R-222 Arquitectura del Computador - LCC

Examen final condición regular

Mesa flotante 1º cuatrimestre 2023

Nombre y apellido: _____

Legajo: _____

IMPORTANTE: Justificar las respuestas e indicar de manera clara y detallada los procedimientos realizados para llegar a las soluciones.

1. (10 puntos) Dado el siguiente código en Assembler x86-64:

```
movb $120, %al
addb $60, %al             # al = ...........   CF=..... OF=.....
addb $128, %al            # al = ...........   CF=..... OF=.....
imulb %al                 # ax = ...........
```

   (a) Indicar el valor del subregistro `al`, luego de ejecutarse cada instrucción `addb`. Expresar el valor en binario, en octal y en hexadecimal.

   (b) Indicar si hubo acarreo y desbordamiento luego de realizarse las operaciones indicadas.

   (c) Indicar el resultado de las sumas en decimal y analizar si dicho resultado es correcto. Considerar tanto el caso `signed` como `unsigned`.

   (d) Indicar el valor del subregistro `ax` en hexadecimal luego de ejecutarse la instrucción `imulb`.

2. (20 puntos) Escribir un código en Assembler x86-64 equivalente al siguiente código en lenguaje C:

```
#include <stdio.h>
int main()
{
    int N=7;
    while(N!=1){
        if(N%2==0){
            N=N/2;
        }
        else{
            N=3*N+1;
        }
        printf("%d ", N);
    }
    return 0;
}
```

   Este código imprime por pantalla una secuencia de números de acuerdo a la conjetura de Collatz a partir del entero `N`. La conjetura de Collatz se define de la siguiente manera:

   - Si el número es par, se divide entre 2.
   - Si el número es impar, se multiplica por 3 y se suma 1.

La conjetura dice que siempre alcanzaremos el 1 para cualquier número con el que comencemos. Por ejemplo, si comenzamos con el número `N=7` obtendremos la siguiente secuencia: `22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1`

3. (20 puntos) Dado el siguiente código en lenguaje c:

```
int suma(int a, int b, int c, int d, int e, int f, int g, int h){
    return a+b+c+d+e+f+g+h;
}

int main(){
    int a=1,b=2,c=3,d=4,e=5,f=6,g=7,h=8;
    return suma(a,b,c,d,e,f,g,h);
}
```

Completar el siguiente código equivalente en Assembler ARM:

```
suma:
    add     ..., ..., ...
    add     ..., ..., ...
    add     ..., ..., ...
    ldr     ..., ...
    add     ... ..., ...
    ldr     ..., ...
    add     ..., ..., ...
    ldr     ..., ...
    add     ..., ..., ...
    ldr     ..., ...
    add     ..., ..., ...
    bx      ...

main:
    push    {lr}
    sub     sp, sp, #20
    mov     ..., ...
    str     ..., ...
    mov     ..., ...
    str     ..., ...
    mov     ..., ...
    str     ..., ...
    mov     ..., ...
    str     ..., ...
    mov     ..., ...
    mov     ..., ...
    mov     ..., ...
    mov     ..., ...
    bl      ...
    add     sp, sp, #20
    ldr     ...
```

4. (20 puntos) Dado el siguiente código en Assembler x86-64:

```
.data
a:      .quad 0xdeadbeef, 0x1122334455667788
b:      .long 0xffffffff
c:      .word 254

.text
.global main
```

```
main:
    movq $a, %rbx
    movl $2, %ecx
    leaq 4(%rbx,%rcx,8), %rax       # rax = .................
    movb (%rax), %dl                # dl = .................
    movl 12(%rbx), %eax             # eax = .................
    .... ..(....), ......          # ax = 0xadbe
    .... ..(....), ......          # rax = 0xffffff11223344
    .... ..(....), ......          # rax = 0xffffffff11223344
    imulw c, %cx                    # cx = .................
    retq
```

(a) Realizar el diagrama de memoria, expresando los valores en hexadecimal e indicando las direcciones de memoria correspondientes. Asumir que la variable `a` está alojada en la dirección `0x404000`.

(b) Completar lo que falta en las líneas de puntos para que cada línea de código se corresponda con el valor del registro indicado en el comentario correspondiente. **IMPORTANTE:** No usar movimientos de valores inmediatos.

5. (15 puntos) Dados los siguientes números en punto flotante norma IEEE 754 simple precisión:

   - `A=0x3ff80000`
   - `B=0x3f780003`
   - `C=0x3e780007`

   (a) Realizar la suma `A+B+C`.

   (b) Convertir el resultado a doble precisión.

   (c) Indicar la instrucción para realizar dicha conversión, tanto en Assembler x86-64 como en ARM.

   (d) Analizar el error cometido por representación en el ítem (a).

6. (15 puntos) Dado un sistema que emplea un esquema de paginación con las siguientes características:

   - La memoria virtual es de 4 GB (2^{32} bytes).
   - El tamaño de página es de 8 KB (2^{13} bytes).
   - La memoria física es de 8 MB (2^{23} bytes).

   (a) ¿Cuál es el formato de las direcciones virtuales? Especificar el tamaño de los campos y su significado.

   (b) Atendiendo al contenido de la TLB y de la tabla de páginas de un proceso mostradas a continuación, indicar la dirección de memoria accedida para las siguientes direcciones virtuales:

   1. 0x00001A07
   2. 0x00007100
   3. 0x00009B06

   Tener en cuenta que las entradas de las tablas tienen 2 bits de control (los dos bits más significativos), de los cuales el más significativo es el bit de presencia y el siguiente es el bit de modificación.

TLB

2	0xA1C
5	0xC04
1	0x043
3	0xB07

Tabla de paginación

0	0xFF1
1	0x043
2	0xA1C
3	0x203
4	0x66A
5	0xC04
6	0x60F
7	0x663
8	0x011

(c) Atendiendo a las direcciones virtuales del apartado anterior, indicar si se debe producir acceso a disco en cada caso. ¿Por qué?

Ejercicio	1	2	3	4	5	6	Total
Puntos	10	20	20	20	15	15	100
Puntos obtenidos							