# Examen de Arquitectura de Computadoras

## 23 de diciembre de 2014

**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*

1. Indique qué es y cuál es la función del chip select (CS) y el output enable (OE).
2. Construya una ROM de 32x8 en base a dos ROMs de 16x8.

## *Pregunta 2*

1. Indique para cada uno de los siguientes sistemas si son capaces de corregir errores:
   - Código de Hamming
   - Paridad
   - Entrelazado 2 de 5
   - Paridad horizontal+vertical
2. Codifique en el sistema de Hamming la siguiente tira : 1001

## *Pregunta 3*

1. Enumere y describa los componentes de la arquitectura de Von Neumann.
2. Enumere y describa las etapas del ciclo de instrucción.

## *Pregunta 4*

En la dirección de memoria ES:[DI] está guardado el entero -15.

- Sabiendo que el valor de ES es 0x1030 y el de DI 0x22AA, ¿en qué posición de memoria física está guardado el entero?
- Especifique el valor de DS para que el acceso a DS:[0x1A] coincida con el entero guardado.
- Explique por qué no es posible especificar un valor para DS de forma tal que DS:[0x5] coincida con la dirección física del dato guardado.

## *Problema 1*

Se desea construir un cuadricóptero, el *SantaCóptero,* con su mecanismo de nivelación basado en un sistema **no dedicado**.
El *SantaCóptero* posee cuatro motores que se encuentran en los extremos de sus brazos. Cada brazo posee un sensor de altura relativa, que permite indicar si está muy bajo o muy alto respecto al centro del *SantaCóptero*. Si está muy bajo, su motor deberá aumentar su velocidad y viceversa. Además, el *SantaCóptero* cuenta con un sensor de golpes que permite reaccionar rápido frente a cambios imprevistos.

Los motores se controlan simultáneamente escribiendo en el puerto de E/S, de solo escritura de 16 bits, en la dirección **MOTORES**. Cada motor se controla con 4 bits, el motor 0 ocupa los bits menos significativos, el motor 1 los siguientes 4 bits, etc.
Los sensores de altura relativa ocupan 2 bits cada uno y están accesibles en el puerto de solo lectura de 8 bits en la dirección **ESTADO**. El sensor 0 ocupa los dos bits menos significativos, el sensor 1 los siguientes dos bits, etc. En cada sensor el bit más significativo indica si está fuera de posición y el menos indica si está más alto (1) o más bajo (0) de lo que debería.
Si está más alto se debe disminuir en uno la velocidad actual de ese motor, si está más bajo se debe aumentar en uno y si está en posición se debe dejar su velocidad actual. Debe tener en cuenta que si un motor ya está en su velocidad máxima, no debe incrementar su velocidad. Análogamente para la velocidad mínima.
En caso de dispararse el sensor de golpes, se ejecuta la rutina de interrupción **golpe()**. En este caso se deben poner al máximo o al mínimo por 500ms los motores cuyos sensores de altura relativa estén fuera de posición. Luego deben volver a su valor anterior. Si durante estos 500ms vuelve a dispararse la interrupción golpe(), ésta se debe ignorar.

El sistema cuenta con un reloj de 20Hz que interrumpe ejecutando la rutina **tiempo()**.

**Se pide:** Implementar todas las rutinas del sistema necesarias para la **máquina no dedicada**.

Nota: la velocidad inicial para los motores es de 8.

## *Problema 2*

Se considera la siguiente estructura de árbol binario:

```
struct
{
     unsigned char hijoIzq;
     unsigned char hijoDer;
     unsigned short dato;
} nodo;
nodo arbol[256]
```

En donde hijoIzq e hijoDer son los índices a los subárboles izquierdo y derecho respectivamente para un nodo dado.
El valor **0** para un índice indica que no existe nodo asociado. El árbol tiene por lo menos un elemento y no hay nodos con **dato** repetido en el mismo.

**Se pide:**

**a)** Escribir en alto nivel un procedimiento recursivo que retorna el dato mayor que se encuentra en el árbol y su índice asociado en el arreglo.
El procedimiento tiene la siguiente firma:

```
void buscoMayor(unsigned char indice, out short mayor, out char indiceMayor)
```

Compilar la rutina en assembler 8086 sabiendo que la variable **arbol** se encuentra a partir de la dirección segmentada **ES:0000**. El parámetro se recibe en el stack y el resultado también se debe retornar por el stack. La rutina debe conservar todos los registros de propósito general.

**b)** Calcular el tamaño mínimo que debe tener el stack para que la función pueda ser ejecutada en todos los casos, cualquiera sea el tamaño y topología del árbol.