4 pages **CMPT 295: Test 2A SOLUTION** **March 11, 2016** **35 marks**

Answer all questions on the test paper. Use the backs of the pages for rough work, if necessary. Be sure your name and student number are on all pages. No calculators, cell phones, or laptops are to be used.

CAUTION: In accordance with the Academic Honesty Policy (S10.01), academic dishonesty in any form will not be tolerated.

1. A programmer plans to write a complex function ,"`FN`," of four 8-byte arguments. In addition to the registers used to pass the arguments and return the result, registers **r8** through **r15** will be used. As well, the function will require temporary storage for 2 local variables.

   (a) Draw a stack frame diagram for this subprogram **(3 marks)**

   SOLUTION:

fp →	local var 2
	local var 1
	r15
	r14
	r13
	r12
	return address

   (b) Write a sequence of instructions representing the prolog code for the function. **(3 marks)**

   SOLUTION:

```
fn:  push  %r12
     push  %r13
     push  %r14
     push  %r15
     sub   $16, %rsp
```

   (c) Write a sequence of instructions representing the epilog code for the function **(3 marks)**

   SOLUTION:

```
add   $16, %rsp
pop   %r15
pop   %r14
pop   %r13
pop   %r12
ret   # optionally part of the epilog
```

   (d) If the value in **%rsp** is `0x7fffff1640` immediately prior to calling the function what will be its value following the creation of the stack frame? **(1 mark)**

   ANSWER: 0x7fffff1640 $- ((7-1) \times 8)_{16}$ =0x7fffff1640 - 30_{16} = 0x7fffff10

2. Searching a list is a frequently occurring activity. The following algorithm searches for a given char in an array of characters, counting the number of times it occurs:

   Let C be the character sought in the array of characters STRNG:

```
int freq( char C, char *STRNG) {
     count = 0;
     i = 0;
    while ( STRNG[i] != 0x00) {
          if (STRNG[i] == C) count = count + 1;
          i = i + 1;
     }
     return count;
}
```

   Translate this algorithm into x86-64 assembly language. **(10 marks)**

   SOLUTION 1: Function implementation:

```
            .text
            .globl    freq
freq:       mov       $0, %eax        # eax = count
loop:       cmpb      $0, 0(%rsi)     # check for end of string
            je        done
            cmpb      %dil, 0(%rsi)   # Compare ASCII char with a string char
            jne       skip
            add       $1, %eax
skip:       add       $1, %rsi        # Determine addr of next char in STRNG
            jmp loop
done:       ret
```

   SOLUTION 2: Main program implementation:

```
            .data
COUNT:      .long     # or .quad
C:          .byte     #ASCII character
STRNG:      .string   #"character string"

            .text
            .globl    freq
freq:       mov       $STRNG, %rsi        # rsi = address STRNG
            mov       -1(%rsi), %rdi      # rdi = ASCII search char
            mov       $0, %eax            # eax = count
loop:       cmpb      $0, 0(%rsi)         # check for end of string
            je        done
            cmpb      %dil, 0(%rsi)       # Compare ASCII char with a string char
            jne       skip
            add       $1, %rax
skip:       add       $1, %rsi            # Determine addr of next char in STRNG
            jmp loop
done:       mov       $COUNT, %rsi
            mov       %eax, 0(%rsi)       # Store eax at addr COUNT
            ret
```

3. A 3-input digital system has the following functional specification:

a	b	c	z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

(a) Obtain a functional specification for the digital system as a Booleans expression that includes only the operators AND, OR, and NOT. **(5 marks)**

ANSWER:

$$z = a \cdot b' \cdot c' + a \cdot b' \cdot c + a \cdot b \cdot c' + a \cdot b \cdot c + a' \cdot b \cdot c$$

(b) List the number of AND, OR, and NOT gates and the number of inputs on each gate that would be required to implement this Boolean expression. **(3 marks)**

ANSWER: One 5-input OR gate, five 3-input OR gates, three NOT gates.

(c) Simplify this expression algebraically, using the laws of Boolean algebra. **(5 marks)**

SOLUTiON:

$$\begin{aligned} xor(a,b,c) &= a \cdot b' \cdot c' + a \cdot b' \cdot c + a \cdot b \cdot c' + a \cdot b \cdot c + a' \cdot b \cdot c \\ &= ab'(c' + c) + abc' + abc + a'bc + abc \\ &= a \cdot b' + ab(c' + c) + a'bc + abc \\ &= a(b' + b) + (a' + a) \cdot b \cdot c \\ &= a + b \cdot c \end{aligned}$$

(d) Draw a logic diagram for the simplified Boolean expression. **(2 marks)**