プログラミング演習１
課題4：アセンブリ言語

2進数の印刷

8進数の印刷

10進数の印刷

本日の授業内容

1. 例題４：2進数の印刷
2. 例題５：8進数の印刷
3. 例題６：10進数の印刷
4. 練習問題

例題4の処理の流れ

1. ループカウンタ設定
2. LSBが1 or 0の判定
3. 文字コード'1' or '0'を計算
4. LSBの表示位置に文字コードを格納
5. 印刷対象を1bit 右シフト
6. 表示位置を1つ左に
7. 1へ戻り，繰り返し

```
;------------------------
; 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
```

LD命令とLAD命令

- [ラベル] LD r1, r2 or adr[,x]
  r2または実効アドレスの内容をr1へロード
  例）LD GR1,DATA02: GR1にDATA02の内容（25）がロード

- [ラベル] LAD r, adr[,x]
  実効アドレスそのものをr1へロード
  例1) LAD GR3,15: GR3にアドレス15をロード
  例2) LAD GR3,-1,GR3: GR3に-1番地+GR3がGR3にロード
  → ループカウンタのデクリメント

LSBが1 or 0の判定

- AND GR1,HEX01 (マスク処理)
  - GR1: 0000 0000 0001 1001 $(25)_{10}$
  - HEX01: #0001 $(0000\ 0000\ 0000\ 0001)_2$

  - GR1: 0000 0000 0000 0001
- つまり
  - LSBは1と判定

文字コードの計算

- 文字コード'0'を基準にするとわかりやすい
  - '0': #30: $(0011\ 0000)_2$
  - '1': #31: $(0011\ 0001)_2$

    '0' + 1で実現(ADDL GR1,ZONE)

    判定結果, '0'
- '9'までは簡単
- 'A'を実現するには・・・
  - #41だから

    '0' + #10 + 1

上位４ビット →

下位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	゜		

ADDL命令(論理加算命令)

- [ラベル] ADDL r1, r2(adr[,x])
  - r1の内容とr2(もしくは実効アドレス)の内容を
    符号なし数値として加算し，r1へ格納
  - マイナスの数は扱えない（マイナスになるとOFフラグが1に）
- (ちなみに)ADDA命令は，算術加算命令
  - [ラベル] ADDA r1, r2(adr[,x])
    r1の内容とr2(または実効アドレス)の内容を
    符号付き数値として加算し，r1へ格納
  - マイナスの数を扱える

ST命令

- [ラベル] ST r, adr[,x]

  rの内容をadr[,x]番地に格納

  - ST GR1,OUTBUF,GR3; GR1を(OUTBUF+GR3)番地に格納
    - GR1: 文字コード ‘0’ or ‘1’
    - OUTBUF: 16個の領域(0から15まで)
    - GR3: ループカウンタ(初期値は15)
  - (OUTBUF+GR3)の領域に文字コードが格納される

格納後，GR3をデクリメントすれば，
格納先が変えられる

OB	OB+1	OB+2	・・・	OB+13	OB+14	OB+15
			・・・			‘0’ or ‘1’

次の判定への準備

- LAD GR1, 0, GR2
  - 判定処理によってGR1の内容は変化してしまう
  - GR2に退避した処理前の内容をリロード
- SRL GR1, 1: 論理右シフト
  - 次に判定するビットをLSBにするために
- LAD GR3,-1,GR3: デクリメント
  - 次の判定結果の格納先OUTBUF+(GR3-1)のため

論理シフト(SLL,SRL)と算術シフト(SLA,SRA)

- 論理シフト
  - [ラベル] SRL(SLL) r, adr[,x]
    rの内容をadr[,x]ビット右（左）にシフト（符号ビット含む）
- 算術シフト
  - [ラベル] SRA(SLA) r, adr[,x]
    rの内容をadr[,x]ビット右（左）にシフト（符号ビット除く）
- 例) SRL (SRA) r(=#FF9F), 4
  - SRL: 1111 1111 1001, 1111 → 0000 1111 1111 1001 (0で埋める)
  - SRA: 1111 1111 1001, 1111 → 1111 1111 1111 1001 (符号ビットで埋める)

終了判定

- 全ビットを判定できたら
  - CPA GR3, ZERO
    GR3(ループカウンタ) < 0かどうかを判定
    （GR3＜0: 最上位ビット格納済み）
  - GR3>=0ならラベル「LOOP」へ戻り，判定処理
  - GR<0なら，OUT OUTBUF OUTLENで出力し，終了

例題５：8進数の印刷

- 例題4を8進数に拡張

```
;-----------------------
; 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
```

8進数について

- 0~7で表記 → 3bitで表す(マスクは#0007になり, 3bit右シフトする)
- CASL2のレジスタは16bit → #FFFFが最大値(符号なし)
- #FFFFを8進数にすると
  - #FFFF = $(1111\ 1111\ 1111\ 1111)_2$
    = $(1\ 111\ 111\ 111\ 111\ 111)_2$
    = $(177777)_8$
- なので
  - ループカウンタ(GR3)は5から0まで
  - 印刷用の領域も6個（6桁）

例題6: 10進数の印刷

- 例題4,5とはちょっと違う
  1. GR1から10000を何回引くと負の数になるか（n回で負になれば万の位はn）
  2. 手順1でGR1は負の数になるので，GR1+10000し，4桁の正の数にする
  3. n ＋’0’でnの文字コードを得る
  - ➢GR1÷10000=n　あまり4桁の正の数

  4. 手順1~3を1000, 100, 10, 1と変えて実行（ループ）

例題6:10進数の印刷

- ソースコード

手順2

```
;----------------------
; 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
```

手順1

手順3

手順1をCで書くと(万の位)

```
1.  GR2 = 0;
2.  DATA10 = 23542;
3.  GR1 = DATA10;
4.  while(1){
5.      GR1 -= 10000;
6.
7.      if(GR1<0) break;
8.      GR2++;
9.  }
```

10進数の印刷で気を付ける点

- ループカウンタ
  - LAD GR3, 15(6) → LAD GR3, 0
    (これまでは下位ビットから求めていたが，10進数では大きい位から計算)
  - デクリメント
    LAD GR3,-1,GR3
    インクリメント
    LAD GR3,1,GR3
  - ループカウンタが2つある
    - 文字列カウント用（文字コード格納用）
    - 割り算用

例題6の2重ループ

- GR2の初期値が-1
  - すぐにインクリメントされる
    （実質0が初期値）
- LOOP2
  - 10^nとの割り算
  - GR1が負の数になるまで
- LOOP1
  - 商の格納
  - 次の割る数を設定

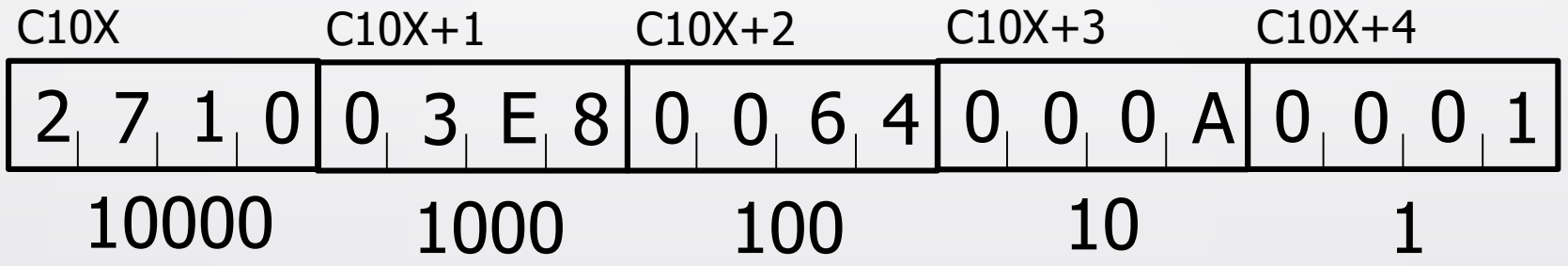
```
;---------------------
; 例6 10進数の印刷
;---------------------
EX6     START
        LD      GR1,DATA10      ;データをGR1にセット
        LAD     GR3,0           ;文字列を数えるカウンタ
LOOP1   LAD     GR2,-1          ;商の設定
LOOP2   LAD     GR2,1,GR2       ;GR2 ← 1 + GR2
        SUBA    GR1,C10X,GR3    ;GR1 ← (GR1)-(C10X+GR3)
        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
```

SUBA(算術減算)とSUBL(論理減算)

- [ラベル] SUBA r1, r2 or adr[,x]
  - r1の内容からr2 or adr[,x]の内容を減算（符号付き数として認識）
  - SUBA r1, 1, r2: r1から(r2+1)番地の値を減算
  - マイナスの数扱える
- [ラベル] SUBL r1, r2 or adr[,x]
  - r1の内容からr2 or adr[,x]の内容を減算（符号なし数として認識）
  - SUBL r1, 1, r2: r1から(r2+1)番地の値を減算
  - マイナスの数扱えない（マイナスになるとOFフラグが1に）

連続領域

- C10X DC 10000,1000,100,10,1

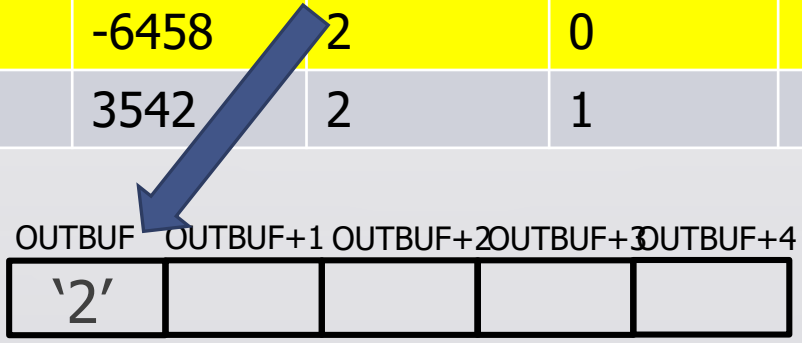


- SUBA GR1,C10X,GR3
  - GR3（ループカウンタ）によってGR1から引く数が変わる

例題6:10進数の印刷

- 各レジスタの値の変化
  - 万の位

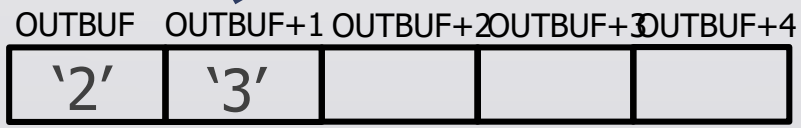
ループ回数	GR1	GR2	GR3	C10X+GR3
0	23542	-1	0	
1	13542	1	0	10000
2	3542	2	0	10000
3	-6458	2	0	10000
4	3542	2	1	1000



例題6:10進数の印刷

- 各レジスタの値の変化
  - 千の位

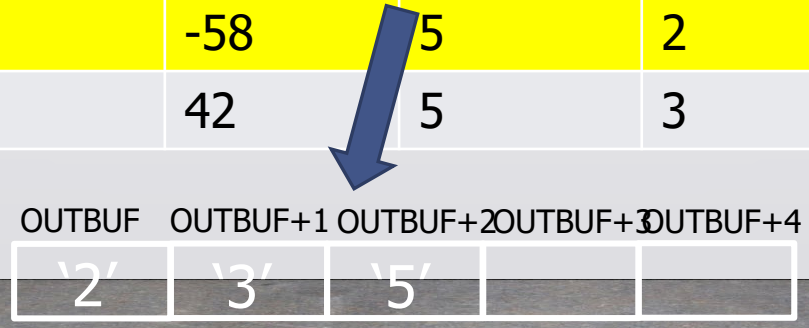
ループ回数	GR1	GR2	GR3	C10X+GR3
0	3542	-1	1	
1	2542	1	1	1000
2	1542	2	1	1000
3	542	3	1	1000
4	-458	3	1	1000
4	542	3	2	100



例題6:10進数の印刷

- 各レジスタの値の変化
  - 百の位

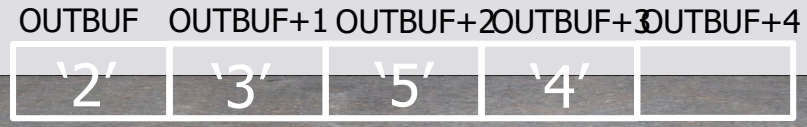
ループ回数	GR1	GR2	GR3	C10X+GR3
0	542	-1	2	
1	442	1	2	100
2	342	2	2	100
3	242	3	2	100
4	142	4	2	100
5	42	5	2	100
6	-58	5	2	100
6	42	5	3	10



例題6:10進数の印刷

- 各レジスタの値の変化
  - 十の位

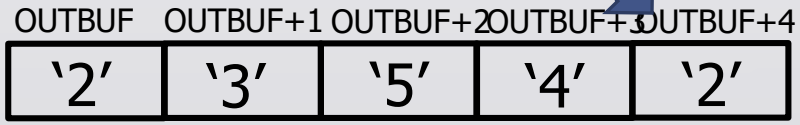
ループ回数	GR1	GR2	GR3	C10X+GR3
0	42	-1	3	
1	32	1	3	10
2	22	2	3	10
3	12	3	3	10
4	2	4	3	10
5	-8	4	3	10
6	2	4	4	1



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	1	1	4	1
2	0	2	4	1
3	-1	2	4	1
4	-1	2	5	



練習課題

1. 16進数の印刷
   - 例題4, 5を参考にすれば簡単

2. 桁区切り
   - 12345 → 12,345
   - ヒント：カンマ(,)も1文字として扱う
   - つまり，用意する領域は6