ET1032 Informática Industrial

10 de julio de 2014

Soluciones al examen.

Pregunta 1. (1,25 puntos) - Indicad qué es el contador de programa y cómo interviene en la ejecución de instrucciones por parte de un procesador.

El contador de programa es el registro del procesador que almacena la dirección de la siguiente instrucción que se va a ejecutar. Al principio del ciclo de ejecución de instrucciones, el contenido del contador de programa se utiliza como dirección de acceso a la memoria para leer de ella la instrucción que se va a ejecutar. La dirección contenida en el contador de programa se incrementa o modifica durante el resto de la ejecución de la instrucción para apuntar a la siguiente instrucción a ejecutar según el flujo del programa.

Pregunta 2. (1,25 puntos) – Justificad por qué es necesario que un procesador disponga de dos modos de funcionamiento con privilegios distintos, modo supervisor y modo usuario, para poder implementar sobre él un sistema operativo multiusuario con protección efectiva entre los distintos usuarios.

Un sistema operativo multiusuario permite la ejecución simultánea de procesos de distintos usuarios. Para que la ejecución incorrecta o malintencionada por parte de un usuario no afecte a los demás o al sistema en sí, es necesario que exista un mecanismo adecuado de protección. La forma más común y eficaz de poder implementar este mecanismo es que el procesador disponga de dos modos distintos de ejecución. Uno de ellos, menos privilegiado, es el modo usuario y tiene limitaciones en cuanto al acceso a los recursos del sistema –memoria, entrada/salida, recursos e instrucciones del procesador. El otro, llamado supervisor, no tiene estas restricciones. En caso de cualquier error, incluyendo entre ellos intentos de violación de estos privilegios, el procesador genera una excepción y pasa a ejecutar código del sistema operativo, en modo supervisor. De esta manera, el sistema operativo puede implementar las medidas adecuadas para proteger a los demás usuarios y a sí mismo frente a estos errores de usuario.

Pregunta 3. (1,25 puntos) – Describid brevemente qué se entiende por dispositivo electrónico optoacoplado. Indicad en qué casos es necesario su uso.

Un dispositivo optoacoplado dispone, en un mismo encapsulado, dos circuitos electrónicos independientes que interactúan de manera óptica, normalmente mediante la emisión de fotones por un LED que actúan sobre la base de un transistor generando el mismo efecto que la corriente de polarización de base. De esta manera se permite que un circuito interactúe con el otro sin conexión eléctrica entre ellos. Por esta razón, estos dispositivos se utilizan para interconectar funcionalmente sistemas electrónicos manteniéndolos totalmente aislados de forma eléctrica, evitando casi totalmente la inducción de ruido eléctrico entre ellos.

Pregunta 4. **(1,25 puntos)** – Cuál será el valor, en hexadecimal, del entero **`dato`** después de ejecutar las instrucciones que aparecen a continuación. Indicad justificadamente todas las soluciones posibles. Suponed enteros de 32 bits.

```
void main()
{
   int dato, *puntero;
   char *frase = "Hola, mundo";

   puntero = (int *)frase;
   dato = *puntero;
   // Seguiría el programa …
}
```

Nota: los caracteres de frase se almacenan como los bytes **0x48, 0x6F, 0x6C, 0x61, 0x2C, 0x20, 0x6D, 0x75, 0x6E, 0x64, 0x6F, 0x00**

En la primera línea de código se realiza una conversión forzada de tipos y se copia el valor del puntero a carácter **`frase`** en el puntero a entero **`puntero`**. Cuando accedemos al contenido de **`puntero`** para copiarlo en **`dato`**, y dado que los enteros son de 32 bits, lo que hacemos es formar un entero con los 4 primeros bytes de **`frase`**.

Ahora bien, dado que la memoria siempre se direcciona por bytes, cuando hemos de formar este valor de 4 bytes podemos hacerlo de dos formas: considerando que el byte de mayor peso es el que ocupa la dirección más baja de la memoria –caso que se denomina *big endian*- o considerando, al contrario, que el que ocupa la dirección más baja de memoria es el de menor peso –*little endian*. Según esto, el valor de **`dato`** será **`0x486F6C61`** en el caso *big endian* y **`0x616C6F48`** en el *little endian*.

Pregunta 5. **(2,5 puntos)** – Suponed que se dispone de cierta biblioteca que incluye la definición de tipo **`complejo`** y las operaciones básicas entre números complejos que aparecen a continuación.

```
typedef struct {
   double r, i;
} complejo;

complejo suma(complejo a, complejo b);
complejo resta(complejo minuendo, complejo sustraendo);
complejo multiplica(complejo a, complejo b);
complejo divide(complejo dividendo, complejo divisor);
```

Se requiere, utilizando esta biblioteca, definir un tipo de datos –una estructura- que represente ecuaciones de números complejos con dos incógnitas, y una función que resuelva sistemas de dos ecuaciones. Se pide:

- Definir un tipo de datos que permita representar una ecuación del tipo $aX + bY + c = 0$, donde todos los coeficientes son números complejos. (**0,5 puntos**)
- Implementar una función **`complejo *resuelve(int *sol, ecuacion eq1, ecuacion eq2)`** que devuelva un vector de 2 complejos con la solución del sistema formado por **`eq1`** y **`eq2`** o dos complejos 0 –parte real e imaginaria igual a 0- si el sistema no tiene solución. Además en el entero **`sol`** se devolverá 0 si el sistema tiene solución, -1 si el sistema no tiene solución, 1 si el sistema tiene infinitas soluciones y -2 si se da otro tipo de error. (**2 puntos**)

Vamos a estudiar en primer lugar los aspectos matemáticos de la función que se pide. Si se tiene un sistema con dos ecuaciones como el siguiente,

$$a_1x + b_1y + c_1 = 0$$
$$a_2x + b_2y + c_2 = 0$$

Se sabe que, de tener soluciones, estas son

$$x = \frac{b_1c_2 - b_2c_1}{a_1b_2 - a_2b_1}, \qquad y = \frac{a_2c_1 - a_1c_2}{a_1b_2 - a_2b_1}$$

Si no se puede calcular la solución, por no existir o por no ser única, la expresión del denominador de ambas soluciones es 0. En este caso, si ambos numeradores también son cero las soluciones son infinitas –sistema compatible indeterminado- y en otro caso no hay solución –sistema incompatible. Una vez hechas estas consideraciones, es sencillo hacer un tipo de datos para las ecuaciones y una función que resuelva el sistema, como aparece a continuación.

```
typedef struct {
   complejo a, b, c;
} ecuacion;


// Constantes para el valor de sol
#define SOL_OK 0
#define SOL_INF       1
#define SOL_NO -1
#define ERROR  -2

complejo *resuelve(int *sol, ecuacion eq1, ecuacion eq2)
{
   // Se podrían usar menos variables, pero así es más claro
   complejo x, y, den, aux1, aux2, numx, numy, *res;
   // Reservamos memoria para el resultado
   if ((res = malloc(2 * sizeof(complejo))) == NULL) {
        *sol = ERROR;
        return NULL;
   }
   // Calculamos el denominador común
   aux1 = multiplica(eq1.a, eq2.b);
   aux2 = multiplica(eq1.b, eq2.a);
   den = resta(aux1, aux2);
   // Calculamos el numerador para la X
   aux1 = multiplica(eq1.b, eq2.c);
   aux2 = multiplica(eq1.c, eq2.b);
   numx = resta(aux1, aux2);
   // Calculamos el numerador para la Y
   aux1 = multiplica(eq1.c, eq2.a);
   aux2 = multiplica(eq1.a, eq2.c);
   numy = resta(aux1, aux2);
   // Si el denominador es 0 es incompatible o indeterminado
   // según los numeradores. Ponemos X e Y a 0 y el valor de sol
   if ((den.i == 0.0) && (den.r == 0.0)) {
        x.i = x.r = y.i = y.r = 0.0;
        if ((numx.i == 0.0) && (numx.r == 0.0) && (numy.i == 0.0) && (numy.r == 0.0))
               *sol = SOL_INF;
        else *sol = SOL_NO;
   }
   // Si es compatible determinado, calculamos X e Y y sol vale 0
   else {
        *sol = SOL_OK;
        x = divide(numx, den);
        y = divide(numy, den);
   }
   // Ponemos los valores en el vector y volvemos
   res[0] = x;
   res[1] = y;
   return res;
}
```

Pregunta 6. **(2,5 puntos)** - Se dispone de un microcontrolador

cuyos pines de entrada/salida tienen las características eléctricas que aparecen en la tabla. El microcontrolador se alimenta a 5V y es capaz de soportar en sus entradas tensiones entre 6 y -1V sin deteriorarse.

V_{IHmin}	3,4 V	V_{OHmin}	4,5 V	I_{Imax}	6 µA
V_{ILmax}	1 V	V_{OLmax}	0,6 V	I_{Omax}	20 mA

En los circuitos que aparecen a continuación las resistencias son de ¼ W, los diodos estándar son de silicio y tienen Vd = 0,7V y los diodos led tienen Vd = 1,2V.

Considerad en primer lugar el circuito de la figura 1. Está diseñado para encender uno u otro led según el pin de salida se encuentre a nivel alto o nivel bajo. Se pide:

- Calcular el valor de las resistencias para que la intensidad mínima que circula por los leds sea de 15 mA. Indicad la intensidad máxima que circulará por cada uno de ellos. (**0,8 puntos**)
- Verificad la corrección del diseño. ¿Es cierto que solo se encenderá uno de los leds? Justificad la afirmación. ¿Qué ocurrirá si el pin se configura como entrada, en vez de como salida? (**0,4 puntos**)

El circuito de la figura 2 sirve para detectar los pasos por cero de la tensión de la red eléctrica. Se pide:

- Calcular el valor de la resistencia que conecta el pin a la fase de 220 V. Indicad las tensiones máxima y mínima que se alcanzarán en dicho pin. (**0,5 puntos**)
- Indicad el error máximo y mínimo que se tendrá a la hora de detectar el paso por 0 –es decir, la mayor tensión de la red eléctrica a partir de la cual podrá detectarse un 0 en el pin, y la mayor tensión de la red eléctrica que garantizará la lectura de un 0 lógico en el pin. (**0,8 puntos**)

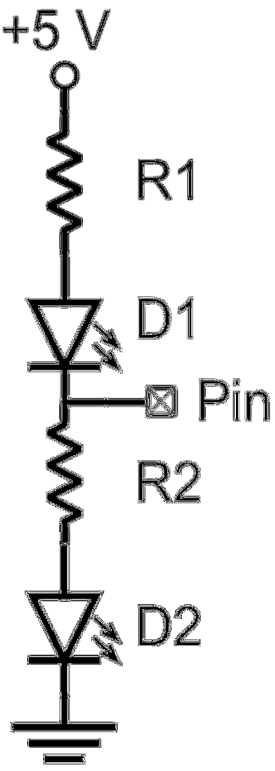

Figura 1

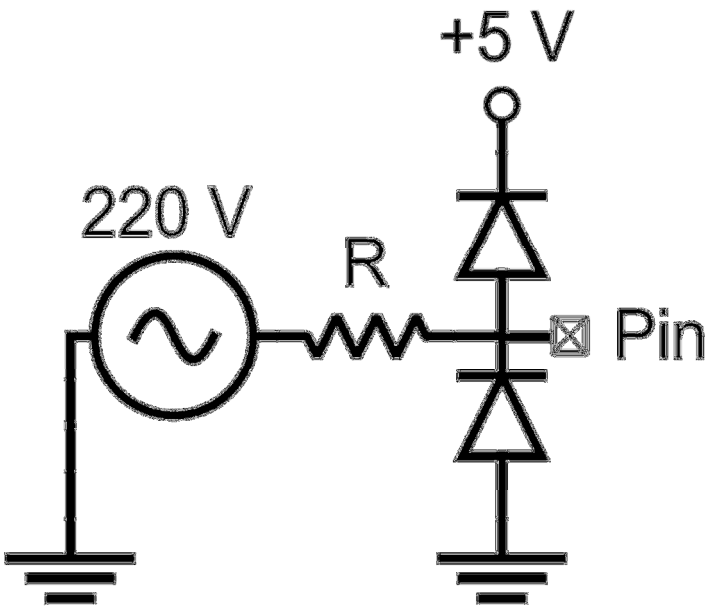

Figura 2

En primer lugar realizaremos los cálculos pedidos, suponiendo que el montaje de la figura 1 funciona correctamente –cosa que verificaremos con posterioridad. Empezaremos calculando la resistencia R2 para que la intensidad que atraviesa D2 sea como mínimo de 15 mA.

$$15mA = \frac{V_{OHmin} - 1{,}2V}{R_2} = \frac{4{,}5V - 1{,}2V}{R_2} \Rightarrow R_2 = \frac{3{,}3V}{15mA} = 220\Omega$$

Con esta resistencia, la intensidad máxima será aproximadamente

$$\frac{5V - 1{,}2V}{R_2} = \frac{3{,}8V}{220} = 17{,}3\text{mA}$$

Repitiendo los cálculos para el sistema formado por D1, que se enciende a nivel bajo, y R1 tenemos.

$$15mA = \frac{5V - V_{OLmax} - 1{,}2V}{R_1} = \frac{5V - 0{,}6V - 1{,}2V}{R_1} \Rightarrow R_1 = \frac{3{,}2V}{15mA} = 213\Omega$$

En este caso la intensidad máxima será aproximadamente

$$\frac{5V - 1{,}2V}{R_1} = \frac{3{,}8V}{213} = 17{,}8\text{mA}$$

Verifiquemos ahora que, efectivamente, solo se enciende uno de los diodos en cada momento. Consideremos que el pin está a nivel alto. Su tensión mínima será de 4,5V como se ha visto, y será suficiente para encender D2. La tensión entre los terminales de D1 será, como máximo, de 5V – 4,5V, es decir de 0,5V. Como esta tensión es inferior a Vd del LED, 1,2V, es seguro que D1 no se enciende. Del mismo modo, cuando el pin está a nivel bajo, la tensión entre los terminales de D2 es como mucho de 0,6V, de nuevo inferior a Vd por lo que el LED no se enciende.

Veamos por último qué ocurre si el pin está configurado como entrada. En este caso podemos considerarlo como una alta impedancia, que no carga el circuito al que está conectado. Entonces lo que tenemos es el conjunto D1, R1, D2, R2 en serie entre 5V y tierra. Así pues, la intensidad que circula es de

$$I = \frac{5V - 1{,}2V - 1{,}2V}{R_1 + R_2} = \frac{2{,}6V}{433} = 6mA$$

En este caso ambos LEDS se encenderán con un brillo mucho menor que en los casos anteriores.

Analicemos ahora el circuito de la figura 2. Los diodos garantizan que la tensión en el pin estará en el rango entre 5,7V y -0,7V. Si consideramos despreciable la corriente consumida por el pin, el valor de la resistencia ha de adecuarse exclusivamente para que no tenga que disipar más potencia de la que es capaz de resistir (0,25W). Los peores casos serán aquéllos en que la tensión de la red tenga su valor máximo o mínimo, que en valor absoluto y dado que la tensión eficaz es de 220V son

$$V_{max} = 220V\sqrt{2} \cong 311V$$

La mayor diferencia de tensión se dará con la tensión de red negativa y será de -311V en la entrada y -0,7 en el pin. Aplicando la fórmula de la potencia derivada de la ley de Ohm tenemos

$$P = \frac{V^2}{R} = \frac{310{,}3^2}{R} \Rightarrow R = \frac{310{,}3^2}{0{,}25} \cong 385K\Omega$$

Así pues, podemos escoger una resistencia mayor que dicho valor, por ejemplo 1M, para que la potencia disipada sea mucho menor que la potencia máxima soportada.

Veamos ahora la incertidumbre que tendremos a la hora de detectar los pasos por cero. Como se ha dicho, los diodos hacen que la tensión en el pin esté limitada entre los valores 5,7V y -0,7V. Cuando la tensión de entrada se encuentra entre estos, y dado que el pin no consume apenas corriente, la tensión en él será igual a la tensión de red. Sabemos por las características del microcontrolador que cualquier valor de entrada superior a V_{IHmin}, 3,4V, es reconocido como un 1 lógico. Por ello, hemos de suponer que cualquier valor inferior a esta tensión puede ser reconocida como un 0 lógico y ser detectada como un paso por 0 de la tensión de red. Por otra parte, también sabemos que cualquier tensión inferior a V_{ILmax}, 1V, es reconocida siempre como un 0 lógico, así que siempre tendremos como mínimo un error de esta magnitud. Resumiendo, el máximo error que podemos cometer es de 3,4V y el menor es de 1V. Estos errores suponen un adelanto en el reconocimiento de pasos por cero desde tensiones positivas y un retraso en los pasos por cero desde tensiones negativas. Ambos errores se pueden compensar por software mediante un uso adecuado de los dispositivos de medida de tiempo del microcontrolador.