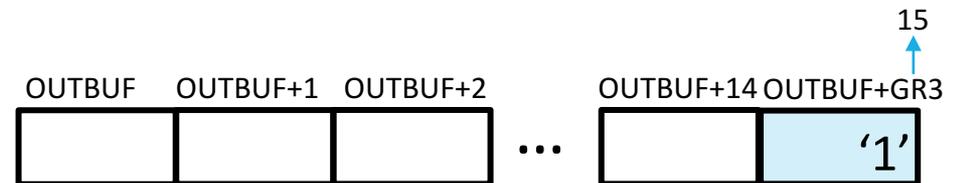
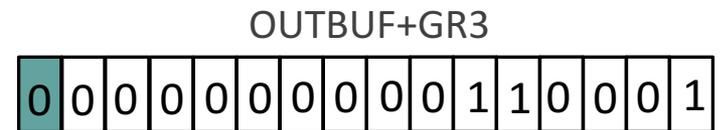
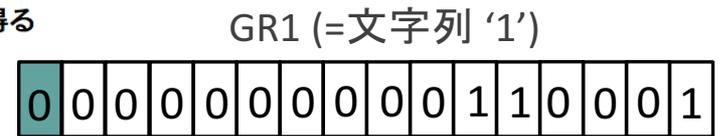
⇒rで指定したレジスタの内容をadr[,x]で指定した主記憶に入れる

PGM	START	
	LD	GR1, DATAA
	ST	GR1, BB
	LAD	GR2, 1
	ST	GR1, BB, GR2
	RET	
DATAA	DC	5
BB	DS	2
	END	



例題4:2進数の印刷

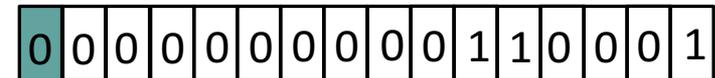
```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```



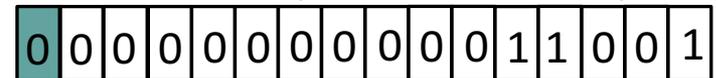
例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```

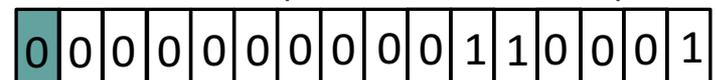
GR1 (= 文字列'1')



GR2 (=DATA02の値, 25)



GR1 (= DATA02の値, 25)



例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO  ;GR3と0を算術比較  
      JPL LOOP    ;GR3 > 0 ならLOOPへ  
      JZE LOOP    ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```

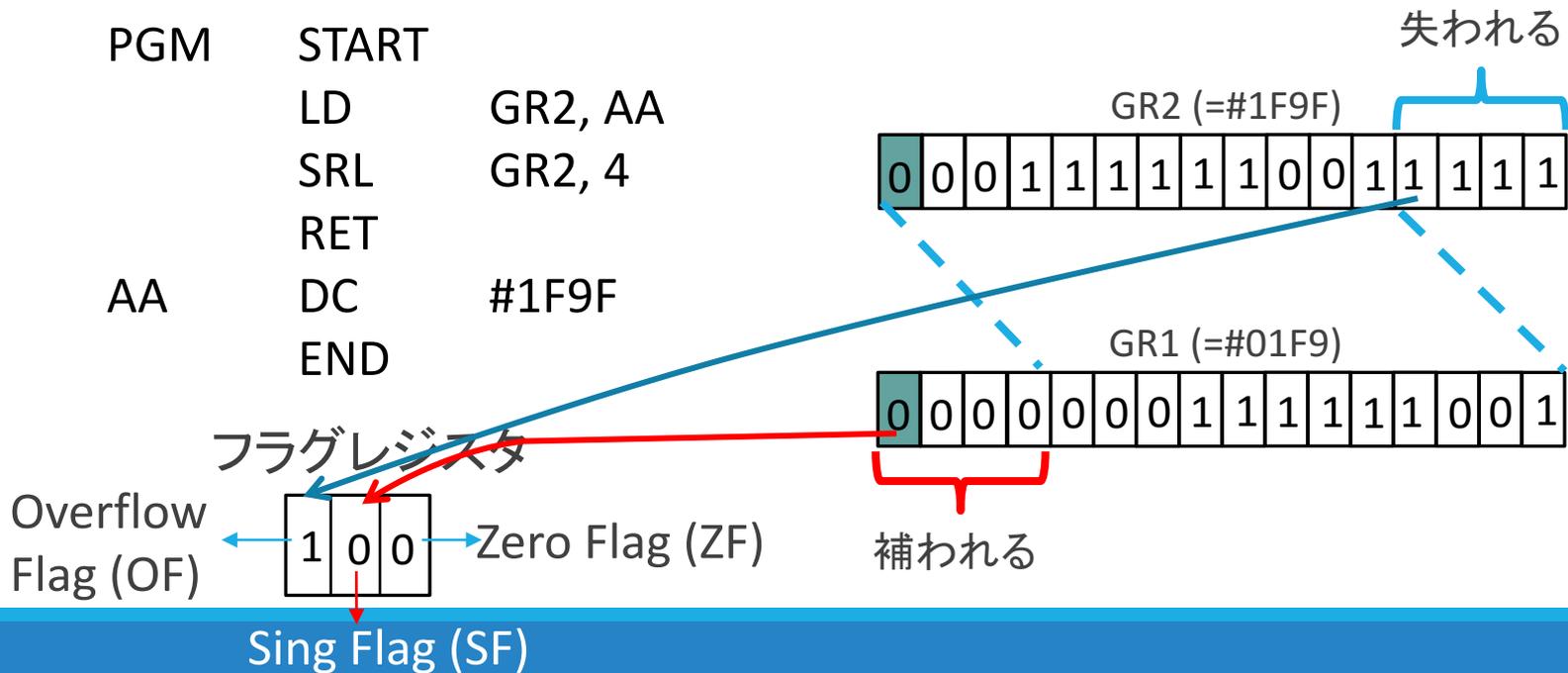


例題4:2進数の印刷

□ SRL (Shift Right Logical; 論理右シフト) 命令

[ラベル] SRL r, adr[,x]

⇒rで指定したレジスタの内容をadr[,x]で指定したビット数だけ右にシフトする (※シフト後空いたビットには0が入る)

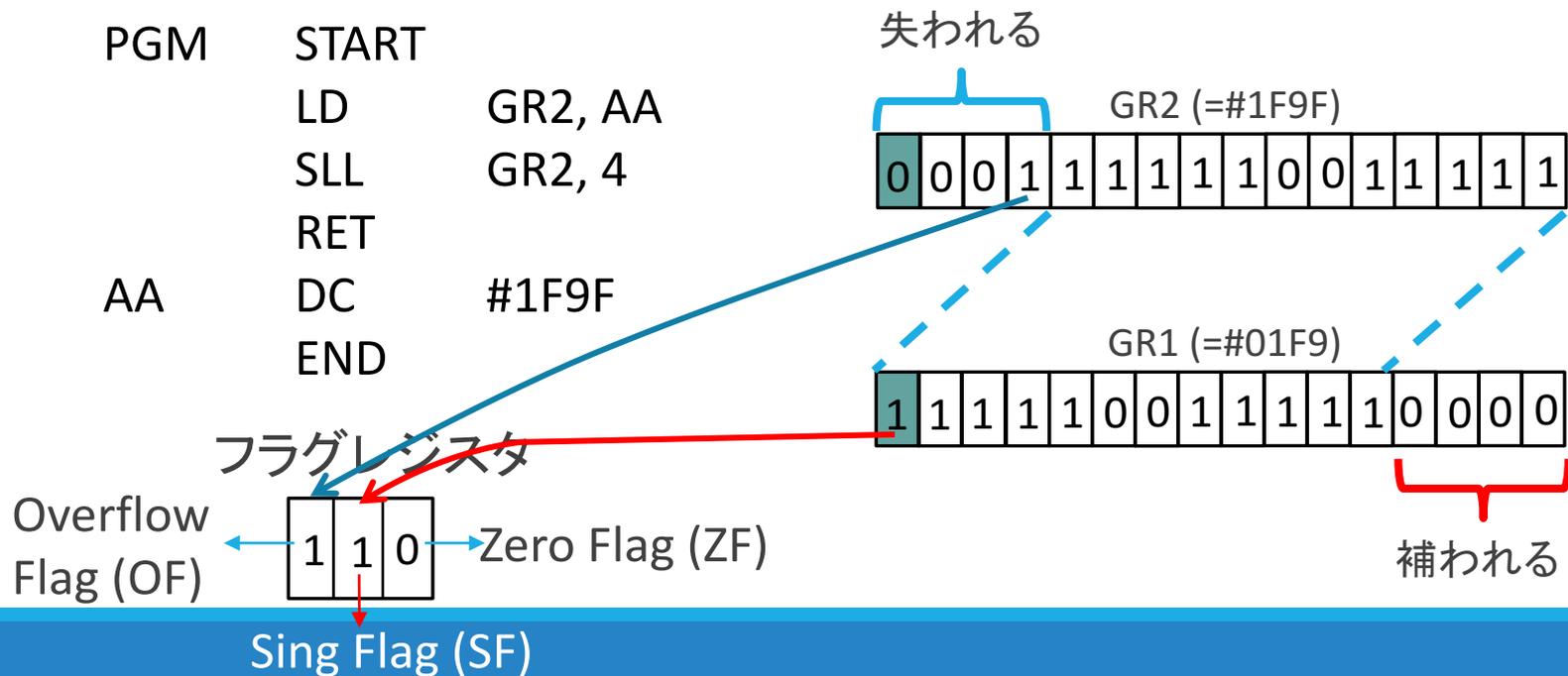


例題4:2進数の印刷

□ SLL (Shift Left Logical; 論理左シフト) 命令

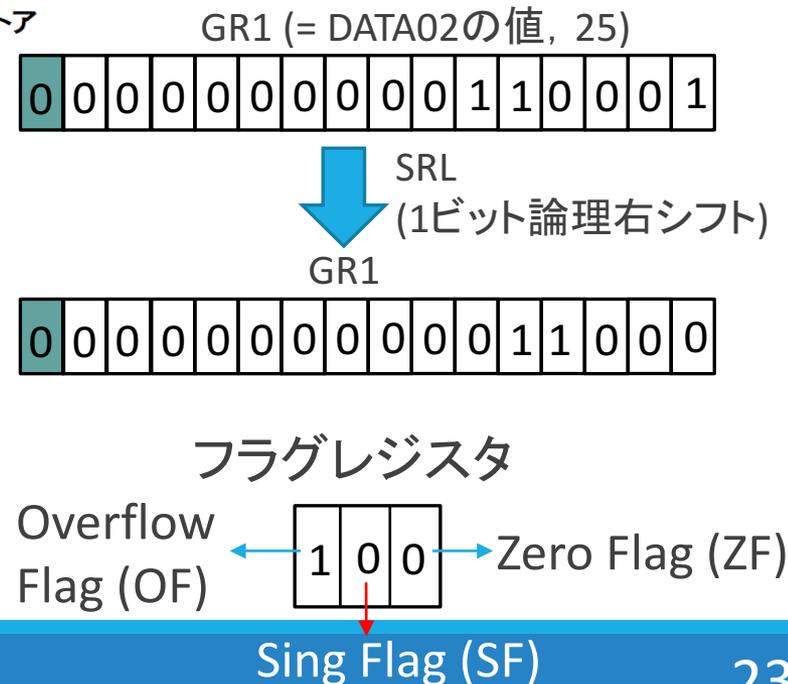
[ラベル] SLL r, adr[,x]

⇒rで指定したレジスタの内容をadr[,x]で指定したビット数だけ左にシフトする (※シフト後空いたビットには0が入る)



例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE   ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2   ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```



例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02 ;GR1 ← (DATA02)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,1      ;GR1を1ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA02 DC 25      ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16      ;16個の領域を確保  
OUTLEN DC 16      ;出力する文字列の長さ16  
END
```

ループカウンタのデクリメント
GR3 = 14

例題4:2進数の印刷

```
-----  
; 例4 2進数の印刷  
-----  
EX4 START  
  LAD GR3,15      ;ループのカウンタの初期化(GR3←15 (16桁の2進数だから))  
  LD  GR1,DATA02  ;GR1 ← (DATA02)  
  LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
        AND  GR1,HEX01 ;GR1と#0001との論理積を取る  
        ADDL GR1,ZONE ;'0'との論理加算により文字コードを得る  
        ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
        LAD GR1,0,GR2 ;GR1を復元  
        SRL GR1,1 ;GR1を1ビット論理右シフト  
        LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
        CPA GR3,ZERO ;GR3と0を算術比較  
        JPL LOOP ;GR3 > 0 ならLOOPへ  
        JZE LOOP ;GR3 = 0 ならLOOPへ  
        OUT OUTBUF,OUTLEN ;文字を出力  
  RET  
DATA02 DC 25 ;←適当に変えてみる  
HEX01  DC #0001  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 16 ;16個の領域を確保  
OUTLEN DC 16 ;出力する文字列の長さ16  
END
```

→ ループカウンタは15～0まで実行

→ ループカウンタがマイナスになったら
(ループが16回実行されたら) 印刷

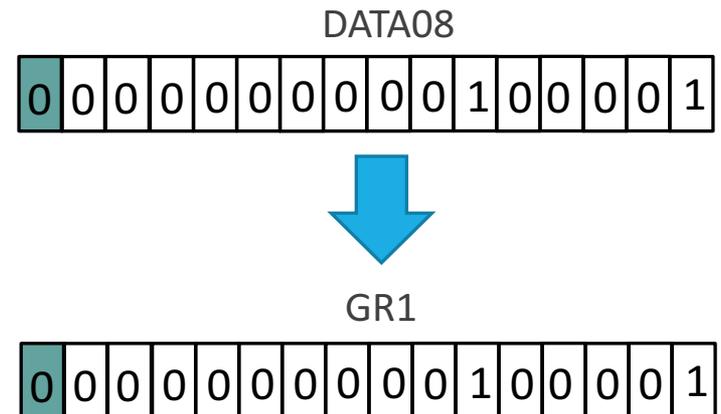
例題5:8進数の印刷

```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,3      ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6      ;6個の領域を確保  
OUTLEN DC 6      ;出力する文字列の長さ6  
END
```

10進数	2進数	8進数	16進数
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

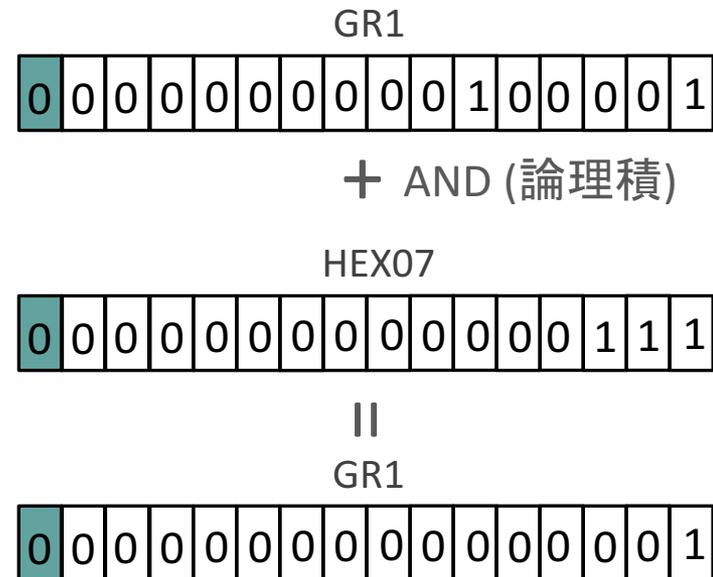
例題5:8進数の印刷

```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,3      ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6      ;6個の領域を確保  
OUTLEN DC 6      ;出力する文字列の長さ6  
END
```



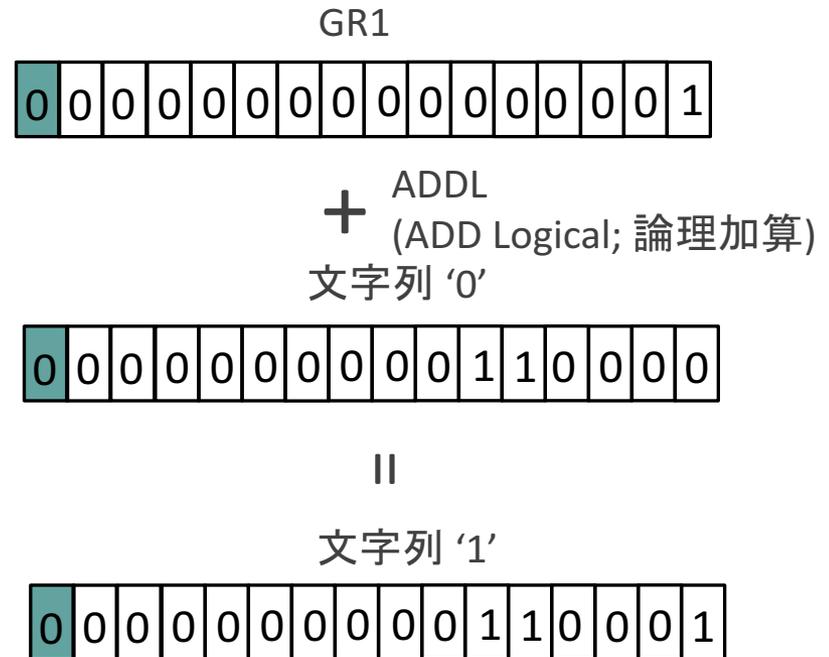
例題5:8進数の印刷

```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,3      ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6      ;6個の領域を確保  
OUTLEN DC 6      ;出力する文字列の長さ6  
END
```



例題5:8進数の印刷

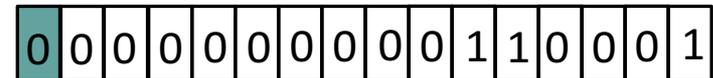
```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2 ;GR1を復元  
      SRL GR1,3    ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO  ;GR3と0を算術比較  
      JPL LOOP    ;GR3 > 0 ならLOOPへ  
      JZE LOOP    ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6      ;6個の領域を確保  
OUTLEN DC 6      ;出力する文字列の長さ6  
END
```



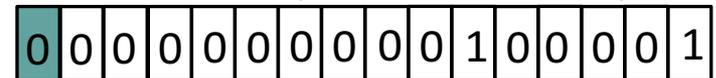
例題5:8進数の印刷

```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,3      ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6      ;6個の領域を確保  
OUTLEN DC 6      ;出力する文字列の長さ6  
END
```

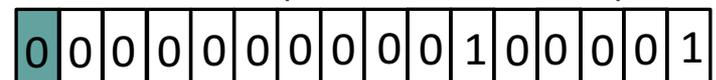
GR1 (= 文字列'1')



GR2 (=DATA08の値, 33)

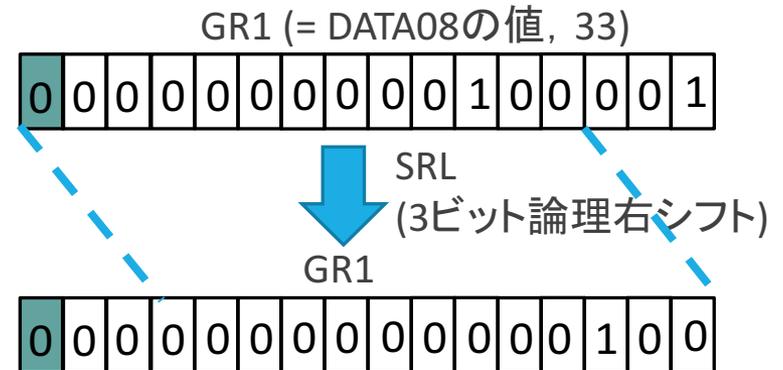


GR1 (= DATA08の値, 33)



例題5:8進数の印刷

```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,3      ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6      ;6個の領域を確保  
OUTLEN DC 6      ;出力する文字列の長さ6  
END
```



例題5:8進数の印刷

```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,3      ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6       ;6個の領域を確保  
OUTLEN DC 6       ;出力する文字列の長さ6  
END
```

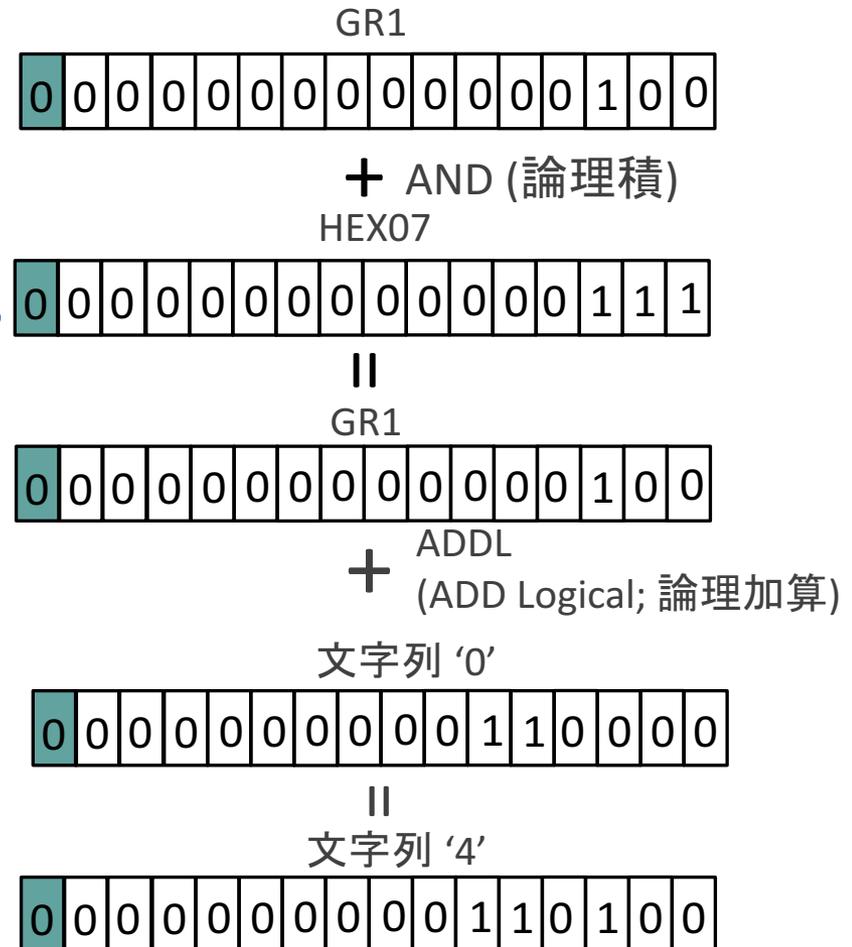
→ ループカウンタのデクリメント
GR3 = 4

例題5:8進数の印刷

```

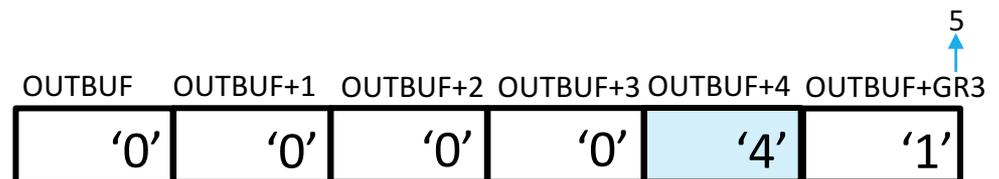
;-----
; 例5 8進数の印刷
;-----
EX5 START
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)
  LD  GR1,DATA08 ;GR1 ← (DATA08)
  LOOP LAD GR2,0,GR1 ;GR1をGR2に退避
    AND  GR1,HEX07 ;GR1と#0007との論理積を取る
    ADDL GR1,ZONE ;'0'との論理加算により文字コードを得る
    ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア
    LAD GR1,0,GR2 ;GR1を復元
    SRL GR1,3 ;GR1を3ビット論理右シフト
    LAD GR3,-1,GR3 ;GR3 ← -1 + GR3
    CPA GR3,ZERO ;GR3と0を算術比較
    JPL LOOP ;GR3 > 0 ならLOOPへ
    JZE LOOP ;GR3 = 0 ならLOOPへ
  OUT OUTBUF,OUTLEN ;文字を出力
  RET
DATA08 DC 33 ;←適当に変えてみる
HEX07 DC #0007
ZONE DC '0'
ZERO DC 0
OUTBUF DS 6 ;6個の領域を確保
OUTLEN DC 6 ;出力する文字列の長さ6
END

```



例題5:8進数の印刷

```
-----  
; 例5 8進数の印刷  
-----  
EX5 START  
  LAD GR3,5      ;ループのカウンタの初期化(GR3←5)  
  LD  GR1,DATA08 ;GR1 ← (DATA08)  
LOOP  LAD GR2,0,GR1 ;GR1をGR2に退避  
      AND  GR1,HEX07 ;GR1と#0007との論理積を取る  
      ADDL GR1,ZONE  ;'0'との論理加算により文字コードを得る  
      ST  GR1,OUTBUF,GR3 ;GR1を(OUTBUF+GR3)番地にストア  
      LAD GR1,0,GR2  ;GR1を復元  
      SRL GR1,3      ;GR1を3ビット論理右シフト  
      LAD GR3,-1,GR3 ;GR3 ← -1 + GR3  
      CPA GR3,ZERO   ;GR3と0を算術比較  
      JPL LOOP      ;GR3 > 0 ならLOOPへ  
      JZE LOOP      ;GR3 = 0 ならLOOPへ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA08 DC 33      ;←適当に変えてみる  
HEX07  DC #0007  
ZONE   DC '0'  
ZERO   DC 0  
OUTBUF DS 6      ;6個の領域を確保  
OUTLEN DC 6      ;出力する文字列の長さ6  
END
```



5

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST   GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD  GR3,1,GR3    ;GR3 ← 1 + GR3  
    CPA  GR3,F5       ;GR3と5を算術比較  
    JMI  LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

例題6:10進数の印刷

□処理の流れ

1. GR1 (=DATA) から10000を何回引くと負になるかを求める
⇒ n回で負になったら, nが万の位の値
2. nを文字列'0'と加算することで文字列に変換
3. GR1は「-????」という4桁以下の負の値になっている
⇒ 10000を足して4桁の正の数にする
4. GR1から引く数を1000に変え, nを求める
5. nを文字列に変換
6. GR1に1000を足して3桁以下の正の数にする
7. ...以下, 100,10,1とGR1から引く数を変えながら繰り返し

例題の補足ページ:

<http://www.ced.is.utsunomiya-u.ac.jp/lecture/2017/prog/p1/kadai4/hosoku.htm>

例題6:10進数の印刷

□ 処理の流れ

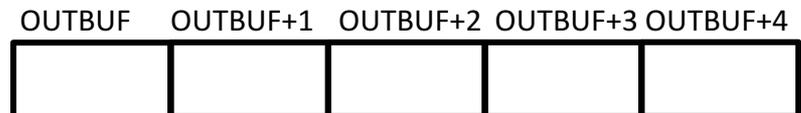
□ 万の位の場合

```
GR2 = 0;
DATA10 = 23542;
GR1 = DATA10;
while (1){
GR1 = GR1 - 10000;
//1回目 GR1 = 13542
//2回目 GR1 = 3542
//3回目 GR1 = -6458
if (GR1 < 0) break;
GR2 = GR2+1;    GR2 =2 ← 万の位: 2
}

```

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
      LD  GR1,DATA10      ;データをGR1にセット  
      LAD GR3,0           ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1        ;商の設定  
LOOP2 LAD  GR2,1,GR2     ;GR2 ← 1 + GR2  
      SUBA GR1,C10X,GR3  ;GR1 ← (GR1)-(C10X+GR3)  
      JPL  LOOP2         ;減算結果が >0 ならばLOOP2へ  
      JZE  LOOP2         ;減算結果が =0 ならばLOOP2へ  
      ADDA GR1,C10X,GR3  ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
      ADDL GR2,ZONE      ;GR2と'0'との論理加算により文字コードを得る  
      ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
      LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
      CPA GR3,F5        ;GR3と5を算術比較  
      JMI LOOP1         ;GR3 < 0 ならLOOP1へ  
      OUT OUTBUF,OUTLEN ;文字を出力  
      RET  
DATA10 DC  23542      ;← 適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```



例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
  LD  GR1,DATA10      ;データをGR1にセット  
  LAD  GR3,0          ;文字列本数  
  LOOP1 LAD  GR2,-1    ;桁の設定  
  LOOP2 LAD  GR2,1,GR2  
  SUBA GR1,C10X,GR3   ;GR1 ← (GR1)-(C10X+GR3)  
  JPL  LOOP2          ;減算結果が >0 ならばLOOP2へ  
  JZE  LOOP2          ;減算結果が =0 ならばLOOP2へ  
  ADDA GR1,C10X,GR3   ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
  ADDL GR2,ZONE       ;GR2と'0'との論理加算により文字コードを得る  
  ST   GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
  LAD  GR3,1,GR3      ;GR3 ← 1 + GR3  
  CPA  GR3,F5         ;GR3と5を算術比較  
  JMI  LOOP1          ;GR3 < 0 ならLOOP1へ  
  OUT  OUTBUF,OUTLEN  ;文字を出力  
  RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

計算する位が変わる度にGR2を-1に初期化→LOOP2でインクリメントしてGR2は0になる

LOOP2で減算する度にGR2をインクリメントして各桁の値を求める

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
SUBA  GR1,C10X,GR3   ;GR1 ← (GR1)-(C10X+GR3)  
JPL  LOOP2          ;減算結果が >0 ならばLOOP2へ  
JZE  LOOP2          ;減算結果が =0 ならばLOOP2へ  
ADDA  GR1,C10X,GR3   ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
ADDL  GR2,ZONE       ;GR2と'0'との論理加算により文字コードを得る  
ST    GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
LAD   GR3,1,GR3      ;GR3 ← 1 + GR3  
CPA   GR3,F5         ;GR3と5を算術比較  
JMI  LOOP1          ;GR3 < 0 ならLOOP1へ  
OUT   OUTBUF,OUTLEN  ;文字を出力  
RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

例題2:大小判定

□ SUBA (SUBtract Arithmetic;算術計算) 命令

[ラベル] SUBA r1, r2

⇒ r1で指定したレジスタの内容からr2で指定したレジスタの内容を符号付きの数として減算, 結果をr1に入れる

[ラベル] SUBA r, adr[,x]

⇒ rで指定したレジスタの内容からadr[,x] で指定した主記憶の内容を符号付きの数として減算, 結果をrに入れる

例題6:10進数の印刷

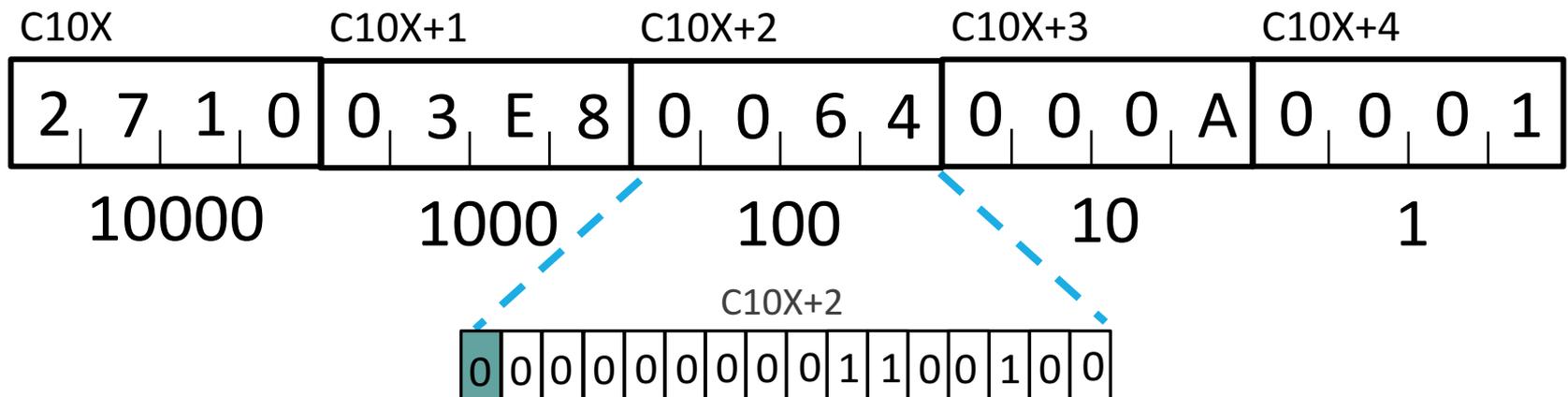
```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
SUBA  GR1,C10X,GR3   ;GR1 ← (GR1)-(C10X+GR3)  
JPL  LOOP2           ;減算結果が >0 ならばLOOP2へ  
JZE  LOOP2           ;減算結果が =0 ならばLOOP2へ  
ADDA  GR1,C10X,GR3   ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
ADDL  GR2,ZONE       ;GR2と'0'との論理加算により文字コードを得る  
ST    GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
LAD  GR3,1,GR3       ;GR3 ← 1 + GR3  
CPA  GR3,F5          ;GR3と5を算術比較  
JMI  LOOP1           ;GR3 < 0 ならLOOP1へ  
OUT  OUTBUF,OUTLEN   ;文字を出力  
RET  
DATA10 DC  23542     ;←適当に変えてみる  
C10X  DC  10000,1000,100,10,1 ;連続領域  
ZONE  DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5    DC  5  
END
```

例題2:大小判定

□ DC (Define Constant) 命令

□ 複数の定数を記述する場合 → 定数をカンマで区切る

C10X DC 10000, 1000, 100, 10, 1



例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0          ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1      ;商の設定  
LOOP2 LAD  GR2,1,GR2   ;GR2 ← 1 + GR2  
SUBA  GR1,C10X,GR3    ;GR1 ← (GR1)-(C10X+GR3) → 今, GR3 = 0 すなわちC10Xの値  
JPL  LOOP2            ;減算結果が >0 ならばLOOP2へ  
JZE  LOOP2            ;減算結果が =0 ならばLOOP2へ  
ADDA  GR1,C10X,GR3    ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
ADDL  GR2,ZONE        ;GR2と'0'との論理加算により文字コードを得る  
ST  GR2,OUTBUF,GR3    ;GR2を(OUTBUF+GR3)番地にストア  
LAD  GR3,1,GR3        ;GR3 ← 1 + GR3  
CPA  GR3,F5           ;GR3と5を算術比較  
JMI  LOOP1            ;GR3 < 0 ならLOOP1へ  
OUT  OUTBUF,OUTLEN    ;文字を出力  
RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X  DC  10000,1000,100,10,1 ;連続領域  
ZONE  DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5    DC  5  
END
```

※実際の演算は16ビットの2進数

$GR1 - C10X = 23542 - 10000$

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA  GR3,F5       ;GR3と5を算術比較  
    JMI  LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

→ GR2 = 1

→ GR1 = 13542

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
SUBA  GR1,C10X,GR3   ;GR1 ← (GR1)-(C10X+GR3) → GR1 - C10X = 13542 - 10000  
JPL  LOOP2           ;減算結果が >0 ならばLOOP2へ  
JZE  LOOP2           ;減算結果が =0 ならばLOOP2へ  
ADDA  GR1,C10X,GR3   ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
ADDL  GR2,ZONE       ;GR2と'0'との論理加算により文字コードを得る  
ST  GR2,OUTBUF,GR3   ;GR2を(OUTBUF+GR3)番地にストア  
LAD  GR3,1,GR3       ;GR3 ← 1 + GR3  
CPA  GR3,F5          ;GR3と5を算術比較  
JMI  LOOP1           ;GR3 < 0 ならLOOP1へ  
OUT  OUTBUF,OUTLEN   ;文字を出力  
RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X  DC  10000,1000,100,10,1 ;連続領域  
ZONE  DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5    DC  5  
END
```

→ GR1 - C10X = 13542 - 10000
GR1 = 3542

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA  GR3,F5       ;GR3と5を算術比較  
    JMI  LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

→ GR2 = 2

→ GR1 = 3542

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10      ;データをGR1にセット  
    LAD GR3,0           ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1      ;商の設定  
LOOP2 LAD  GR2,1,GR2   ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3  ;GR1 ← (GR1)-(C10X+GR3) → GR1 - C10X = 3542 - 10000  
    JPL  LOOP2         ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2         ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3  ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE      ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3      ;GR3 ← 1 + GR3  
    CPA GR3,F5         ;GR3と5を算術比較  
    JMI LOOP1         ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

→ GR1 - C10X = 3542 - 10000
GR1 = -6458

↓
GR1 + C10X = -6458 + 10000
GR1 = 3542

例題6:10進数の印刷

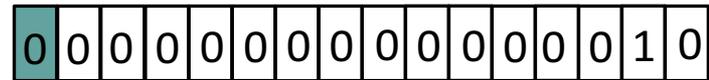
```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0          ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST   GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD  GR3,1,GR3    ;GR3 ← 1 + GR3  
    CPA  GR3,F5        ;GR3と5を算術比較  
    JMI  LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

➡ GR2 = 2

例題6:10進数の印刷

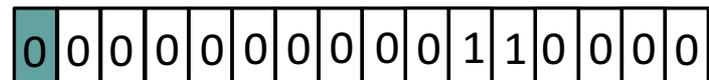
```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10      ;データをGR1にセット  
    LAD GR3,0           ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1       ;商の設定  
LOOP2 LAD  GR2,1,GR2    ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3   ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2         ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2         ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3   ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE       ;GR2と'0'との論理加算により文字コードを得る  
    ST   GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD  GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA  GR3,F5        ;GR3と5を算術比較  
    JMI  LOOP1         ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542      ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

GR2 = 2



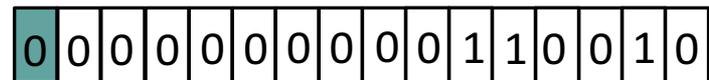
+ ADDL
(ADD Logical; 論理加算)

文字列 '0'



||

文字列 '2'

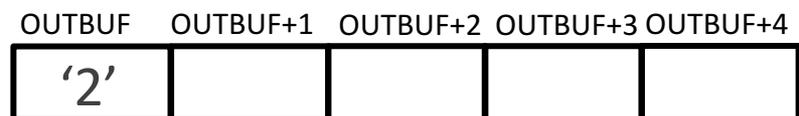
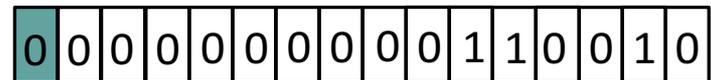


例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA  GR3,F5       ;GR3と5を算術比較  
    JMI  LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542     ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

GR3 = 0

GR2 (=文字列 '2')



例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA  GR3,F5       ;GR3と5を算術比較  
    JMI  LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542     ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

→ GR3 = 1

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0          ;文字列を数えるカウンタ  
LOOP1 LAD GR2,-1      ;商の設定  
LOOP2 LAD GR2,1,GR2   ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL LOOP2         ;減算結果が >0 ならばLOOP2へ  
    JZE LOOP2         ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA GR3,F5        ;GR3と5を算術比較  
    JMI LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC 23542      ;←適当に変えてみる  
C10X   DC 10000,1000,100,10,1 ;連続領域  
ZONE   DC '0'  
OUTBUF DS 5  
OUTLEN DC 5  
F5     DC 5  
END
```

→ F5 = 5
G3 < F5 なのでLOOP1へ

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD GR2,-1     ;商の設定  
LOOP2 LAD GR2,1,GR2  ;GR2 ← 1 + GR2  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA  GR3,F5       ;GR3と5を算術比較  
    JMI LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC 23542      ;←適当に変えてみる  
C10X DC 10000,1000,100,10,1 ;連続領域  
ZONE DC '0'  
OUTBUF DS 5  
OUTLEN DC 5  
F5 DC 5  
END
```

→ GR2 = -1

例題6:10進数の印刷

```
-----  
; 例6 10進数の印刷  
-----  
EX6  START  
    LD  GR1,DATA10    ;データをGR1にセット  
    LAD GR3,0         ;文字列を数えるカウンタ  
LOOP1 LAD  GR2,-1     ;商の設定  
LOOP2 LAD  GR2,1,GR2  ;GR2 ← 1 + GR2 → GR2 = 0  
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)  
    JPL  LOOP2        ;減算結果が >0 ならばLOOP2へ  
    JZE  LOOP2        ;減算結果が =0 ならばLOOP2へ  
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算  
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る  
    ST  GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア  
    LAD GR3,1,GR3     ;GR3 ← 1 + GR3  
    CPA  GR3,F5       ;GR3と5を算術比較  
    JMI  LOOP1        ;GR3 < 0 ならLOOP1へ  
    OUT  OUTBUF,OUTLEN ;文字を出力  
    RET  
DATA10 DC  23542     ;←適当に変えてみる  
C10X   DC  10000,1000,100,10,1 ;連続領域  
ZONE   DC  '0'  
OUTBUF DS  5  
OUTLEN DC  5  
F5     DC  5  
END
```

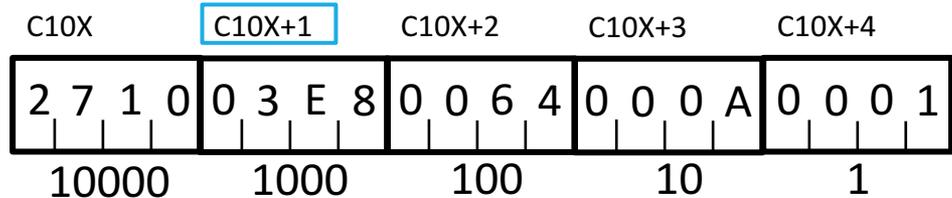
例題6:10進数の印刷

```

;-----
; 例6 10進数の印刷
;-----
EX6  START
    LD  GR1,DATA10    ;データをGR1にセット
    LAD GR3,0         ;文字列を数えるカウンタ
LOOP1 LAD  GR2,-1     ;商の設定
LOOP2 LAD  GR2,1,GR2 ;GR2 ← 1 + GR2
    SUBA GR1,C10X,GR3 ;GR1 ← (GR1)-(C10X+GR3)
    JPL  LOOP2       ;減算結果が >0 ならばLOOP2へ
    JZE  LOOP2       ;減算結果が =0 ならばLOOP2へ
    ADDA GR1,C10X,GR3 ;GR1が負になっているので(C10X + GR3)番地のデータを加算
    ADDL GR2,ZONE     ;GR2と'0'との論理加算により文字コードを得る
    ST   GR2,OUTBUF,GR3 ;GR2を(OUTBUF+GR3)番地にストア
    LAD  GR3,1,GR3    ;GR3 ← 1 + GR3
    CPA  GR3,F5       ;GR3と5を算術比較
    JMI  LOOP1       ;GR3 < 0 ならLOOP1へ
    OUT  OUTBUF,OUTLEN ;文字を出力
    RET
DATA10 DC  23542    ;←適当に変えてみる
C10X   DC  10000,1000,100,10,1 ;連続領域
ZONE   DC  '0'
OUTBUF DS  5
OUTLEN DC  5
F5     DC  5
END

```


 $GR1 = 3542, GR3 = 1$
 $(GR1) - (C10X+1) = 3542 - 1000$

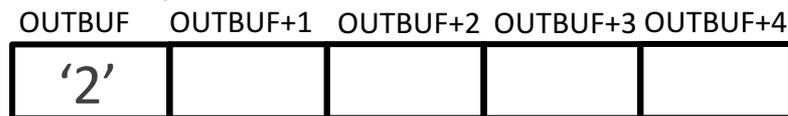


例題6:10進数の印刷

□各レジスタの値の変化

□万の位

ループ回数	GR1	GR2	GR3	C10X+GR3
0	23542	-1	0	
1	23542	0	0	10000
2	13542	1	0	10000
3	3542	2	0	10000
4	-6458	2	0	10000
4	3542	2	1	1000

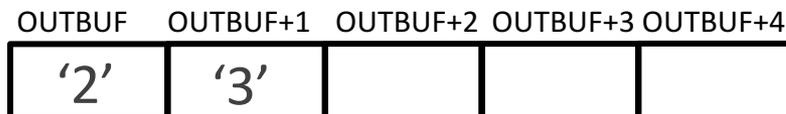


例題6:10進数の印刷

□各レジスタの値の変化

□千の位

ループ回数	GR1	GR2	GR3	C10X+GR3
0	3542	-1	1	
1	3542	0	1	1000
2	2542	1	1	1000
3	1542	2	1	1000
4	542	3	1	1000
5	-458	3	1	1000
5	542	3	2	100



例題6:10進数の印刷

□各レジスタの値の変化

□百の位

ループ回数	GR1	GR2	GR3	C10X+GR3
0	542	-1	2	
1	542	0	2	100
2	442	1	2	100
3	342	2	2	100
4	242	3	2	100
5	142	4	2	100
6	42	5	2	100
7	-58	5	2	100
7	42	5	3	10

OUTBUF OUTBUF+1 OUTBUF+2 OUTBUF+3 OUTBUF+4

'2'	'3'	'5'		
-----	-----	-----	--	--

例題6:10進数の印刷

□各レジスタの値の変化

□十の位

ループ回数	GR1	GR2	GR3	C10X+GR3
0	42	-1	3	
1	42	0	3	10
2	32	1	3	10
3	22	2	3	10
4	12	3	3	10
5	2	4	3	10
6	-8	4	3	10
6	2	4	4	1

OUTBUF OUTBUF+1 OUTBUF+2 OUTBUF+3 OUTBUF+4

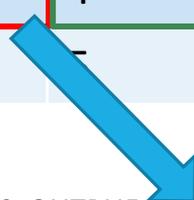
'2' '3' '5' '4'

例題6:10進数の印刷

□各レジスタの値の変化

□一の位

ループ回数	GR1	GR2	GR3	C10X+GR3
0	2	-1	4	
1	2	0	4	1
2	1	1	4	1
3	0	2	4	1
4	-1	2	4	1
4	-1	2	4	1



OUTBUF	OUTBUF+1	OUTBUF+2	OUTBUF+3	OUTBUF+4
'2'	'3'	'5'	'4'	'2'

課題

□課題5

□例題4, 5を参考に

⇒ マスク処理, シフト演算, 文字列変換

□講義ページの**ヒント**も要確認

□課題6

1. 文字コードの説明は今日の講義資料も参考に
2. 3桁毎の判別をどのように行うか考える
3. 例題6を参考に