



ARQUITECTURA DE COMPUTADORAS PARA INGENIERÍA¹

Trabajo Práctico N° 1 Técnicas Digitales

Primer Cuatrimestre de 2016

1. Demostrar formalmente la siguiente igualdad:

$$\overline{AB + BC + CA} = \overline{A} \overline{B} + \overline{B} \overline{C} + \overline{C} \overline{A}$$

2. Demostrar que el término $A\overline{C}$ resulta superfluo en la siguiente expresión lógica:

$$AB + \overline{B} \overline{C} + A\overline{C}$$

3. Expresar $F = AC + B(D + \overline{A}) + BDC$ como:

- a) Una suma expandida de productos.
- b) Un producto expandido de sumas.

En ambos casos, notar sus respectivas formas canónicas tanto en forma expandida como compacta.

4. Bosquejar un *diagrama de Veitch* que describa la función analizada en el ejercicio anterior. A su vez, utilice un *mapa de Karnaugh* para representar al complemento de la misma.
5. Empleando la *notación canónica compacta*, definir las funciones w , y , z de un circuito combinacional de cuatro entradas A , B , C y D , donde se verifica que:
 - La salida z será 1 cuando la mayoría de las entradas sean 1,
 - La salida y será 1 cuando haya menos de tres entradas que sean 1,
 - La salida w será 1 cuando dos o tres entradas sean 1.

6. Considerando la función lógica $F(A, B, C, D) = \sum(1, 3, 7, 13, 15)$, determinar tanto su *mínima suma de productos* como su *mínimo producto de sumas*, aplicando el método gráfico. ¿Cuál resulta más simple de obtener?

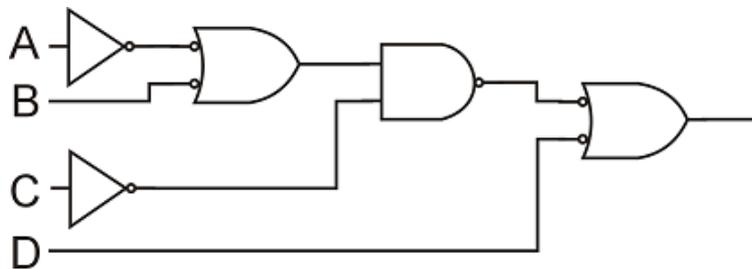
¿Las soluciones obtenidas resultan *mínimas*?

7. Sea $F(A, B, C, D) = \sum(2, 7, 8, 13)$, con $\sum_0(0, 1, 3, 5, 10, 12)$.

- a) Determinar **todos** los implicantes primos de F utilizando el **método tabular** y calcular **todas** las soluciones irredundantes *mínima suma de productos* para F aplicando el **método algebraico**.
- b) Idem anterior, pero esta vez obteniendo soluciones para el *mínimo producto de sumas*.

¹Fecha tentativa de finalización del práctico: Jueves 31 de Marzo de 2016.

8. Implementar las soluciones obtenidas en el ejercicio 6 con compuertas **NAND** e **INV** para su *mínima suma de productos*, y con compuertas **NOR** e **INV** para su *mínimo producto de sumas*. Muestre el diagrama lógico de detalle especificando claramente la polaridad de la lógica utilizada.
9. Se desea implementar un circuito con cuatro entradas A , B , C , D y cuatro líneas de control W , X , Y , y Z . El propósito de cada línea de control radica en *seleccionar* una determinada entrada. Implemente el circuito con a lo sumo 5 compuertas lógicas convencionales. ¿Existe algún inconveniente con la solución planteada? En caso de que su respuesta sea afirmativa proponga una solución alternativa.
10. Considerando el siguiente diagrama de detalle de un circuito implementado con compuertas **NAND** de dos entradas e inversores:



- a) Analizar sobre este esquema las condiciones que pueden conducir a la validación de la salida, lo cual se traducirá en la obtención de la función de salida, esto es, partiendo de la salida y retrocediendo hasta llegar a las entradas.
 OBS: En este análisis, procesar información lógica, esto es, no eléctrica. En otras palabras, analizar el circuito prescindiendo de los niveles de tensión.
- b) Si las compuertas y los inversores tienen un retardo de 3 ns (*i.e.*, $3 \times 10^{-9}\text{ s}$), ¿cuál es el retardo necesario para que la salida esté *establecida* en el peor caso, esto es, con todos los niveles de salida de las compuertas cambiando?
- c) Indicar en cada entrada y salida de compuerta cuál es la función lógica de las variables, partiendo desde las entradas del circuito, hasta llegar a su salida.
11. Considerando la función de cuatro variables $F(A, B, C, D) = \sum(1, 6, 7, 9, 10, 14), \sum_o(13, 15)$ implementada mediante un **MUX**, con las entradas B y C conectadas como líneas de dirección y las variables A y D usadas para generar las funciones residuo:
- a) Determinar todas las funciones residuo.
- b) Calcular número máximo de funciones residuo que se podrían generar con las variables A y D y obtenerlas a partir de las diversas combinaciones de *miniterms*.
 OBS: Desarrollar una tabla en dos partes: función de un lado y complemento del otro tomando los *miniterms* faltantes.
- c) Excluyendo las funciones triviales, generar las restantes funciones residuo de dos variables con a lo sumo cinco compuertas **NAND** de dos entradas y cinco **INV**.

12. Implementar las funciones requeridas para controlar un display digital de siete segmentos utilizando:
- Siete multiplexores de 8 entradas de datos.
 - Tres chips de dos multiplexores de 4 a 1 cada uno y compuertas.
 - Un decodificador BCD y compuertas **NAND**. Resolver el problema de forma directa y atacando a la función complemento. ¿Existe alguna diferencia entre ambas implementaciones? ¿Cuál recomendaría?
 - Utilizando un **PLA**, indicando el tamaño requerido.
 - Empleando una **ROM**, indicando su programación y el tamaño requerido.
13. Simular las implementaciones realizadas en los ejercicios 12b, 12c y 12e utilizando el software *LogiSim*¹.
14. Determinar el tamaño de **PLA** requerido para poder implementar:
- Cuatro multiplexores de dos entradas de datos cada uno, compartiendo líneas de selección de a dos **MUXs**.
 - Dos multiplexores de cuatro entradas y líneas de selección independientes.
 - Un *Priority Encoder* de ocho entradas y tres salidas.
15. Realizar la tabla de verdad de una función OR exclusivo (XOR) de 4 variables (A, B, C, D). Luego,
- Obtener la mínima suma de productos e implementarla con compuertas **NAND** e **INV**.
 - Implementar la función obtenida
 - Empleando 3 compuertas **XOR** de 2 entradas.
 - Utilizando compuertas **NAND** con salida *open collector*, invirtiendo el OR cableado.
16. Emular los siguientes dispositivos:
- Un flip-flop **S-R** a partir de un flip-flop **D**.
 - Un flip-flop **J-K** a partir de un flip-flop **D**.
 - Un flip-flop **D** a partir de un flip-flop **T**.
 - Un flip-flop **J-K** a partir de un flip-flop **T**.
 - Un flip-flop **D** a partir de un flip-flop **J-K**.
17. Implementar un contador binario sincrónico módulo 10 (esto es, de 0 a 9) a partir de los siguientes componentes:
- Con flip-flops individuales tipo **J-K**, controlando sus entradas.
 - Con un contador de cuatro etapas *up-down* con carga en paralelo.

¹*LogiSim* es una herramienta educativa libre (GPL) para diseñar y simular circuitos lógicos digitales. Puede descargarse desde la página web de la materia: http://cs.uns.edu.ar/~pmd/ac_ing.

18. Utilizando *LogiSim* implemente un contador binario módulo 8 cuya evolución esté controlada por un botón. A su vez, la implementación debe contar con un botón adicional que permita restaurar el estado original del contador. Monitoree la salida del mismo en aras de verificar el correcto funcionamiento del circuito.
19. Simular el ejercicio 17b utilizando *LogiSim*.
20. Implementar el comportamiento de un semáforo utilizando *LogiSim* y los siguientes componentes:
 - Tres diodos emisores de luz (LED) para modelar las luces del semáforo.
 - Una memoria ROM para almacenar la información que será enviada a los LEDs.
 - Un contador binario para direccionar la memoria. Considere la utilización de un *botón* que funcione como *reset* del contador.
 - Un reloj con un período de 1 *seg* para forzar la evolución del contador.
21. Verificar el funcionamiento de los circuitos del ejercicio 16 utilizando *LogiSim*.
22. Un *contador Moebius* (esto es, un contador que modifica un solo bit por cada cambio de estado), puede ser fácilmente implementado usando un registro de desplazamiento. Teniendo esto en cuenta:
 - a) Bosquejar un *contador Moebius* módulo seis a partir de un registro de desplazamiento implementado con flip-flops **J-K**.
 - b) Desarrollar las seis señales periódicas de salida.
 - c) Verificar que dividen el período de partida en seis intervalos iguales y que cada señal tiene un *duty cycle* de exáctamente 50%.

OBS: En un tren de pulsos ideal (uno compuesto de pulsos rectangulares), el *duty cycle* es la razón entre la duración del pulso y el período del mismo. Por caso, para un tren de pulsos en el que la duración del pulso es de $1\mu s$ y el período es de $4\mu s$, el *duty cycle* resulta ser del 25%.
 - d) Indicar como se decodifican los estados 001, 110, 000 y 111.
 - e) Suponiendo que la transición del estado 011 al estado 001 recién deba realizarse cuando aparezca un 1 en una cierta señal *e* (esto es, *e* hace las veces de señal de *enable* para esa transición), ¿cómo hay que modificar al contador para que contemple a la señal *e*?
 - f) Modificar el control sobre la transición entre estados propuesta en el inciso anterior para que la señal *e* ahora regule el paso del estado 110 al estado 111.