Examen de Arquitectura de Computadoras

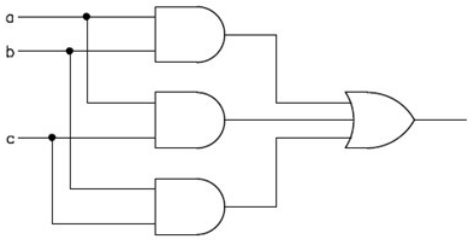
21 de julio de 2017

Instrucciones:

- Indique su nombre, apellido y número de cédula en todas las hojas que entregue.
- Escriba las hojas de un solo lado.
- Las hojas deben estar numeradas y en la primer hoja debe escribirse el total.
- Empiece cada ejercicio en una hoja nueva.
- No puede utilizar material ni calculadora.
- **Apague su celular.**
- La duración del examen es de tres horas. En dicho tiempo debe también completar sus datos.
  Solo se contestarán dudas de letra. No se aceptan preguntas en los últimos 30 minutos del examen.
- Para aprobar debe contestar correctamente un ejercicio entero y dos preguntas.

Pregunta 1

Demuestre que la expresión f = bc + ab + a$\bar{b}$c + $\bar{a}$bc y el siguiente circuito representan la misma función booleana.



Pregunta 2

Sume estos dos números en punto flotante media precisión: 1001100000111001 y 0010100101101111.

Pregunta 3

¿Qué es un banco de registros? ¿Es un elemento esencial en la propuesta de Von Neuman? Justifique.

Pregunta 4

En una CPU de 16 bits con 64 Kbytes de memoria se coloca una cache de 128 Bytes, tamaño de bloque 8 Bytes y 16 lineas; con correspondencia totalmente asociativa.
a) Indique como se interpreta una dirección de memoria en relación con el sistema de cache.
b) Calcular, en cada caso, el hit rate de los accesos a memoria al ejecutar los siguientes códigos, si la matriz B se almacena por filas y la cache está inicialmente vacía:

I)

```
short B[8][8];
short suma = 0;

for ( int  i=0; i<8; i++){
   for ( int j=0; j<8; j++){
       suma += B[j][i];
   }
}
```

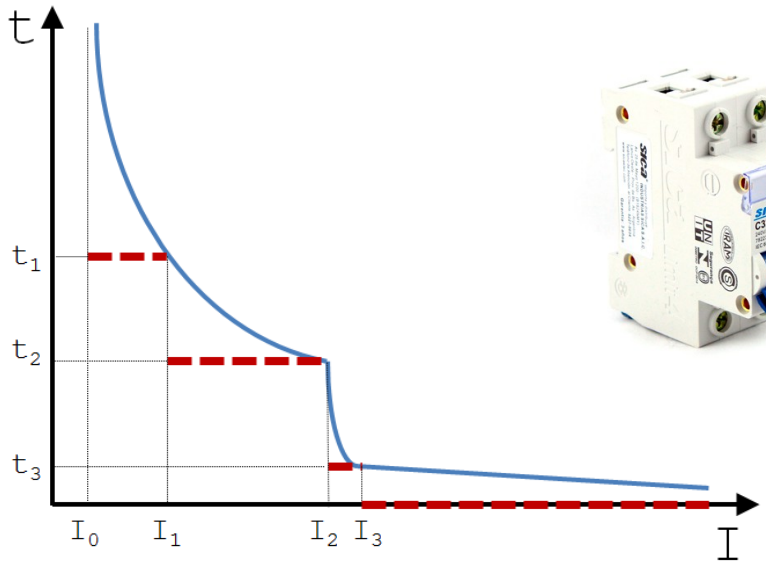
II)

```
short B[8][8];
short suma = 0;

for ( int  i=0; i<8; i++){
   for ( int j=0; j<8; j++){
       suma += B[i][j];
   }
}
```

Problema 1

El gráfico muestra, en trazo continuo, la denominada "curva" de una llave termo-magnética. Esta curva especifica el tiempo **"t"** en el cual la llave "salta" en función de la corriente **"I"** que circula por la llave. Cuando la llave "salta", abre el circuito, interrumpiendo el paso de la corriente.



Rango I	Tiempo que demora en "saltar"
menor que I0	infinito
entre I0 e I1	t1
entre I1 e I2	t2
entre I2 e I3	t3
mayor que I3	0

Se pretende emular el comportamiento de una llave termo-magnética en un sistema con un microprocesador dedicado, aproximando la curva por los segmentos de recta punteados (ver tabla).

Se mide la intensidad **I** de la corriente para determinar si la llave debe saltar. Sea **t** el tiempo desde que la intensidad medida es mayor a I_0. Si **t** es mayor al límite de tiempo asociado al rango de la intensidad medida, la llave salta (se abre el circuito).

El sistema contará con un botón para abrir o cerrar el circuito, equivalente al accionador manual de una llave termo-magnética. Cuando el botón es pulsado se debe abrir el circuito en caso de que esté cerrado, y se debe cerrar en caso que esté abierto.

Para ello se dispone de los siguientes elementos:

- Un contactor (llave eléctrica comandada) que abre o cierra el circuito en función del valor del bit 0 (un 1 cierra el circuito) del byte de E/S (solo escritura) en la dirección **CONTACTOR**.

- Un medidor de intensidad de corriente. Para medir es necesario iniciar el proceso de lectura escribiendo en el byte de E/S en la dirección **MEDIDA**. Pasado 1 ms desde dicha escritura, el valor (en miliAmperes - mA) de la intensidad **I** de la corriente estará disponible en la palabra (16 bits) de E/S en la dirección **CORRIENTE**. No se puede solicitar una nueva lectura hasta que haya terminado la anterior.

- Un timer que interrumpe con una frecuencia de 100 Khz invocando a la rutina **timer()**.

- Un botón que cada vez que es pulsado genera una interrupción invocando a la rutina **boton()**.

Se pide: Implementar todas las rutinas necesarias para emular la llave termo-magnética, sabiendo que t_1 = 10 minutos, t_2 = 1 minuto, t_3= 1 segundo, I_0= 10A, I_1= 20A, I_2 = 30A e I_3 = 35A.

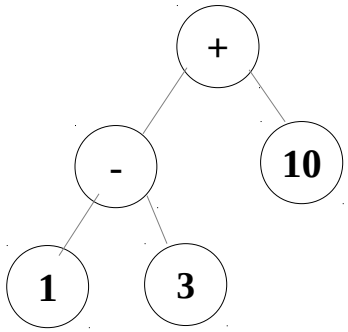
Problema 2

Se almacena una expresión que involucra sumas y restas de enteros como un árbol binario. Los nodos hoja contienen los valores de los operandos y los nodos que no son hoja contienen la operación codificada.

A continuación se muestra un ejemplo de un árbol binario que almacena una expresión y se presenta el código en alto nivel de una función que dado un árbol de este tipo devuelve el resultado de evaluar la expresión almacenada en él.

```
typedef struct{
    short val;
    nodo* izq, der;
} nodo
```

```
short evaluar(*nodo){
  if (nodo → izq != NULL){
    short res_a = evaluar(nodo → izq);
    short res_b = evaluar(nodo → der);
    if (nodo → val == 0){
      return res_a + res_b
    }else{
      return res_a - res_b
    }
  }else{
    return nodo → val;
  }
}
```



Árbol de ejemplo. Codifica la expresión ((1-3)+10) y por lo tanto el resultado de evaluarlo es: 8

Se pide:

A) Compilar la función en 8086. Los punteros son desplazamientos en el segmento ES. Los parámetros se pasan y devuelven por stack, no es necesario preservar los registros.

B) Dado un árbol de N nodos, calcular cuál es el consumo del stack en el peor caso.

Solución Pregunta 1

El circuito en el diagrama representa la función g = ab + bc + ac.
Realizando las tablas de verdad de ambas funciones vemos que son idénticas.

La tabla de verdad de g es la siguiente:

a	b	c	g = ab+bc+ac
0	0	0	00+00+00 = 0
0	0	1	00+01+01 = 0
0	1	0	01+10+00 = 0
0	1	1	01+11+01 = 1
1	0	0	10+00+10 = 0
1	0	1	10+01+11 = 1
1	1	0	11+10+10 = 1
1	1	1	11+11+11 = 1

La tabla de verdad de f es la siguiente:

a	b	c	f = bc + ab + a$\bar{b}$c + $\bar{a}$bc
0	0	0	00+00+010+010+100 = 0
0	0	1	01+00+011+011+101 = 0
0	1	0	10+01+010+010+110 = 0
0	1	1	11+01+011+001+111 = 1
1	0	0	00+10+100+110+000 = 0
1	0	1	01+10+101+111+001 = 1
1	1	0	10+11+110+100+010 = 1
1	1	1	11+11+111+101+011 = 1

Solución Pregunta 2

x=1|00110|0000111001 , y=0|01010|0101101111
Se alinean las mantisas llevando x hacia y. F_x = 0,00010000111001, F_y = 1,0101101111.
Resto el más chico al más grande:

```
  1,01011011110000
- 0,00010000111001
--------------------------
  1,01001010110111
```

El exponente queda igual y el signo es positivo por lo tanto el resultado final es: 0|01010|0100101011

Solución Pregunta 3

Un registro es una unidad de memoria formada por flip-flops y el banco de registros es un conjunto de registros que se encuentran dentro de la CPU. No es un elemento escencial de la arquitectura, es un artefacto tecnológico que incrementa la performance acercando memoria rápida a la Unidad de Control. A los efectos de las operaciones a realizar son posiciones de memoria que almacenan datos igual que la memoria “común”, que se distinguen por su manera particular de ser direccionadas.

Solución Pregunta 4

a) 15 – TAG – 3 | 2 – BYTE – 0

b) Un short ocupa 2 bytes por lo que cada línea de la cache contiene 4 entradas de la matriz.

Programa I: El acceso es por columnas y toda la primer columa resulta en un miss, pero dado que toda la matriz entra en la cache, las siguientes 3 columnas resultan en hit. Lo mismo pasa con las columnas 5 – 6,7,8. Por lo tanto el HIT RATE es 75%.

Programa II: El acceso es por filas. El primer y el quinto elemento de cada fila son misses pero el resto son hit, por lo que nuevamente el HIT RATE es 75%

Solución Problema 1

```
#define t1              10 * 60 * 100000        // El timer es de 100 Khz
#define t2              1 * 60 * 100000
#define t3              1 * 100000
#define I0              10 * 1000               // El medidor de corriente da el valor en mA
#define I1              20 * 1000
#define I2              30 * 1000
#define I3              35 * 1000
#define DEMORAMEDIDA    1 * 100                 // 1 ms = 100 * 10 us
#define ABIERTA         0
#define CERRADA         1

int ticsI; ticsMedida, corrienteI;
boolean estadoLlave;

void interrupt timer() {
      ticsI++;
      if (corrienteI <= I0) ticsI = 0;
      else if (corrienteI <= I1) {
            if (ticsI >= t1) {
                  estadoLlave = ABIERTA;
                  out(CONTACTOR, 0);
            }
      }
      else if (corrienteI <= I2)
            if (ticsI >= t2) {
                  estadoLlave = ABIERTA;
                  out(CONTACTOR, 0);
            }
      }
      else if (corrienteI <= I3)
            if (ticsI >= t3) {
                  estadoLlave = ABIERTA;
                  out(CONTACTOR, 0);
            }
      }
      else {
            estadoLlave = ABIERTA;
            out(CONTACTOR, 0);
      }
      ticsMedida++;
      if (ticsMedida >= DEMORAMEDIDA) {
            corrienteI = in(CORRIENTE);         // Leo valor de I
            ticsMEDIDA = 0;
            out(MEDIDA, 0);                     // Lanzo una nueva medida
      }
}
```

```
void interrupt boton() {
      if (estadoLlave == ABIERTA) {
            estadoLlave = CERRADA;
            out(CONTACTOR, 1);
            corrienteI = 0;                    // para que no salte sin medir nuevamente
            ticsMEDIDA = 0;
            out(MEDIDA, 0);                    // Lanzo una nueva medida
      }
      else {
            estadoLlave = ABIERTA;
            out(CONTACTOR, 0);
      }
}

void main(){
      // instalo interrupciones
      ticsI = 0;
      estadoLlave = ABIERTA;
      out(CONTACTOR, 0);                       // llave abierta = no circula corriente
      enable();
      while (TRUE);
}
```

Solución Problema 2

A)

```
evaluar proc near

     push bp
     mov bp, sp
     push ax,
     push bx,
     push cx,

     mov bx, [bp + 4] // *nodo
     cmp word ptr es:[bx + 2], 0 // if (nodo → izq != NULL)
     je ELSE
     push es:[bx + 2]
     call evaluar
     pop ax // short res_a = evaluar(nodo → izq);
     push es:[bx + 4]
     call evaluar
     pop cx // short res_b = evaluar(nodo → der);
     cmp word ptr es:[bx], 0
     jne ELSE2
     add ax, cx
     jmp FIN
ELSE2:
     sub ax, cx
     jmp FIN

ELSE:
     mov ax, es:[bx]

FIN:
     mov [bp+4], ax
     pop cx
     pop bx
     pop ax
     pop bp
     ret

evaluar endp
```

B) Llamada = 2 bytes parámetro + 2 bytes IP
   Preservo -> 8 bytes * llamada
   => consumo stack = 12 bytes * llamadas

El peor caso se da cuando ningún hijo derecho tiene hijos. Puede pensarse como dos listas paralelas una de largo L y otra de largo L-1.La cantidad de nodos es N = L + L – 1. L coincide con la cantidad total de llamadas anidadas (incluyendo la inicial), por lo que se tiene que el consumo del stack es: 12 * ((N + 1) / 2) bytes