

2015年度コンピュータアーキテクチャ論演習		第1回 レポート	
学籍番号		氏名	

レポートは手書きで作成する必要はない。手書きの場合には丁寧な文字で記述すること。

問1 以下の項目を自分の言葉で説明せよ。図等を用いて十分な説明をすること。

- (a) MIPSアーキテクチャの設計原則とは？(4個答えなさい)
- (b) 典型的なプロセッサの命令実行は、5つのステージからなる。それぞれを簡単に説明しなさい。
- (c) big endianとlittle endianの違いについて説明しなさい。
- (d) MIPSアーキテクチャにおける、スタックポインタの利用方法を説明しなさい。
- (e) 現在広く利用されているMIPS以外のプロセッサアーキテクチャを3種類調べ、それらの特徴について簡単に述べなさい。

問2

MIPSアーキテクチャにおける命令実行では、1命令ごとにPC(プログラムカウンタ)が4ずつインクリメントされる。ただし、分岐命令やジャンプ命令の時にはPCは通常とは異なる増減をする。本演習で利用可能な命令のうち、この場合に対応するのはBEQ命令、J命令、JR命令、JAL命令である。

- (1) この4命令それぞれの場合に、分岐またはジャンプ可能なアドレスの範囲を答えなさい。
- (2) 下記のコード列の先頭アドレスが0x30048にあるとする。BEQ命令とJ命令を16進数で表しなさい。

```

BEGIN:   addi $t0, $zero, 0
         addi $t1, $zero, 1
LOOP:    slt $t2, $a0, $t1
         beq $t2, $zero, FINISH
         add $t0, $t0, $t1
         addi $t1, $t1, 2
         j  LOOP
FINISH:  add $v0, $t0, $zero

```

(3) 下記の非常に長いコードの一部分を考える。左側の数値は命令のアドレスを意味する。この部分を含んだプログラムをアセンブルしようとしたがアセンブルエラーが発生した。なぜエラーが発生したのか、またどのようにプログラムを修正すればエラーが出なくなるか説明せよ。ただし、「there」ラベルの命令のアドレスは変更できないとする。

```

0x0040005C          beq $s0, $s2, there
.....(省略)
0x00444444  there:  add $s0, $s0, 1

```

問3 以下の例にしたがって、与えられたCプログラムを、(1)疑似命令と、(2) MIPS アセンブリプログラムの両方に変換せよ。各変数はメモリ上に配置されているとする。

例 :  $f = g + h + i + j$ ;

(疑似命令)

```
$s0 = g;  
$s1 = h;  
$s0 = $s0 + $s1  
$s1 = i  
$s0 = $s0 + $s1  
$s1 = j  
$s0 = $s0 + $s1  
f = $s0
```

(MIPSアセンブリ)

```
lw $s0, g($0)  
lw $s1, h($0)  
add $s0, $s0, $s1  
lw $s1, i($0)  
add $s0, $s0, $s1  
lw $s1, j($0)  
add $s0, $s0, $s1  
sw $s0, f($0)
```

(1) if (a == b + 0x100) { d = a + 4;} else {d = b - 1;}  
(2) for ( i = 999; i >= 0 ; i -= 4) sum = sum+c[i];

cはint型の配列。la命令を用いて配列の先頭アドレスをレジスタに転送して、各要素にアクセスすること。

こと。

問4

(1) 下記のコードは2つの配列を処理して重要な値をレジスタの\$v0 に収める。各配列はN1, N2語からなっており、ベース・アドレスは\$a0 と\$a1 にそれぞれ収められており、N1とN2がそれぞれ\$a2 と\$a3 に収められているとする。このコードの各行にコメントを加えなさい。また、このコード全体の処理について簡単に説明せよ。特に、\$v0 には何が返されるか。

```
      sll $a2, $a2, 2  
      sll $a3, $a3, 2  
      add $v0, $zero, $zero  
      add $t0, $zero, $zero  
outer: add $t4, $a0, $t0  
      lw $t4, 0($t4)  
      add $t1, $zero, $zero  
inner: add $t3, $a1, $t1  
      lw $t3, 0($t3)  
      bne $t3, $t4, skip  
      addi $v0, $v0, 1  
skip:  addi $t1, $t1, 4  
      bne $t1, $a3, inner  
      addi $t0, $t0, 4  
      bne $t0, $a2, outer
```

(2) このコードをクロック周波数が2GHz のマシン上で実行したとする。このマシンにおける各命令に必要なサイクル数は下の表に示したとおりである。N1とN2が任意の値の場合、このコードを実行するのに必要な時間(秒数)をN1とN2の関数として表しなさい。計算過程も説明せよ。

命令	サイクル数
add, addi, sll	1
lw, bne	2

(3) このコードを手続きとして呼び出せるように変更することを考える。入力変数と返り値がどうなるかを考慮した上で、その手続きの関数プロトタイプ宣言を C 言語で答えなさい。さらに、同じ処理をするように関数本体も C 言語で記述しなさい。

#### 問5

課題 3 (配列のコピー), 5 (行列の積), 6 (サブルーチンを用いた行列の積), 7 (手続きを用いたプログラムの作成) のソースコードを提出せよ。この際、以下の通りにコメントを入れること。

・使用しているレジスタと、その用途の一覧表を作成せよ。その際、使用しているレジスタにアドレスが入っているかデータが入っているかを明確に区別せよ。

#### 問6

下記の機能を実現する関数を作成し、そのソースコードを提出しなさい。動作確認のコードを含めたコードを作成すること。必ずxspimでシミュレーションし動作を確認すること。ただし、シフト命令や乗算命令は使わずに、本演習で利用可能な命令のみを使うこと。

- (1) 32ビット整数aをnビット右シフトする関数: `int right_shift(int a, int n)`

動作確認は0xffffffffを1, 16, 28ビット右シフトした結果を確認すること。

- (2) 32ビット整数の一部分を切り出す処理について考える。以下では、2進数を0b1111 (= 0xf = 10進数で15) のように表すことにする。例えば、32ビット整数aの下位4ビットを切り出すには

`a & 0b1111`

を計算すればよい。「0b1111」を4ビットのビットマスクと呼ぶ。nビットのビットマスクは

`(1 << n) - 1`

という計算式で得ることができる。

32ビット整数aの下位nビットを切り出す関数: `int slice(int a, int n)`