SimpFunLogQM: UNA HERRAMIENTA PARA EL ENTRENAMIENTO EN LA SIMPLIFICACIÓN E IMPLEMENTACIÓN DE FUNCIONES LÓGICAS

F. GARCIA-LAGOS Y G. JOYA

Departamento de Tecnología Electrónica. Escuela Técnica Superior de Ingeniería de Telecomunicación. Universidad de Málaga. 29071-Málaga. España.

En este trabajo se describe el programa SimpFunLogQM. Se trata de un programa de entrenamiento para la simplificación e implementación de funciones lógicas. Sus principales objetivos son su utilización como herramienta de trabajo para el aprendizaje del método de simplificación de funciones lógicas Quine-McCluskey mediante la simulación de sus etapas, el análisis de riesgos de una implementación de una función lógica y su propuesta de implementación para evitarlos y la implementación de una función lógica mediante bloques funcionales multiplexores. El programa se ha diseñado para el sistema operativo Windows y se encuentra accesible libremente en http:/face1.dte.uma.es.

1. Introducción

La herramienta presentada en este trabajo, el programa SimpFunLogQM, está orientada al entrenamiento de estudiantes de diseño lógico en la simplificación de funciones mediante el método de Quine-McCluskey, el análisis de riesgos de una implementación particular y su eliminación y la implementación de funciones lógicas mediante bloques funcionales multiplexores. Entendemos que estas tareas, aunque no de manera exclusiva, contribuyen de manera importante a la consolidación de los conocimientos sobre diseño lógico y al desarrollo de la destreza en la implementación de funciones. Por otra parte, el método de simplificación Quine-McCluskey la herramienta utilizada en los problemas reales de simplificación además de ser el método alternativo adecuado a los mapas de Karnaugh cuando el número de variables independientes de la función es mayor que 5.

El programa está diseñado para ejecutarse en el sistema operativo Windows y se manejo resulta amigable gracias a la utilización de diferentes ventanas y controles estándares típicos del sistema operativo. Actualmente, está siendo utilizado en fase de prueba por los alumnos de Laboratorio de Electrónica Digital de las titulaciones de Ingeniería Técnica de Telecomunicación de la E.T.S. de Ingenieros de Telecomunicación (Universidad de Málaga) y está disponible, junto con su manual de usuario, en la sección “software” de la dirección de Internet http:/face1.dte.uma.es [1] .

En lo que resta de este artículo se describe de manera escueta las distintas utilidades que componen el programa y se muestra un ejemplo de aplicación concreto.

2. Características principales del programa

El programa se ha diseñado para presentar una interfaz de usuario amigable y fácil de utilizar. Sus funcionalidades se encuentran accesibles a través del menú principal, que permite entre otras acciones especificar la función lógica que debe ser simplificada o implementada y determinar la forma final de la expresión simplificada. A continuación se comentan las opciones más importantes del programa.

2.1.- Especificación, salvaguarda y recuperación de funciones lógicas.

El primer lugar se debe especificar la función lógica con la que se va a trabajar. Esto se puede hacer de dos formas distintas: recuperando una función guardada en disco o especificando, a través de un cuadro de diálogo, una nueva función. Con la opción de especificar una función aparece el cuadro de diálogo de la figura 1. Los campos que deben especificarse son los siguientes: un nombre genérico para la función, las etiquetas de cada una de las variables independientes y los mintérminos y términos indiferentes de la misma. Estos últimos campos pueden especificarse mediante una expresión lógica seleccionando la opción correspondiente. La función lógica actual puede salvarse en disco para su uso posterior. Una vez especificados todos los campos, el programa comprobará que estos son coherentes, rechazándolos en el caso de alguna incoherencia o error. En caso contrario, la función especificada se muestra en la ventana principal. Con la opción de abrir una función lógica guardada, el usuario únicamente deberá especificar el nombre del fichero. El programa comprueba que el formato de los datos es correcto y la definición de función coherente y rellena de forma automática el cuadro de diálogo de la figura 1.

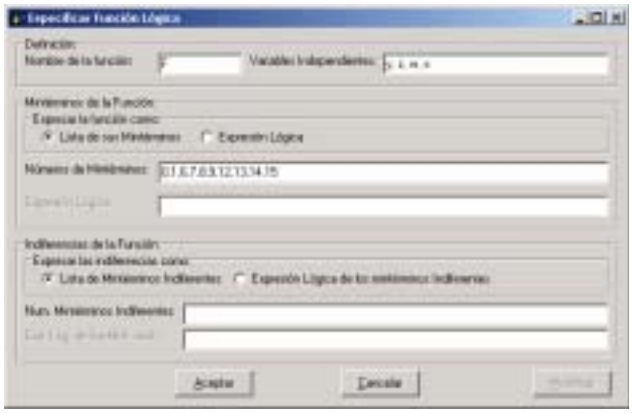

Figura 1: *Cuadro de diálogo que muestra los campos requeridos para especificar una función lógica. Esta misma ventana se utiliza para especificar una nueva función lógica o*

2.2.- Simplificación de la función lógica.

La funcionalidad principal del programa es simplificar la función lógica activa siguiendo el método de Quine-McCluskey [2,3].En el proceso de simplificación el programa presenta de forma tabulada los resultados intermedios y finales, es decir, la lista de implicantes de cada iteración del algoritmo, la lista de implicantes primos, la lista de implicantes primos esenciales y la cobertura de la función. La cobertura de la función es convertida a una expresión lógica, que se presenta al usuario en un cuadro de texto y constituye la solución obtenida por el algoritmo.. Esta expresión aparecerá como una suma de productos o como un productos de sumas dependiendo de la elección de usuario.

2.3.- Análisis de riesgos de la función lógica.

La aplicación dispone de una opción para buscar y eliminar los riesgos (*glitches o azares*) de una implementación propuesta. Mediante un cuadro de diálogo se muestran aquellos términos

de la expresión mínima que potencialmente pueden generar el comportamiento transitorio no deseado y se sugieren los términos que pueden incluirse para eliminarlo.

2.4.- Implementación de funciones lógicas con bloques funcionales multiplexores.

Esta opción tiene como objetivo el análisis de las diferentes posibilidades de implementar la función lógica actual mediante bloques multiplexores [4]. El tipo de multiplexor con el que se desea hacer la implementación se especifica genéricamente indicando el número de variables de control del mismo. Una vez especificado el tipo de multiplexor, el usuario tiene la posibilidad de especificar las variables que se utilizarán como variables de control o bien dejar al programa que busque la elección óptima para esta función particular. En este último caso, se muestra una lista con todas las posibles combinaciones de variables de control, junto con el coste de dicha implementación. El programa selecciona, de forma automática, la implementación de coste mínimo. La función de coste puede modificarse para ser adaptada a diseño particulares. Adicionalmente, el programa reordena la tabla de verdad de la función lógica actual de forma que el usuario puede fácilmente analizar cómo se han obtenido las expresiones lógicas de cada una de las entradas de dato del multiplexor en la implementación realizada.

3. Ejemplo de utilización del programa

En esta sección mostramos, a través de un ejemplo, las diferentes utilidades del programa. La función lógica para este ejemplo se define, como suma de mintérminos, por la ecuación (1):

$$F(y,z,w,x) = \sum_{4} m(0,1,6,7,8,9,12,13,14,15) \qquad (1)$$

En primer lugar debemos rellenar el cuatro de diálogo "Especificar Función Lógica", para reflejar los parámetros de la función. La figura 1 muestra el valor de cada cuadro de texto en este ejemplo. A continuación el programa comprueba que los datos son coherentes y muestra, en la ventana principal, la función F en forma de tabla de verdad. A partir de este momento las opciones del menú correspondientes a las herramientas de minimización, análisis de riesgos e implementación con multiplexores se habilitan.

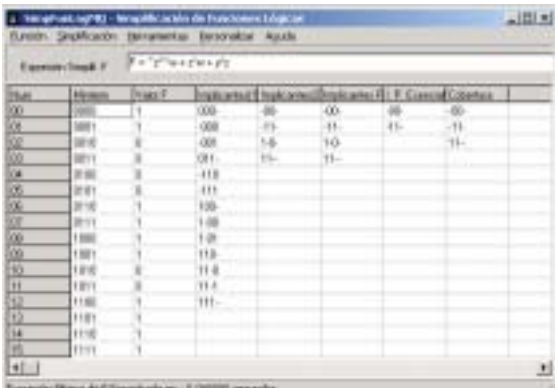

Figura 2. *Implicantes, implicantes primos, implicantes primos esenciales y cobertura de la función, obtenidos en el*

A través de la opción "Simplificar", el programa aplica el algoritmo de Quine-McCluskey para obtener la expresión mínima. Dependiendo de si se ha seleccionado "Obtener Suma de Productos" u "Obtener Producto de Sumas", el programa genera una de las soluciones siguientes: $\bar{z} * \overline{w} + z * w + y * \overline{w}$ ó $(z + \overline{w}) * (y + z + w)$. Además de la expresión mínima, en la ventana principal aparecen ordenados los implicantes, implicantes primos, implicantes primos esenciales y la cobertura (mínima) de la función, tal como muestra la figura 2.

Para comprobar si la función minimizada contiene riesgos, seleccionamos la opción del menú "Analizar Riesgos" apareciendo el cuadro de diálogo mostrado en la figura (3). Como la función presenta riesgos, la ventana muestra los implicantes primos que son problemáticos

(cuadro de la izquierda) y los términos productos que hay que añadir a la expresión simplificada de la función para evitarlos (cuadro de la derecha). Para este ejemplo, habrá que añadir el término producto $y * \overline{w}$.

Para implementar la función con un multiplexor, por ejemplo, de dos variables de control, en el cuadro de texto "Número de VC del Multiplexor" especificamos un 2. Seleccionando la opción "Selección Automática de las VC" el programa analiza el coste de cada combinación posible de variables de control (según la función de coste actual) y muestra el resultado en el cuadro de diálogo, tal y como muestra la figura 4.

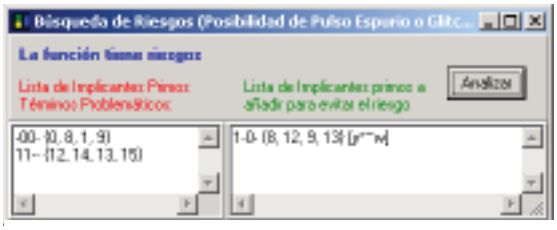

Figura 3. *Resultado del análisis de riesgos de la función del ejemplo.*

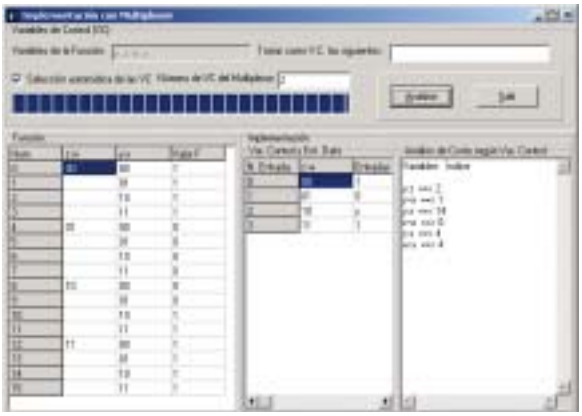

Figura 4. *Ventana con el resultado del análisis de coste de diferentes elecciones en las variables de control (derecha) y la*

4. Conclusiones

Este artículo presenta una herramienta orientada tanto al entrenamiento de estudiantes de diseño lógico como a la ayuda en el diseño e implementación práctico. Así, presenta las utilidades de minimización de funciones por el método de Quine-McCluskey, el análisis de azares y la implementación de funciones mediante bloques funcionales multiplexores. Para cada una de éstas utilidades no sólo se presenta el resultado final, sino que para cada proceso se presenta de forma tabulada una serie completa de pasos intermedios, permitiendo así un mayor nivel de conocimiento interno de los algoritmos utilizados. La aplicación se ha diseñado con una interfaz amigable que hace uso de controles adecuados para la presentación de la información generada, y está siendo utilizada en fase de prueba por los estudiantes de Ingeniería Técnica de Telecomunicación de la E.T.S.I. de Telecomunicación de la Universidad de Málaga.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Departamento de Tecnología Electrónica de la Universidad de Málaga. Agradecemos la colaboración y comentarios de los estudiantes que han utilizado el programa hasta la fecha.

Referencias

[1] FACE (Foro de Aprendizaje Cooperativo en Electrónica). http:/face1.dte.uma.es.
[2] J. McCluskey. *Introduction to the Theory of Switching Circuits*. McGraw-Hill (1965).
[3] J. F. Wakerly. *Digital design. Principles and practices.*. Prentice Hall (2000).
[4] F. García-Lagos, G. Joya, J. A. Rodríguez y R. Ron, *Problemas de Electrónica Digital.* Colección manuales de la Universidad de Málaga, SPICUM (2001).