R-222 Arquitectura del Computador - LCC

Examen Final - 9 de febrero de 2023

Examen condición libre

Nombre y apellido: _____

Legajo: _____

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

1. (30 puntos) Dado el siguiente código en lenguaje C:

```
#include <stdio.h>
int a = 0x7fffffff, b = 1;

int foo1(){
    a=a+(b<<31);
    return a;
}

int foo2(){
    a=a+b;
    return a;
}

int main(){
    printf("0x%x 0x%x\n ", foo1(), foo2());
    return 0;
}
```

   (a) Indicar lo que se imprime en pantalla al ser ejecutado.

   (b) Indicar el valor de las banderas `CF` y `OF` luego de realizarse cada una de las sumas.

   (c) Escribir un código equivalente en Assembler x86-64.

   (d) Dibujar un esquema que muestre la evolución de la pila.

2. (30 puntos) Dados dos arreglos de enteros de dimensión conocida declarados como variables globales:

```
int list1[5]={1, 2, 3, 4, 5};
int list2[5]={6, 7, 8, 9, 10};
```

   Escribir un programa en Assembler x86-64 tal que imprima por pantalla cada una de las componentes del arreglo que resulta de concatenar ambos arreglos. Es decir, con los valores de los arreglos mostrados debería imprimir `1 2 3 4 5 6 7 8 9 10`.

3. (20 puntos) Dado el siguiente código en Lenguaje C:

```
#include <stdio.h>
struct empleado {
    char nombre[10];
```

```
    char apellido[10];
    short dia_ingreso, mes_ingreso, anio_ingreso, edad;
    float sueldo;
};

struct empleado e1 = {"Juan", "Lopez", 9, 10, 2010, 45, 65536.0};

int main()
{
    printf ("Nombre: %s\n Apellido: %s\n Sueldo: %f\n", e1.nombre, e1.apellido, e1.sueldo);
    return 0;
}
```

(a) Dibujar el esquema de memoria, indicando los valores almacenados y las direcciones de memoria. Asumir que la estructura `e1` está alojada en la dirección `0x404040`.

(b) Escribir un código equivalente en Assembler x86-64.

4. (10 puntos) Dado el siguiente programa en Lenguaje C:

```
#include <stdio.h>
int main(){
    int a = 0x7fffffff, b = 0x1;
    printf("%x\n %x\n", a+b+(b<<31), a+b);
    return 0;
}
```

(a) Indicar lo que sale impreso en pantalla al ejecutarse el programa.

(b) Indicar si los resultados de las sumas son correctos.

(c) Indicar si hubo acarreo.

(d) Indicar si hubo desbordamiento.

5. (20 puntos) Dado el siguiente código en Assembler x86-64:

```
                          .global main
                          main:
0x0000000000401102:               movq    $0x4000, %rdi
0x0000000000401109:               movq    $0x1122334455667788, %rsi
0x0000000000401113:               pushq   %rdi
0x0000000000401114:               pushq   %rsi
0x0000000000401115:               call    foo
0x000000000040111a:               popq    %rsi
0x000000000040111b:               popq    %rdi
0x000000000040111c:               retq

                          foo:
0x000000000040111d:               pushq   %rbp
0x000000000040111e:               movq    %rsp,%rbp
0x0000000000401121:               movq    0x10(%rbp), %rax    # <--- %rax=?
0x0000000000401124:               movq    0x18(%rbp), %rdi    # <--- %rdi=?
0x0000000000401127:               movq    %rbp,%rsp
0x000000000040112a:               popq    %rbp
0x000000000040112b:               ret
```

(a) Escribir el contenido de la pila luego de ejecutarse el prólogo de la función `foo`, suponiendo como estado inicial `rbp=0x7fffffffe088` y `rsp=0x7fffffffe000`.

(b) Indicar el valor de los registros `rax` y `rdi` luego de ejecutarse las líneas correspondientes a sus comentarios.

(c) Indicar el valor de retorno del programa.

(d) Supongamos que hipotéticamente no tenemos disponible la instrucción `call`. Indicar un código Assembler alternativo para poder reemplazar a dicha instrucción en el código anterior.

6. (15 puntos) Dado el siguiente código en C que convierte de hectáreas a acres:

```
#include <stdio.h>
    float hectareas_2_acres(float a){
    return a*2.471;
}

int main(){
    float a=7.0;
    printf("%f hectareas = %f acres\n", a, hectareas_2_acres(a));
    return 0;
}
```

Escribir un código equivalente en Assembler ARM para la función `hectareas_2_acres`.

7. (15 puntos) Dados los siguientes números en norma IEEE 754 simple precisión:

   - `A=0x00780000`
   - `B=0x00f80000`

   (a) Realizar el procedimiento para realizar la suma `S = A + B` en punto flotante simple precisión IEEE 754, mostrando el resultado en formato hexadecimal.

   (b) Indicar si el resultado es un número normalizado.

   (c) Indicar el menor número normalizado representable para esta forma de representación.

   (d) Convertir el resultado a doble precisión.

8. (30 puntos) Dado el siguiente código en C que realiza la suma elemento a elemento de dos arreglos de enteros:

```
#include <stdio.h>

int a[]={1, 2, 3, 4};
int b[]={5, 6, 7, 8};
int c[]={0, 0, 0, 0};

void suma(int a[], int b[], int L){
    int i;
    for(i=0; i<L; i++){
        c[i] = a[i] + b[i];
    }
}

int main(){
    int i, L=4;
    suma(a, b, L);
    return 0;
}
```

Escribir un código equivalente en Assembler ARM.

9. (15 puntos) Dadas las siguientes instrucciones:

   A. `mov r0, #0x1fe00`

   B. `mov r0, #0x1fc00`

   C. `mov r0, #0x102`

   D. `mov r0, #0x104`

E. `mov r0, #0xffffff00`

F. `mov r0, #0xffff`

(a) Indicar cuáles de las mismas son válidas dentro de la arquitectura ARM.

(b) En los casos positivos, indicar cómo se realiza la codificación del valor inmediato dentro de la instrucción.

(c) En los casos negativos, indicar si se puede implementar utilizando la instrucción `mvn` y cómo.

(d) Si no es posible utilizando la instrucción `mvn`, indicar otra opción para poder cargar el valor inmediato en el registro indicado.

10. (15 puntos) Un computador posee un sistema de memoria virtual implementada mediante paginación que utiliza páginas de 8 KB. El computador proporciona un espacio de memoria virtual de 2^{32} bytes y tiene 2^{23} bytes de memoria física. Si la tabla de páginas correspondiente a un proceso en ejecución es la siguiente:

Bit de presencia	Bit de modificado	Número de página virtual	Marco de página/ bloque de swap
1	0	0	1
0	0	1	7
1	1	2	9
1	0	3	14
1	0	4	8
1	1	5	3
0	0	6	25
0	1	7	16
0	0	8	23
1	0	9	78

(a) Indique el formato de la dirección virtual, indicando los campos y el número de bits de los mismos.

(b) ¿Cuál es el tamaño que ocupa el espacio de direcciones virtual de este proceso?

(c) Exprese en MB el tamaño de la memoria principal.

(d) Indique la dirección física correspondiente a la dirección virtual `0x0000608A`.

(e) Indique la dirección física correspondiente a la dirección virtual `0x0000208A`.

(f) Indicar en cuál de las dos traducciones anteriores se ha tenido que hacer un "swap".

(g) ¿Cuántos MB se transfieren entre disco y memoria principal cuando ocurre un fallo de página?

Ejercicio	1	2	3	4	5	6	7	8	9	10	Total
Puntos	30	30	20	10	20	15	15	30	15	15	200
Puntos obtenidos											