Boletín de ejercicios 4 de Sistemas Digitales (a realizar entre el 11 y el 18 de octubre, 23/24)

i) La primera tarea a realizar durante las horas destinadas a actividades no presenciales de SD consiste en repasar/estudiar la materia vista en las clases de teoría. ¡Si tienes dudas vete a tutorías!

ii) La segunda tarea a realizar consiste en resolver las siguientes cuestiones y problemas:

1) Indica cuales de las siguientes afirmaciones son ciertas:

a) Se dice que dos grupos de bits son adyacentes si la distancia entre ellos es igual a 1

b) Un código puede ser continuo sin que la primera y la última de sus palabras código sean adyacentes

c) Un código puede ser cíclico pero no ser continuo.

d) Un código puede ser continuo pero no ser cíclico.

e) Los códigos de *Gray* son ponderados.

f) Una de las principales aplicaciones de los códigos de Gray es codificar la posición de objetos móviles.

g) Los códigos de paridad son útiles para detectar si una palabra presenta errores en varios de sus bits.

2) Los siguientes valores de 8 bits se grabarán en una cinta magnética de 9 pistas, una vez que hayan sido codificados en un código de paridad *par*. Indica los valores a grabar en dicha cinta.

a) 10100100 *b*) 01001010 *c*) 10101011 *d*) 01010110

3) Determina cuales de los siguientes códigos con paridad *par* son erróneos.

a) 100110111 *b*) 011001000 *c*) 101111110101101

4) Determina cuales de los siguientes códigos con paridad *impar* son correctos:

a) 1101011 *b*) 101101 *c*) 01001011 *d*) 0110100

5) ***a***) Cierto alumno ha escrito las instrucciones en ***C*** que se indican a continuación (en la parte derecha puedes ver los tipos de variables que tiene definido el compilador que utiliza):

signed short int $x = 125, y = 5, z = 0;$

$z = x + y;$

Indica el valor en binario guardado en la posición de memoria asignada a la variable ***z*** justo después de que el procesador ejecute la instrucción $z = x + y;$

Tipo de variable	Tamaño	Rango de valores
(unsigned) ***char***	8 bits	[0, 255]
signed ***char***	8 bits	[–128, +127]
unsigned short (***int***)	8 bits	[0, 255]
(signed) short (***int***)	8 bits	[–128, +127]
(signed) ***int***	16 bits	[–32768, +32767]
unsigned (***int***)	16 bits	[0, 65535]

b) El alumno no quedó satisfecho con el resultado obtenido al ejecutar el código anterior, por lo que decidió probar con las siguientes instrucciones, para ver si la ALU (*Arithmetic Logic Unit*) de su procesador estaba averiada o no:

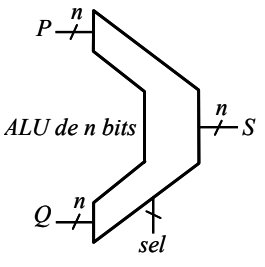
unsigned short int $x = 250,\ y = -8,\ z = 0;$

$z = x - y;$

Indica el valor en binario y en base 10 guardado en la posición de memoria asignada a la variable ***z*** justo después de que el procesador ejecute la instrucción: $z = x - y;$

Nota: *unsigned* ≡ sin signo, *signed* ≡ con signo.

Dato 1: la *unidad aritmético-lógica* (ALU: *Arithmetic Logic Unit*) de un procesador de *n* bits sólo admite datos (operandos) de *n* bits y los resultados que genera los representa utilizando *n* bits.



Dato 2: las ALUs realizan sumas con números binarios (cantidades representadas en el sistema de numeración de base $b = 2$). En el dibujo de la derecha, P y Q son las entradas de los operandos y S es la salida en la que se muestra el resultado. Las entradas *sel* sirven para indicarle a la ALU el tipo de operación (*aritmética* o *lógica*) que debe realizar con los datos presentes en las entradas.

Pista: mira la página 61 de las diapositivas del tema 1

c) El alumno (un pesado… si, lo sé) tampoco quedó satisfecho con el resultado obtenido en el apartado anterior, por lo que decidió hacer una última prueba con las siguientes instrucciones:

unsigned short int $x = 9,\ y = 3,\ z = 0,\ t = 7,\ u = 0;$

$z = x - y;$

$u = y - t;$

Indica el valor en binario y en base 10 guardado en las posiciones de memoria asignadas a las variables ***z*** y ***u*** justo después de que el procesador ejecute las instrucciones: $z = x - y;\ u = y - t;$

6) Demuestra la *veracidad/falsedad* de las siguientes igualdades, teniendo en cuenta que ***x*** e ***y*** representan elementos de un conjunto B sobre el que se ha definido un *Álgebra de Boole*. Sólo puedes utilizar axiomas y teoremas del *Álgebra de Boole*

i) $\overline{x + y \cdot x} = \overline{x}$ *ii*) $x(x + y + \overline{x}) = 1$

7) Indica cuál o cuales de las siguientes afirmaciones son ciertas y por qué:

i) $\overline{x+y\cdot z} = \overline{a+b}$ / $a = x$, $b = y\cdot z$ *ii*) $\overline{x+y\cdot z} = \overline{a\cdot b}$ / $a = x+y$, $b = z$ *iii*) $1+2\cdot 3 = 9$

8) Obtén las identidades duales de las siguientes expresiones:

a) $x + 1 = 1$ *b*) $x\cdot x = x$ *c*) $x + 0 = x$ *d*) $x + \overline{x} = 1$ *e*) $\overline{a+b} = \overline{c}\cdot\overline{d}$

f) $n(x + y) = n\cdot x + n\cdot y$ *g*) $j(z, y, x) = \overline{x\cdot y + z}$ *h*) $m(z, y, x) = \overline{x\cdot\overline{y + z}}$ *i*) $x + y\cdot z = (x + y)\cdot(x + z)$

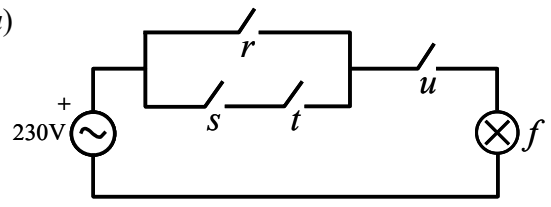
9) Representa la *tabla de verdad* de las siguientes funciones lógicas:

i) $f(c,b,a) = c\,\overline{b}\,a + \overline{c}\,b$ *ii*) $g(c,b) = (b + \overline{c})(c + \overline{b})$ *iii*) $h(c,b,a) = \overline{a}\cdot\overline{c + \overline{b}}$

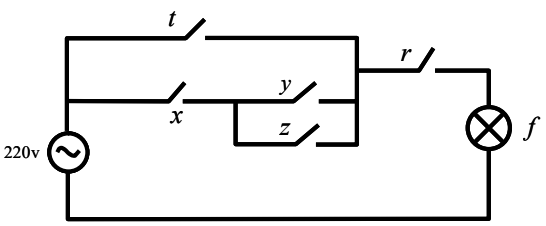
10) Representa la tabla de verdad de la función: $\beta(b,a) = \overline{b}\ a + b\ a + b\ \overline{a} + \overline{b}\ \overline{a}$. Hay quien opina que β no es una función. ¿Tu qué opinas?. ¿Por qué?. ¿En tu opinión $\alpha(b,a) = (\overline{b} + a)(b + a)(b + \overline{a})(\overline{b} + \overline{a})$ es una función?

11) Determina una expresión lógica que modele el estado de las bombillas en los siguientes circuitos. Indica cómo defines las funciones y las variables lógicas que utilices. Nota: supón que las bombillas no están fundidas y que la red eléctrica suministra energía (si no entiendes los circuitos vete a tutorías).

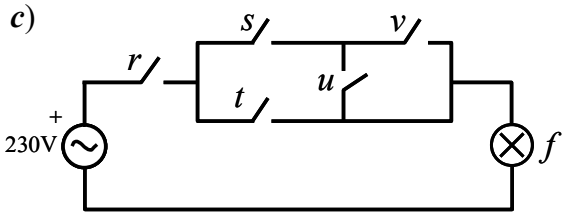
a)



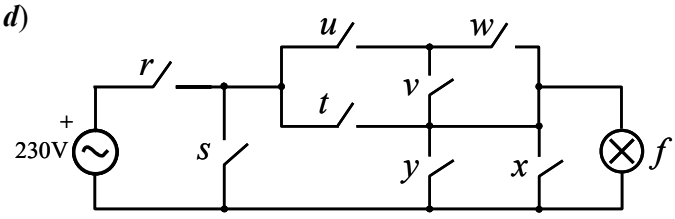
b)



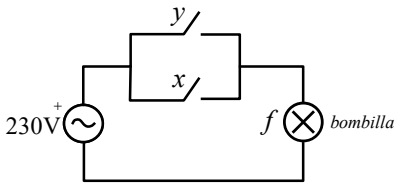
c)



d)



12) Determina una expresión lógica de la función *f* que modela el estado de la bombilla en el circuito indicado en la parte derecha, teniendo en cuenta las definiciones de dicha función y de las variables asociadas al estado de los interruptores que se indican a continuación:



i)

$$y,x=\begin{cases}0 \ \textit{si el interruptor está cerrado}\\ 1 \ \textit{si está abierto}\end{cases} \qquad f(y,x)=\begin{cases}1 \ \textit{si la bombilla está encendida}\\ 0 \ \textit{si está apagada}\end{cases}$$

ii)

$$y,x=\begin{cases}1 \ \textit{si el interruptor está cerrado}\\ 0 \ \textit{si está abierto}\end{cases} \qquad f(y,x)=\begin{cases}0 \ \textit{si la bombilla está encendida}\\ 1 \ \textit{si está apagada}\end{cases}$$

iii)

$$y,x=\begin{cases}0 \ \textit{si el interruptor está cerrado}\\ 1 \ \textit{si está abierto}\end{cases} \qquad f(y,x)=\begin{cases}0 \ \textit{si la bombilla está encendida}\\ 1 \ \textit{si está apagada}\end{cases}$$

Pasatiempos (*unas preguntas para pensar un poco*):

a) Si se multiplican dos números binarios (*base* = 2) de *n* bits cada uno, ¿cuál es el valor máximo que puede tomar el resultado?

b) Determina el número mínimo de bits que se necesita para representar el resultado de una multiplicación de dos números binarios, enteros, de *n* = 8, 16, 32 y 64 bits (considera que tanto el multiplicando como el multiplicador tienen *n* = 8, 16, 32 y 64 bits y que representan cantidades en binario natural, sin signo).

Nota 1: los ingenieros que escriben código para ser ejecutado lo más rápido que sea posible en un *DSP* (*Digital Signal Processor*), en un *DSC* (*Digital Signal Controller*) o en un *Microcontrolador* de coma fija necesitan saber la respuesta a las preguntas anteriores (si quieren que el código se ejecute correctamente).

c) Un profesor de una universidad estadounidense ha publicado un documento técnico en el que analiza las características de las diferentes estructuras que se pueden utilizar a la hora de implementar (programar) un filtro digital IIR (*Infinite Impulse Response*). En dicho documento el profesor afirma lo siguiente:

"*Para determinar el valor de w*[*n*] *definido mediante la siguiente ecuación en diferencias sólo hay que realizar N multiplicaciones y N sumas*"

$$w[n]=x[n]-b_1\cdot w[n-1]-b_2\cdot w[n-2]-b_3\cdot w[n-3]-\cdots-b_N\cdot w[n-N]$$

Suponiendo que los valores de $x[n]$, b_1, b_2, b_3, . . ., b_N, $w[n-1]$, $w[n-2]$, $w[n-3]$, . . ., $w[n-N]$ son números binarios conocidos, el que haya que realizar *N* multiplicaciones para determinar el valor de $w[n]$ parece bastante lógico, pero lo de que haya que realizar *N* sumas requiere algún tipo de explicación, ¿verdad?. Explica brevemente porqué el profesor afirma que el procesador que ejecute la ecuación en diferencias anterior tiene que realizar *N* sumas y no *N* restas (no te enrolles!!!!).

Nota: puedes asumir que en el documento publicado no hay erratas y que el profesor sabe un montón sobre *sistemas digitales*… (y que sabe sumar y restar). *Comentario*: en tercero verás un poco de filtros digitales

Algo en lo que pensar: en cierta ocasión, una persona que paseaba por un bosque (una persona con un título universitario) se encontró con un anciano que estaba plantando un nogal (el anciano no tenía ningún título). La conversación que se mantuvo entre ambos fue la siguiente:

Titulado: *¿Anciano, por qué hace eso*?

Anciano: *Pues verá usted, ¡porque me encantan las nueces!*

Titulado: *Anciano, no castigue su encorvada espalda sobre ese hoyo. ¿Acaso no ve que cuando crezca el nogal, usted ya no vivirá para recoger sus frutos*?

Anciano: *Señor, si mis ancestros hubieran pensado como usted, yo nunca habría podido comer nueces.*

¿Qué ocurrirá el día que dejen de existir personas que piensen y que actúen como el anciano y que sólo haya gente con 'títulos universitarios'?... en SD, los conceptos se comprenden estudiando la teoría y resolviendo los boletines de ejercicios. Limitarse a ir a las clases de problemas a ver como el profesor de turno resuelve ejercicios sólo vale para perder el tiempo.

La información no es conocimiento, el conocimiento no es sabiduría.
Frank Zappa
Sólo hay 2 cosas infinitas, el Universo y la estupidez humana,
aunque de la primera no estoy seguro
Albert Einstein