República Bolivariana de Venezuela

Universidad de Carabobo

Facultad de Ingeniería

Escuela de Ingeniería Eléctrica y de Telecomunicaciones

Departamento de Sistemas y Automática

Cátedra de Lógica Digital

**Práctica # 2 – Sesión Única**

**Objetivo: Implementar circuitos combinacionales con salidas múltiples utilizando VHDL y la tarjeta de desarrollo BASYS2**

Sección #05 de Laboratorio

**Integrantes:**

Carlos Hernández

C.I.: 25.829.471

Gianfranco Gasbarri

C.I.: 26.654.860

Fecha de entrega: 12/02/19

**Pre-Laboratorio**

1. **Responda las siguientes preguntas:** 
   1. **¿Cuántos displays 7-segmentos existen?** Existen 4 displays
   2. **¿Qué tipo de display tiene la tarjeta?** Ánodo común
   3. **¿Cuál es la nomenclatura de las señales asociadas con los displays?** De izquierda a derecha, AN1-AN4
   4. **¿Qué valor lógico se usa para encender los segmentos?** 0
   5. **¿Qué valor lógico se usa para activar un display específico?** 1
   6. **¿Qué procedimiento debe realizarse para mostrar un dígito cualquiera en los displays existentes?**

Una vez identificado el tipo de display que se va a utilizar (en nuestro caso, el ánodo común). Esto supone que cada segmento se va a encender con un valor “0” lógico y va a estar apagado en caso contrario.

Se debe realizar la tabla de la verdad de cada uno de los segmentos para las entradas de cada uno de los bits que conforman la palabra BCD de entrada (4 bits) correspondientes a los valores del 0 al 9 en decimal.

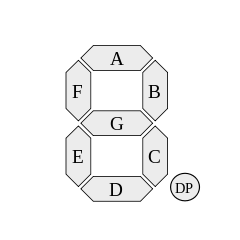
Una vez realizada, ya tenemos cada segmento asociado a todas las posibles combinaciones y ya podremos mostrar un dígito en un display.

1. **Diseño del conversor CDB-7Seg:**
   1. **Construya la tabla de la verdad para cada segmento, obtenga la expresión simplificada y su representación con compuertas lógicas:**

Ver **Anexo 1**

**Nota:** se seguirá la convención universal para la numeración de

cada uno de los segmentos, tal como se indica en la siguiente figura



* 1. **Dibuje y simule el diseño del conversor utilizando LogicWorks. Imprima el diseño:**

Ver **Anexo 2**

**Post-Laboratorio**

* **Diagramas de bloque, programa VHDL, simulación y tablas de la verdad (Lab#2):**

Ver **Anexo 4**

**Conclusiones**

En la presente práctica, se desarrollaron varios componentes correspondientes a los conversores de números binarios de 4 bits (ya sea correspondiente a una palabra BCD válida o a cualquiera de los símbolos del sistema hexadecimal) a un display 7 segmentos.

Al diseñar el primero de ellos, pudimos observar que al proporcionar una entrada entre 0 y 9, los segmentos seguían el comportamiento que se definió al armar la función asignando manualmente sus valores en la tabla de la verdad. Pudimos observar que el componente del inversor tenía como salidas 7 funciones lógicas distintas, cada una de ellas dependiendo de las entradas D-C-B-A.

Además, nos percatamos de que en varias de dichas funciones, se podía duplicar alguna expresión, la cual podíamos aprovechar para reutilizar varias veces, simplificando de manera significativa el diseño. Al hacer esto, las combinaciones restantes que no representan una palabra BCD válida (10-15) generaban salidas diferentes a las generadas cuando no se simplificaba. Esto se debe a que fueron definidas como **don’t care** y se adaptaron al proceso de simplificación automático.

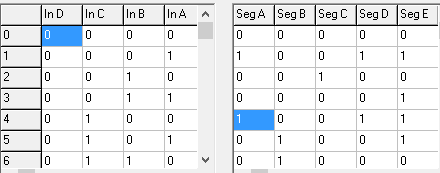
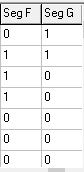
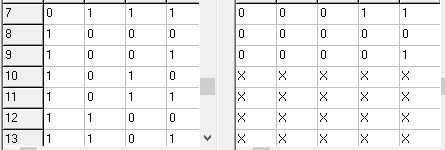
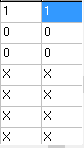
El diseño realizado con el software LogicWorks y el descrito con VHDL comparten varias similitudes. En ambos, las entradas y las salidas son las mismas, y al realizar las simulaciones las salidas fueron las esperadas para los valores definidos (del 0 al 9). Sin embargo, para las salidas correspondientes a las combinaciones restantes variaban debido a que en el diseño descrito con VHDL, las mismas se establecieron de manera que pintaran un guión (“-“). Esto quiere decir: los segmentos del A al F en 1 y el G en 0.

Otra diferencia notable en el momento de realizar el diseño, es que en el software LogicWorks se debió representar cada uno de los circuitos lógicos para las 7 funciones, lo cual representa una actividad laboriosa. En cambio, la definición por VHDL simplificaba esto, ya que simplemente se debía expresar las 7 salidas concatenadas para cada uno de las posibles entradas.

**ANEXOS**

**Anexo 1**

**Anexo 1.1**

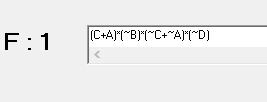
**Tabla de la verdad general**

**Anexo 1.2**

**Segmento A**

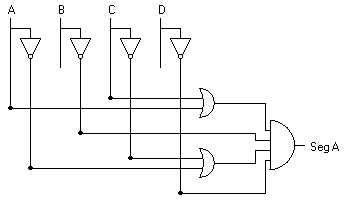
**Anexo 1.2.1**

**Expresión POS simplificada**



**Anexo 1.2.2**

**Representación con compuertas lógicas**

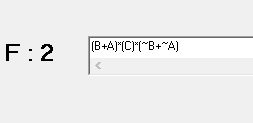


**Anexo 1.3**

**Segmento B**

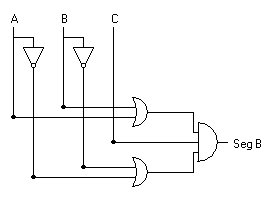
**Anexo 1.3.1**

**Expresión POS simplificada**



**Anexo 1.3.2**

**Representación con compuertas lógicas**

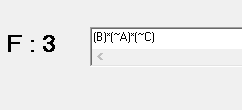


**Anexo 1.4**

**Segmento C**

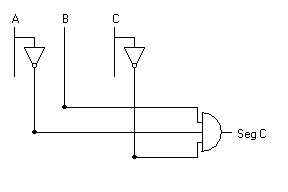
**Anexo 1.4.1**

**Expresión POS simplificada**



**Anexo 1.4.2**

**Representación con compuertas lógicas**

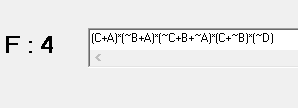


**Anexo 1.2**

**Segmento D**

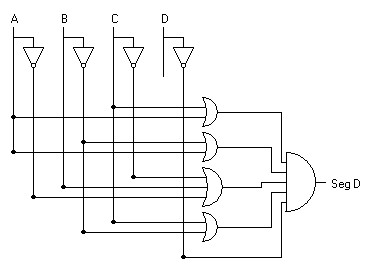
**Anexo 1.5.1**

**Expresión POS simplificada**



**Anexo 1.5.2**

**Representación con compuertas lógicas**

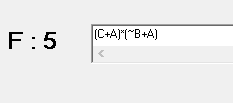


**Anexo 1.6**

**Segmento E**

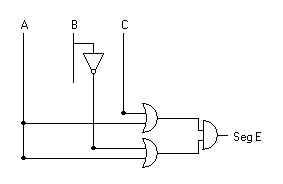
**Anexo 1.6.1**

**Expresión POS simplificada**



**Anexo 1.6.2**

**Representación con compuertas lógicas**

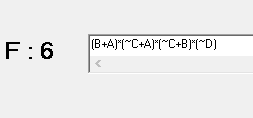


**Anexo 1.7**

**Segmento F**

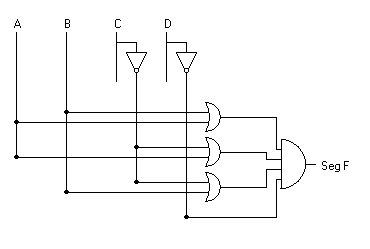
**Anexo 1.7.1**

**Expresión POS simplificada**



**Anexo 1.7.2**

**Representación con compuertas lógicas**

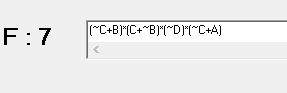


**Anexo 1.8**

**Segmento G**

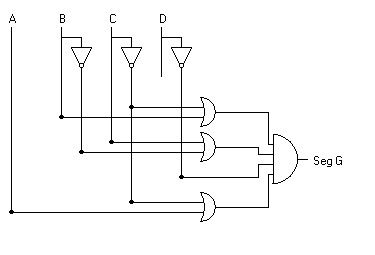
**Anexo 1.8.1**

**Expresión POS simplificada**



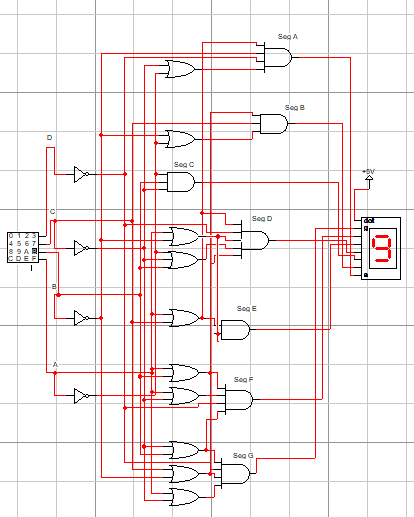
**Anexo 1.2.2**

**Representación con compuertas lógicas**



**Anexo 2**

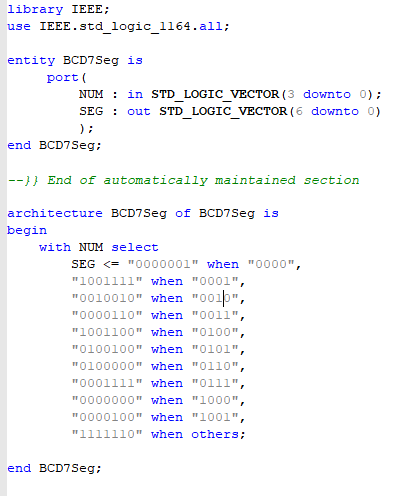
**Diseño del conversor en LogicWorks**

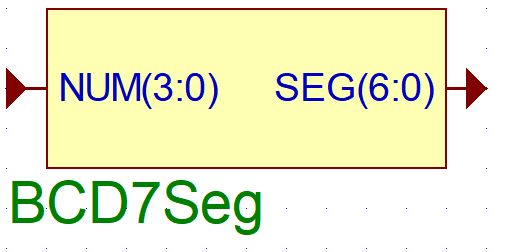


**Anexo 3**

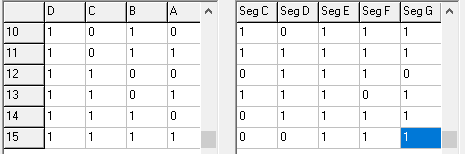
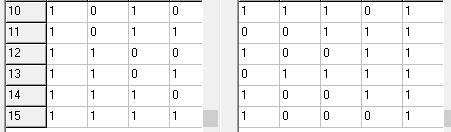
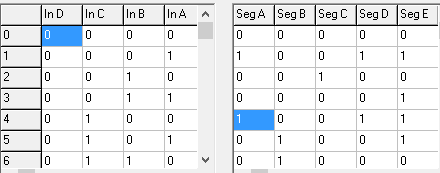
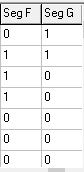
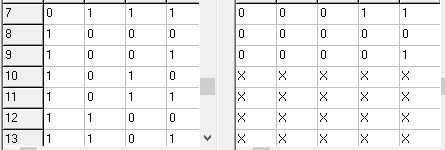
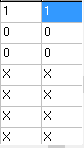
**Anexo 3.1**

**Código y símbolo de BCD7Seg**

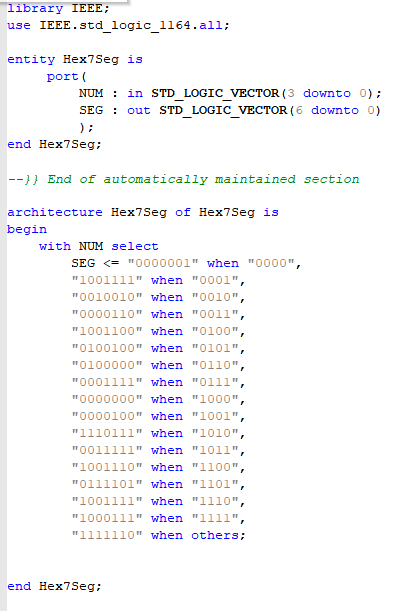
****

****

**Anexo 3.2.1**

**Tabla de la verdad de Hex7Seg**

**Anexo 3.2.2**

**Símbolo y código de Hex7Seg**

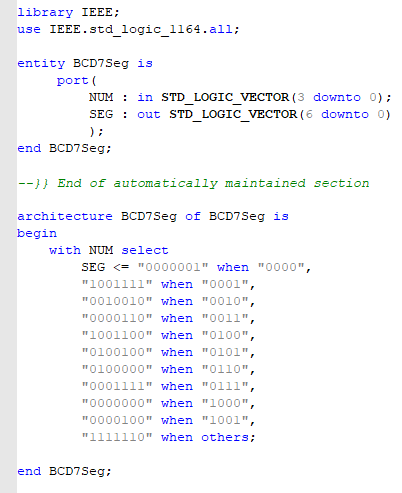
**Anexo 4**

**Anexo 4.1**

**Diagramas de bloque**

**Anexo 4.2**

**Código VHDL**



**Anexo 4.3**

**Tablas de la verdad**

**Anexo 4.4**

**Simulación**