PROYECTO VERILOG

Indicaciones Generales.

1. El proyecto Verilog requiere como entregables dos partes:

    i. El informe escrito.
    ii. Los códigos fuente.

2. Son requisitos de admisibilidad del proyecto:

    i. Que se presente el informe escrito completo.
    ii. Que se presenten todos los códigos fuente requeridos debidamente completados y que compilen apropiadamente en iverilog.

3. Si un proyecto no satisface los requisitos de admisibilidad el mismo no se revisará y se asignará una nota de cero.

4. El informe debe ser presentado de una manera ordenada y clara. Puede hacerse en manuscrito o con un procesador de texto. En cualquier caso se debe entregar en formato pdf.

5. Los códigos fuente serán compilados en iverilog y la persona estudiante debe asegurarse que dichos códigos compilen adecuadamente en este compilador HDL. Todo los códigos deben estar debidamente documentados, con los respectivos comentarios en las declaraciones de las estructuras de datos y en los encabezados, de tal manera que sea fácil de leer, entender y reutilizar. En el encabezado de cada archivo de código se debe incluir su nombre, número de carné y la fecha de entrega.

PROBLEMA #1

Es posible diseñar un generador del bit de paridad, sea paridad par o impar, para una palabra de L bits mediante una solución estructural. Para ello se pueden utilizar tres módulos funcionales en una estructura de M niveles. Los módulos funcionales se denominan ONES_COUNT, PARITY_GEN y PARITY_SOLVER, cuyas variables terminales se muestran en la Figura #1, donde:

i. El Módulo ONES_COUNT pone en sus salidas (w_i,w_j) el valor decimal de la cantidad de entradas activas.
ii. El Módulo PARITY_GEN activa su salida si la cantidad de entradas activas, proveniente de un módulo ONES_COUNT antecesor más el bit x_K de la palabra de

entrada satisface la paridad.

iii. El Módulo PARITY_SOLVER genera el bit de paridad R a partir de dos Módulos PARITY_GEN antecesores y el bit x_L de la palabra de entrada

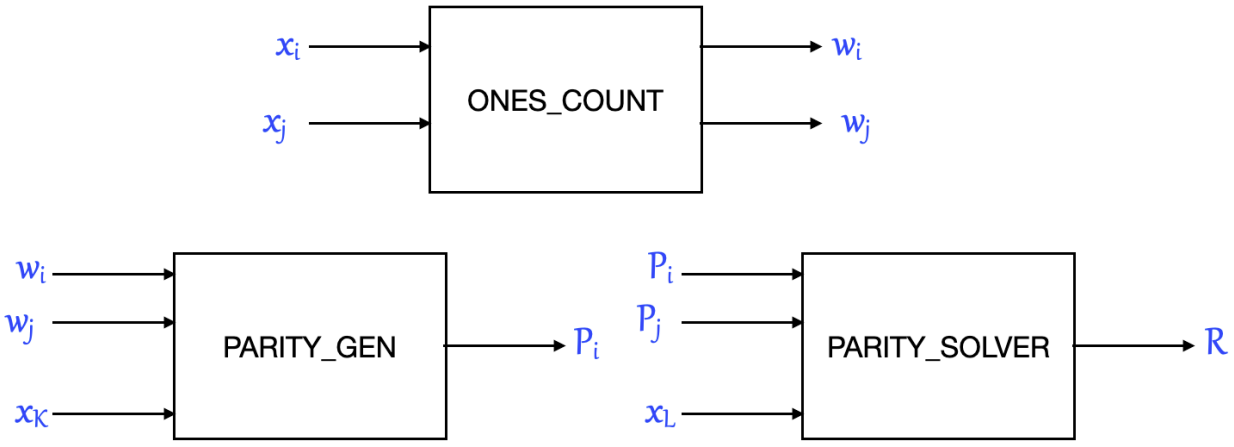

FIGURA #1
Diagrama de bloques de los módulos a diseñar.

A partir se estos módulos es posible encontrar una solución estructural para una palabra de L bits, siendo que para diferentes valores de L la solución estructural será diferente. Entonces se debe:

a. Encontrar las ecuaciones lógicas para los circuitos ONES_COUNT, PARITY_GEN y PARITY_SOLVER que resuelva para paridad PAR. Se deben incluir todos los M.K. que sustenten la solución.
b. Desarrollar los módulos en Verilog para los circuitos ONES_COUNT, PARITY_GEN y PARITY_SOLVER. El módulo ONES_COUNT debe ser resuelto con multiplexores 4x1 y debe *instanciar* el módulo MUX4x1_Case visto en clase. Los módulos PARITY_GEN y PARITY_SOLVER se deben implementar con un nivel de abstracción funcional.
c. Diseñar una arquitectura llamada PARITY_SEVEN, que describa la estructura de un generador de paridad para una palabra de 7 bits $x=x_6x_5x_4x_3x_2x_1x_0$ Dicha arquitectura debe estar basada en los circuitos ONES_COUNT, PARITY_GEN y PARITY_SOLVER y debe generar un bit de paridad par denominado R.
d. Desarrollar un módulo en Verilog que describa la arquitectura PARITY_SEVEN. La entrada de este módulo es el ***vector*** de 7 bits *X* y la salida es el escalar R.

En la sección del informe denominada Problema #1 debe incluir dos subsecciones:

1. Diseño de bloques. Esta sección debe incluir todo el procedimiento de diseño de la lógica combinacional para los circuitos ONES_COUNT, PARITY_GEN y PARITY_SOLVER.
2. Diseño de la arquitectura. En esta sección se debe presentar el diagrama topológico de la arquitectura diseñada para PARITY_SEVEN, en tal arquitectura se deben indicar los *nets* necesarios para la descripción en Verilog. Se debe incluir una explicación de la solución.

Los entregables del problema #1 son:

1. El informe con las secciones especificadas.
2. Los códigos fuente en Verilog para los módulos, ONES_COUNT, PARITY_GEN, PARITY_SOLVER y PARITY_SEVEN. Cada uno de los módulos *instanciados* debe aparecer en archivos separados. Toda *instanciación* debe realizarse por descripción nombrada.

Nota: Para la revisión de este problema se ha implementado un testbench que utiliza los módulos de su solución, además del módulo MUX_4X1_Case. Preste atención a los nombres de los módulos y de las variables así como los tipos de estructuras de datos. Si su solución realiza un cambio a la especificación NO compilará en el testbench y se entenderá que no funciona. En ningún caso se harán ajustes o modificaciones al código entregado en procura de que compile.

PROBLEMA #2.

En un proceso industrial se cuenta con una mezcladora donde se obtiene un producto intermedio en el proceso. La mezcladora recibe tres sustancias provenientes de tres almacenadores A1, A2 y A3, que son dosificadas por medio de tres electroválvulas V1, V2 y V3. Adicionalmente la mezcladora cuenta con un motor para el eje de mezclado (M), dos sensores de peso P1 y P2, una Bomba de Vaciado (B), un Relé Temporizador (RT) y una Unidad de Control Digital (UCD). Además se cuenta con un botón de inicio que envía una señal denominada IN para iniciar el subproceso de mezclado. El diagrama funcional del subproceso de mezclado se muestra en la Figura #2.

El Relé Temporizador (RT) recibe una entrada de selección T para cargar uno de dos tiempos a contar: T1 (T=0) o T2 (T=1) y una señal de inicio denominada S. Cuando el temporizador recibe la señal de S inicia el conteo de tiempo (T1 o T2) según lo indicado en su entrada T y activa la salida TOK cuando finalice el tiempo indicado. Dicha señal es enviada a la Unidad de Control Digital de la mezcladora. La señal T debe estar activa siempre con el valor del tiempo a contar. La señal S debe ser un pulso con una duración de medio ciclo del reloj, iniciando en el inicio del estado. Cada vez que se da una orden de inicio por S el RT borra la salida TOK.

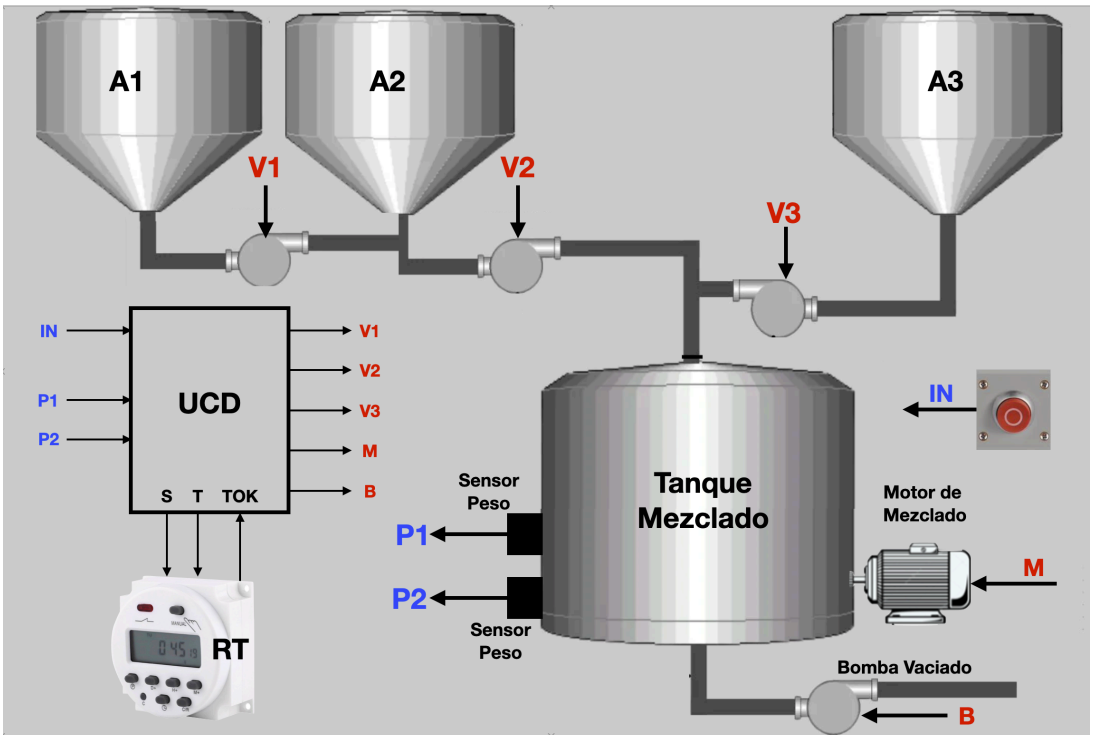

Figura #2.
Diagrama del proceso de mezclado.

La Unidad de Control Digital (UCD) debe operar de acuerdo a los siguientes requerimientos funcionales del proceso.

i. Al inicio del proceso el tanque de mezclado está vacío y se espera que se presione el botón de Inicio para empezar el procesamiendo de una tanda de mezclado.
ii. Al recibirse la señal de inicio (IN) se deben activar las electroválvulas V1 y V2 para la dosificación de las dos primeras sustancias.
iii. Las sustancias provenientes de los almacenadores A1 y A2 ingresarán al tanque de mezclado hasta que se active la señal del sensor de peso correspondiente P1 para V1 y P2 para V2, momento en el que se deben desactivar las electroválvulas V1 y V2, respectivamente. Los valores de P1 y P2 no necesariamente se darán de manera simultánea, pudiendo activarse primero P1 o primero P2.
iv. Cuando se alcance el peso P1 debe iniciarse el proceso de mezclado arrancando el motor de la mezcladora M. Observación: Note que el proceso de mezclado puede iniciarse aún con V2 abierta pues el arranque de M no depende de que se haya alcanzado el peso P2.
v. El motor de la mezcladora debe operar por un tiempo T1, para ello se debe seleccionar dicho tiempo por medio de T y enviar la señal de S en cuanto se arranque el motor. Observación: El peso P2 se alcanzará a lo más durante la corrida del tiempo T1.

vi. Al finalizar T1 se debe activar la electroválvula V3 para el ingreso de la sustancia del almacenador A3. V3 debe activarse en el inicio retardado del estado como una salida de Mealy.
vii. Una vez activa V3 se debe cargar el Relé Temporizador con el tiempo T2 y activar la señal S, manteniendo activo el motor de la mezcladora.
viii. Al finalizar el tiempo T2 se debe parar el motor de mezclado, se desactiva V3 y se activa la Bomba de Vaciado (B). Dicha bomba estará activa hasta que se desactiven los sensores P1 y P2 indicando que se ha vaciado la mezcladora y que puede darse inicio a una nueva tanda de mezclado.

Para la Unidad de Control Digital se requiere:

a. Describir la máquina mediante un diagrama ASM.
b. Desarrollar un módulo en Verilog, llamado Mezcladora que describa esta máquina clase 4.
c. Desarrollar un Testbench para comprobar el módulo desarrollado. Para este Testbench se deben diseñar los vectores de prueba que puedan recorrer la máquina por todos sus estados.
d. Generar el archivo correspondiente .vcd y realizar el diagrama de tiempos con los resultados del testbench.

En la sección del informe denominada Problema #2 debe incluir tres subsecciones:

a. Diagrama ASM. Se debe incluir una explicación general del diagrama ASM, incluyendo la definición de estados, la asignación de estados y la declaración de las variables de estado.
b. Diseño de los vectores de prueba. Para cada vector de prueba debe explicarse porqué se eligió (a qué estado cambia la máquina con él y que salidas se activan cuando corresponda)
c. Diagramas de tiempo. Se debe incluir una captura del diagrama de tiempo resultante de la simulación. Dicho diagrama de tiempo debe ser generado en Gtkwave y debe incluir al menos: la señal de reloj y de reset, todas las entradas primarias y salidas primarias de la máquina y el vector de estado presente y el vector de próximo estado. Adicionalmente se debe incluir un análisis detallado del diagrama de tiempo donde se concluya que la simulación de la máquina de estados descrita por el módulo Mezcladora opera de acuerdo a todos los alcances de su definición.

Los entregables son:

a. La sección del informe del Problema #2 con las subsecciones requeridas.
b. Los códigos fuente del module Mezcladora y del testbench desarrollado.

Nota: Si el testbench carga los vectores de prueba desde un archivo se debe incluir el archivo que contiene los vectores de prueba.