EIE

Escuela de
Ingeniería Eléctrica

Universidad de Costa Rica

Ingeniería eléctrica

Circuitos digitales II

IE-0523

Tarea dos

Estudiante:

Brenda Romero Solano

Carné:

B96953

Profesor:

Ana Eugenia Sánchez Villalobos

Fecha de entrega:

12/05/2024

Primer ciclo

2024

Resumen

En este trabajo, se diseñó un módulo que permite operaciones matemáticas básicas controlado por un reloj y señales de habilitación y reinicio. El usuario puede seleccionar el modo de operación a través de las entradas del módulo, y el resultado se genera en la salida correspondiente.

Para verificar el correcto funcionamiento de la calculadora, se diseñaron pruebas específicas que se ejecutan mediante un archivo .v llamado calculadora_tester.v. Este archivo inicializa los valores de entrada y activa números aleatorios en las entradas de los operandos, observando los resultados en la salida correspondiente (c). Además, se implementó el comando "$display" para obtener una previsualización del resultado en la consola.

En la sección de análisis de resultados, se examinan detalladamente los resultados obtenidos de cada prueba realizada. Se inicia con la inicialización de las señales y se continúa con pruebas específicas de suma, resta, multiplicación y "left shift". Se observa el comportamiento del sistema en diferentes situaciones, incluyendo casos límite para evaluar su robustez.

Finalmente, se realizó una comparación entre el diseño original y el sintetizado utilizando la librería proporcionada "cmos_cells". Aunque se detectó un cambio en el valor de una de las señales durante el primer ciclo de reloj en la simulación sintetizada, se concluyó que no afectaba el funcionamiento global de la calculadora.

Ejecución del programa

Para utilizar el programa, es importante seguir un conjunto de pasos específicos que garantizan su correcto funcionamiento. En primer lugar, el usuario debe asegurarse de tener el archivo ZIP del proyecto y descomprimirlo en el directorio de su elección. Si se está utilizando Windows, es preferible descargar el ambiente OSS CAD SUITE y para correrlo, abrir la terminal desde el archivo start.bat y una vez en este, es necesario moverse a la carpeta de la tarea mediante el comando “cd”, además de asegurarse que la dirección a Makefile esta agregada a la variable de entorno.

El siguiente paso es ejecutar el comando “make”. Este comando automatiza una serie de tareas esenciales para la ejecución del programa. Al utilizar “make”, el programa se compilará automáticamente, lo que significa que se reviran los archivos en busca de problemas. Además, “make” generará el archivo .vcd; que es esencial para el análisis y la visualización de la simulación; el archivo sintetizado en el cual se presentan que compuertas se pueden implementar para solucionar el ejercicio.

Una de las ventajas más destacadas de utilizar “make” es su capacidad para abrir GTKwave de forma automática. GTKwave es una herramienta de visualización que permite inspeccionar y analizar el comportamiento de los circuitos electrónicos simulados. Al abrirlo automáticamente, el usuario puede explorar de manera inmediata los resultados de la simulación, lo que facilita la comprensión del funcionamiento del programa.

Por último, pero no menos importante, “make” se encarga de limpiar los archivos temporales generados durante el proceso de compilación y ejecución. Esto garantiza que el entorno de trabajo esté siempre ordenado y libre de archivos innecesarios, lo que contribuye a una mejor gestión del proyecto en general.

Es importante destacar que, si el usuario lo prefiere, cada una de estas acciones puede llevarse a cabo de forma independiente utilizando las banderas correspondientes al ejecutar el comando “make”. Por ejemplo, utilizar “make compile” para compilar los archivos, “make run” para generar el archivo .vcd, “make view” para visualizar el archivo en GTKwave, “make synth” para sintetizar el módulo del DUT mientras que “make show_jpg” y “make show_svg” para generar una imagen que se pueda apreciar en ambos sistemas operativos, por último, “make clean” permite eliminar los archivos resultantes.
Si se desea comprobar la salida del código sintetizado, se puede usar el mismo testbench que viene en la carpeta de la tarea, mientras se eliminen los símbolos que corresponde a los comentarios de los include, y se comente el que contiene a calculadora.v original.

Esta flexibilidad permite adaptar el proceso según las necesidades y preferencias individuales del usuario. En resumen, seguir estos pasos asegura una experiencia fluida y eficiente al utilizar el programa.

Creación del código

Para poder implementar de forma satisfactoria una calculadora de 8 bits, se creó un módulo el cual puede recibir un máximo de dos números; este se controla por un reloj y señales de habilitación y reinicio. El modo de operación se ingresa a través de las entradas del módulo, mientras que el resultado se genera en la salida. El comportamiento del módulo está definido por un proceso siempre activo que se dispara en cada flanco positivo del reloj. Cuando la señal enb está activada, el proceso selecciona la operación según el modo especificado y calcula el resultado, que luego se asigna a la salida. Si se activa la señal de reinicio, la salida se establece en cero. Si ninguna operación coincide con el modo

especificado, la salida se mantiene. Para simplificar la cantidad de líneas del código se implementó un case evaluando la variable modo para seleccionar que tipo de operación realizará la calculadora. Sin embargo, para mejor la comprensión que pueda tener una persona del código implementado, se crearon parámetros con los nombres respectivos de la operación y el valor que corresponden a la misma.

Para confirmar el correcto funcionamiento de la calculadora, se diseñaron ciertas pruebas, las cuales se escribieron en un mismo archivo .v llamado calculadora_tester.v, en donde, primeramente, se inicializan los valores de las salidas del tester que corresponden a las entradas del DUT, y luego activar con números escogidos aleatoriamente en las salidas de a y b con el fin de observar el valor que tendrá la salida c del DUT, además, se implementó el comando "$display" para tener una previsualización del resultado obtenido desde la consola.

Por último, se implementó un testbench que conectara tanto al DUT como al tester y que además generara el archivo .vcd para la visualización de los waveforms, aprovechando que, al sintetizar el código, este debe funcionar como el código original, solo se añadieron dos líneas de código comentadas para la visualización del waveform una vez realizada la síntesis.

Análisis de resultados

A continuación, se procede a analizar los resultados obtenidos de cada una de las pruebas realizadas, en la figura 1, es posible observar que se aplicó la señal de reset para poder inicializar la salida del DUT, mientras que gracias al initial, las señales de entrada serán 0.

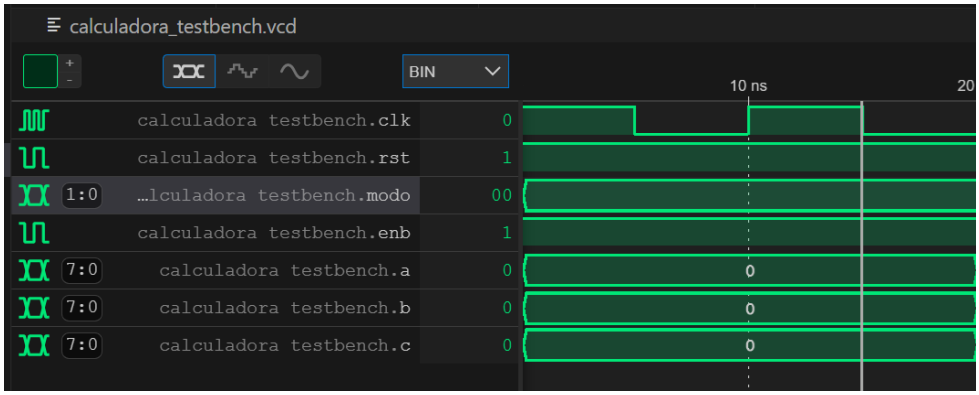

Figura 1.Inicialización de las señales.

Una vez comprobado que las señales se inicializan correctamente, se procedió a verificar que sucede al sumar dos números, donde en la figura 2, durante los 20 ns de la simulación, podemos apreciar que se efectúa correctamente la operación suma, ya que el modo es 00, de los dos números ingresados. Lo anterior también sucede en la figura 3, donde se ingresan otros valores conservando el mismo

modo de sumado y el resultado mostrado en dicha figura es correcto.

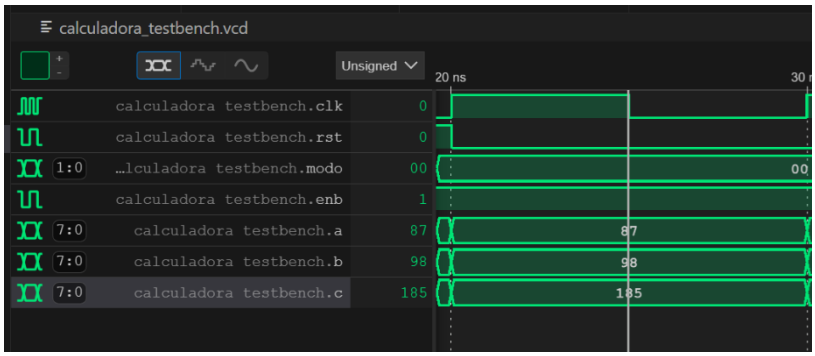

Figura 2. Primera prueba.

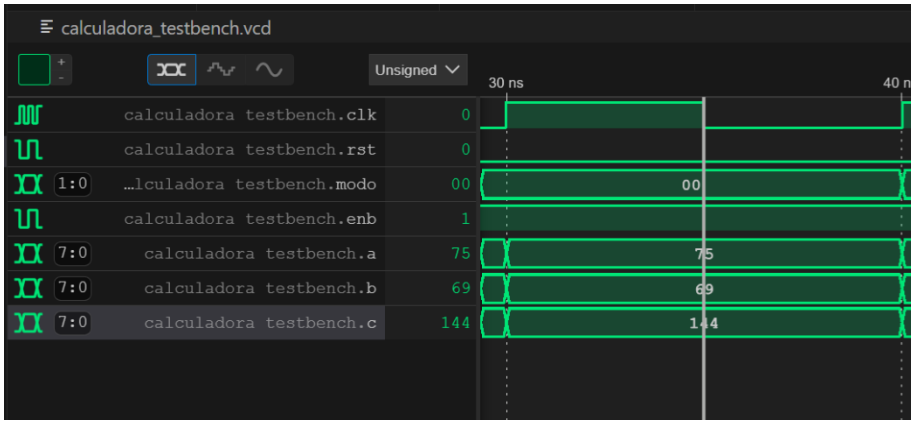

Figura 3.Segunda prueba.

Posteriormente, se verifico que el modo resta, cuyo código corresponde a 01, funcionara correctamente, por lo que para la tercera prueba realizada en la figura 4, se cambió el modo para poder restar y es posible verificar que el resultado mostrado es correcto. Lo anterior también se puede verificar en la figura 5.

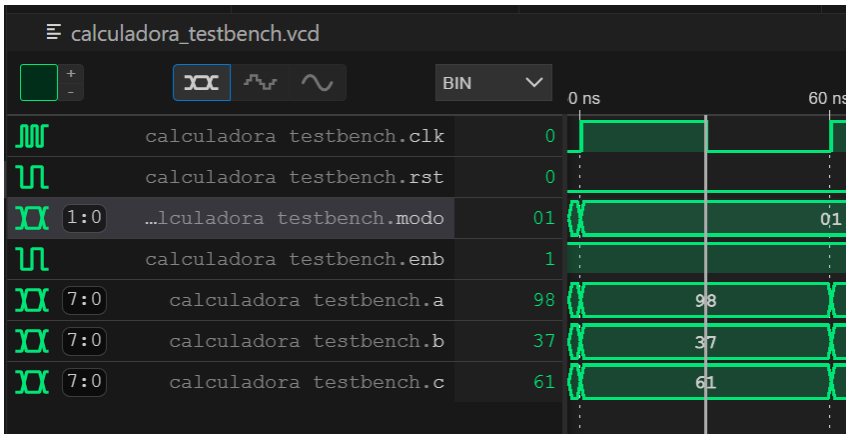

Figura 4. Tercera prueba realizada.

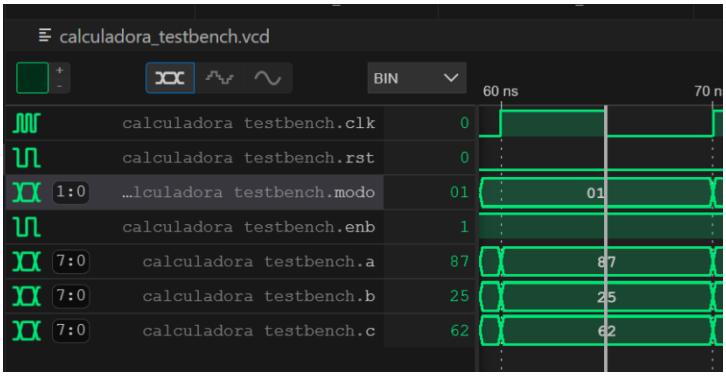

Figura 5. Cuarta prueba del sistema

A continuación, se procedió a verificar el funcionamiento del modo multiplicación donde se ingresaron dos números aleatorios los cuales se aprecian en la figura 6 y 7, en dichas figuras se aprecia que el modo de la calculadora es 10, que corresponde a la operación producto y que el resultado es correcto, además se puede mencionar que, durante la sexta prueba, el resultado de la multiplicación fue el caso limite que evita un posible acarreo.

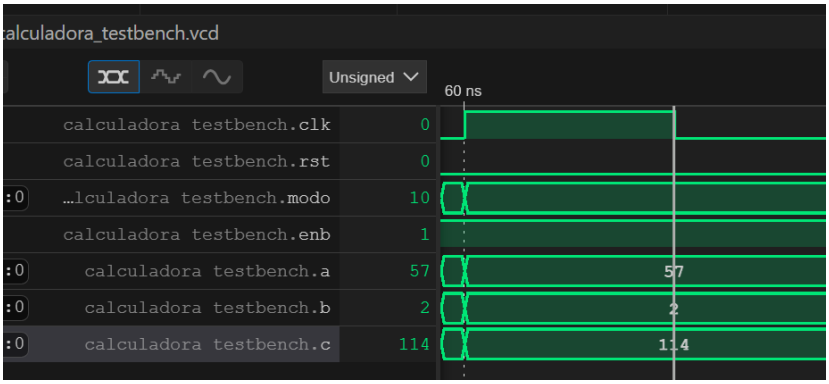

Figura 6. Quinta prueba del sistema

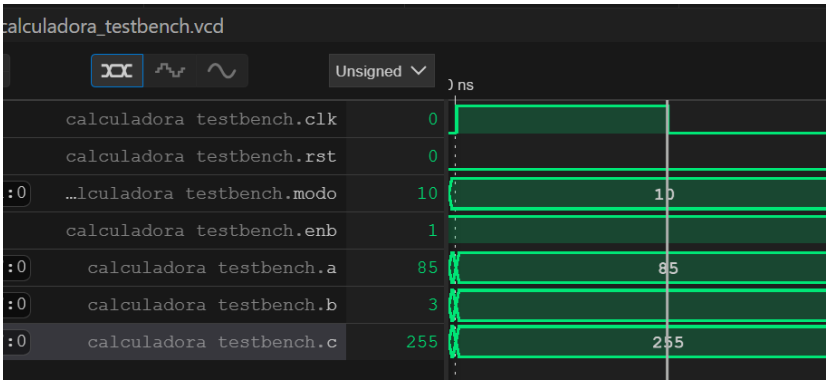

Figura 7. Sexta prueba del sistema

Por último, se probó la funcionalidad de “left shift”, la cual mueve un valor determinado de 0 hacia la izquierda el número binario ingresado y rellena con ceros la cantidad de espacios alterados, por lo que al cambiar a modo 11, la salida serán 3 ceros del movimiento realizado más los dígitos más

significativos del número en la entrada a, este comportamiento es posible apreciarlo en las figuras 8 y 9.

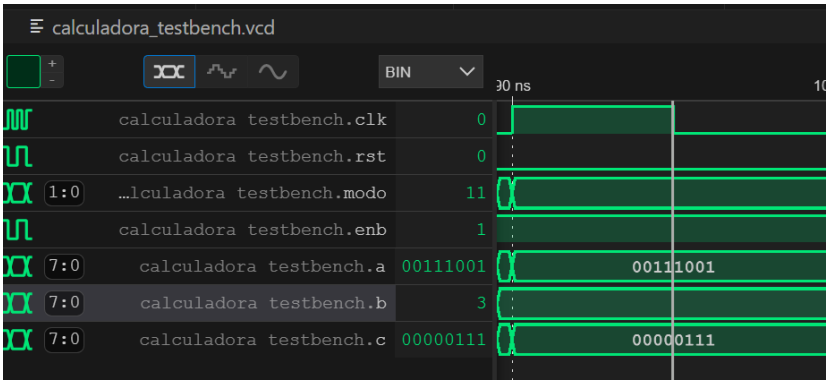

Figura 8. Séptima prueba del sistema

Figura 9. Octava prueba del sistema

Finalmente, se procedió a verificar si el código sintetizado obtenido al realizar los comandos mostrados en la bandera synth del makefile, utilizando la librería proporcionada por la profesora “cmos_cells”, genera el mismo waveform que el diseñado por el estudiante, al utilizar el mismo testbench, agregar el modulo que contiene el comportamiento de las compuertas y el archivo sintetizado al mismo el cual está compuesto por las compuertas mostradas en la figura 10. Se procedió a correr la simulación dando el resultado de la figura 12; en dicha figura, se puede apreciar que, durante el primer ciclo de reloj, el valor de c, se vuelve no importa, se cree que Yosys, considero que esto no afectaría el funcionamiento de la calculadora ya que no presenta ningún cambio después de este con respecto al waveform original de la figura 11.

```
12.1.2. Re-integrating ABC results.
ABC RESULTS:               NOT cells:       60
ABC RESULTS:               NOR cells:      237
ABC RESULTS:              NAND cells:      267
ABC RESULTS:          internal signals:     315
ABC RESULTS:             input signals:      28
ABC RESULTS:            output signals:       8
Removing temp directory.
Removed 0 unused cells and 131 unused wires.
```

Figura 10. Compuertas generadas

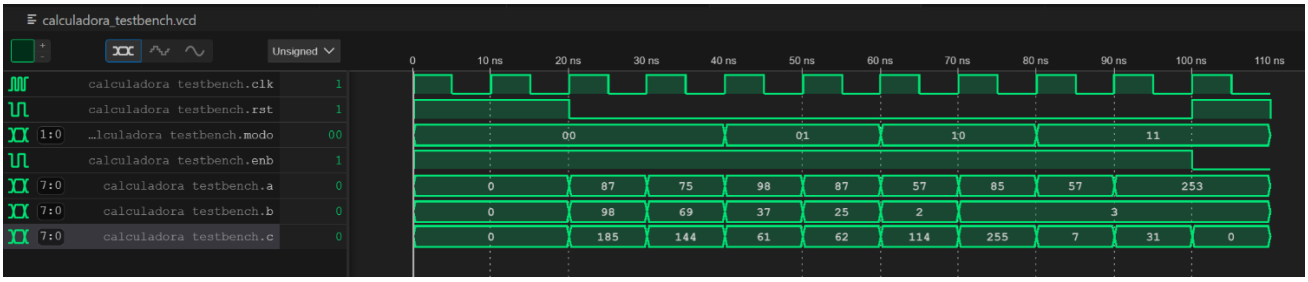

Figura 11. Waveform original

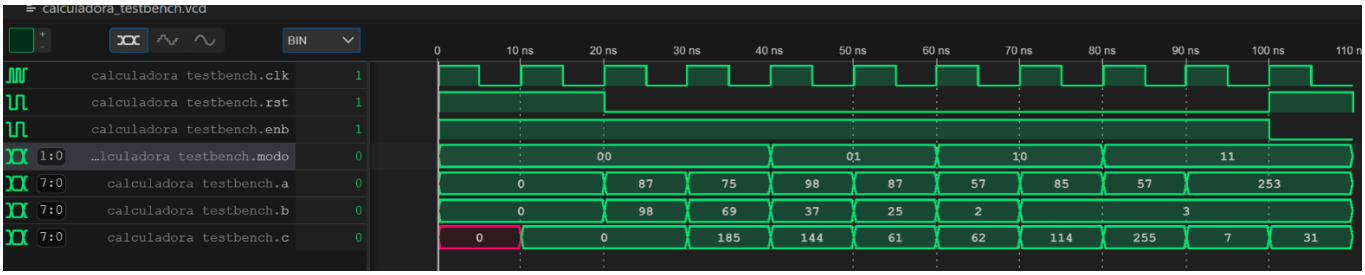

Figura 12. Waveform del archivo sintetizado

Conclusiones

Los resultados obtenidos de las pruebas realizadas sobre la calculadora revelan un desempeño consistente en todas las operaciones evaluadas. Después de la revisión de cada momento de la simulación, desde la inicialización de las señales hasta la comparación entre el diseño original y el sintetizado, se pueden extraer conclusiones significativas.

Inicialmente, la fase de inicialización mostró que las señales se configuran adecuadamente al aplicar la señal de reset, lo que garantiza un estado predecible y estable al inicio de la simulación. Durante las pruebas de suma, resta, multiplicación y “left shift” se evidenció un comportamiento coherente y preciso de la calculadora, confirmando la correcta implementación de la lógica de cada operación. La variedad de pruebas realizadas, incluyendo casos límite, permitió verificar su correcto funcionamiento en diversas situaciones.

Además, la comparación entre el diseño original y el sintetizado, utilizando la librería proporcionada "cmos_cells", reveló resultados consistentes. Aunque se detectó un cambio en el valor de una de las señales durante el primer ciclo de reloj en la simulación sintetizada, este fenómeno no impactó negativamente en el funcionamiento global de la calculadora. En consecuencia, se puede concluir que la síntesis del diseño no compromete la precisión ni la integridad funcional del sistema.

Aspectos a mejorar

Pese a ser un laboratorio relativamente sencillo, debo de leer con más atención las indicaciones ya que en un inicio confundí que el left shift era con la entrada b.