

Bitácora de Proyecto Grupal 1 - Máquinas de Estados y Comunicación UART

1st Javier Tenorio Cervantes
carnet: 2020065308

2nd Randall Bolaños Lopez
carnet: 2019043784

3th Kevin Lobo Juárez
carnet : 2020087823

Repositorio de Github utilizado en el Proyecto

En la siguiente bitácora se presenta los avances progresivos del proyecto, haciendo referencia a cada commit del repositorio de GitHub donde se evidencia un desarrollo continuo a lo largo del tiempo designado.

Enlace del repositorio: <https://github.com/Randall-BL/RBolanos-compu-archi-found-2G1-2024>

26/09/2024

Se comienza con la construcción de las tablas de verdad para el decodificador 1, el cual realiza la decodificación de 4 a 2 bits, esto para los valores binarios: 1000,1100,1110,1111. Estos deben tener un valor en 2 bits equivalente a 00,01,10,11 respectivamente. Para esto se construyó la siguiente tabla:

A	B	C	D	Y ₀	Y ₁	Decimal
1	0	0	0	0	0	1
1	1	0	0	0	1	2
1	1	1	0	1	0	3
1	1	1	1	1	1	4

TABLE I

TABLA DE VERDAD PARA EL DECODIFICADOR. DECODIFICADOR DE 4 A 2 BITS.

Mediante esta tabla de verdad se obtuvieron las salidas Y₀ y Y₁ respectivamente, esto haciendo uso de suma de productos y simplificación de álgebra booleana.

$$\begin{aligned} Y_0 &= ABC\bar{D} + ABCD = ABC(\bar{D} + D) = ABC(1) = ABC \\ Y_1 &= AB\bar{C}\bar{D} + ABCD = AB(\bar{C}\bar{D} + CD) \end{aligned}$$

27/09/2024

Se inició la construcción de un módulo decodificador utilizando QuartusPrime y System Verilog, que cumpliera con los requerimientos establecidos anteriormente, para esto se hizo uso de compuertas lógicas guiándose con las salidas Y₀ y Y₁ encontradas. Se terminó de implementar dicho decodificador de manera exitosa, se obtuvo el siguiente diagrama que contiene todas las compuertas utilizadas para su construcción.

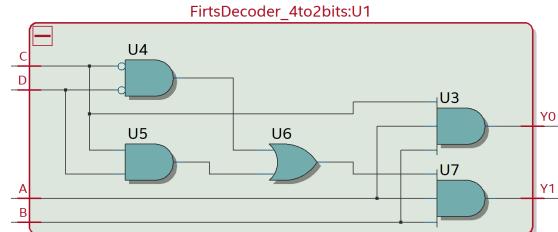


Fig. 1. Decodificador de 4 a 2 bits en SystemVerilog

# Tiempo	A	B	C	D		Y ₀	Y ₁
# -----							
# 10	1	0	0	0		0	0
# 20	1	1	0	0		0	1
# 30	1	1	1	0		1	0
# 40	1	1	1	1		1	1

Fig. 2. Resultados del testbench para el decodificador

5/10/2024

Se comenzó a implementar la construcción de la ALU, se construye un sumador de un bit complejo, el sumador completo de 2 bits, la resta, la operación AND y la operación OR totalmente funcionales.

08/10/2024

Empezamos la investigación y construcción de la comunicación entre el Arduino y la FPGA, en un inicio iniciamos probando realizarla mediante SPI, sin embargo se dificultó el proceso por la sincronización. Decidimos cambiar la forma de la comunicación por UART. Iniciamos la implementación del módulo RX en System Verilog y en el Arduino, se logró que el código del arduino funcione bien, el de la FPGA tiene algunos fallos de sincronización.

10/10/2024

Se continuó trabajando en la comunicación UART, además se compró un convertidor lógico bidireccional de 3.3 voltios a 5 voltios. Esto para evitar daños en la FPGA, se realizó la soldadura correspondiente. Implementación del módulo RX y

TX en SystemVerilog, se realizaron las primeras pruebas de comunicación entre el arduino y la FPGA.

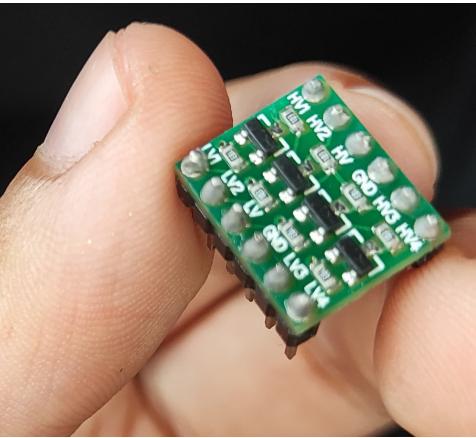


Fig. 3. Resultados del testbench para el decodificador

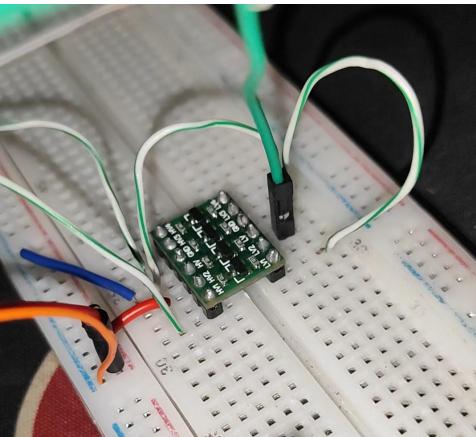


Fig. 4. Resultados del testbench para el decodificador

14/10/2024 - 15/10/24

Se termina realizar la comunicación UART correctamente, ademas implementamos el handshake para el cierre de comunicación y realizamos varias pruebas tanto en el envío como en el recibimiento de datos en la FPGA y el Arduino. Ademas se continua con la implementación de la ALU, se implementan las FLAGS de estado.



Fig. 5. Resultados del UART pruebas de envio de datos

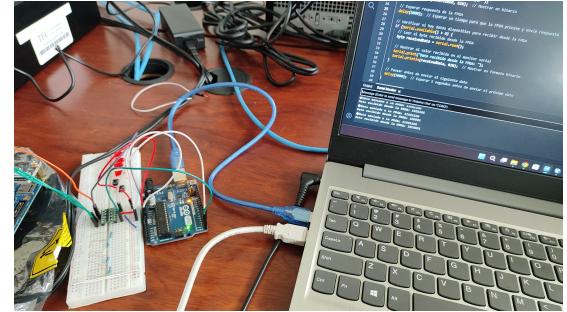


Fig. 6. Resultados del UART pruebas de recepcion de datos

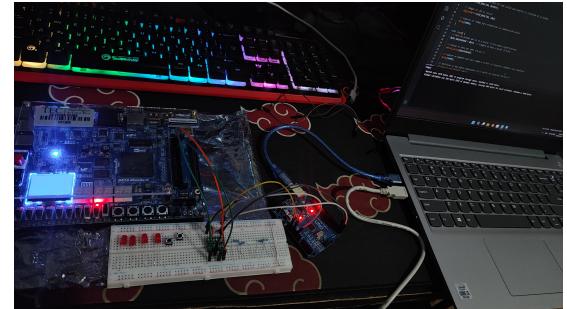


Fig. 7. Prueba del funcionamiento del handshake

18/10/2024

Se empezó la modificación de la comunicación UART, ya que estaba secuencial y se necesita realizar totalmente estructural, se plantean posibles soluciones, sin embargo no se logró el correcto funcionamiento de los módulos con las correcciones aplicadas. Se continua intentando modificar la comunicación exitosamente.

19/10/2024- 20/10/24

Se termina la construcción de la ALU con todas las operaciones y flags de estado, todo de forma estructural utilizando compuertas lógicas. Se implementa el modulo PWM que controla la velocidad de motor en CD. Se termina la construcción del modulo y se procedió a construir un testbench para probar los módulos actuales y obtener resultados esperados. Se logro construir un testbench funcional y se obtienen resultados positivos en el mismo.

```
run -all
Deci 1000, Decoders Result: 00, ALU IN1: 01, ALU Operation: SUM, ALU result: 01, Motor speed: 10000000, Flags = Zi: 0, Ci: 0, Vi: 0, Si: 0
Deci 1100, Decoders Result: 01, ALU IN1: 11, ALU Operation: SR, ALU result: 11, Motor speed: 11111111, Flags = Zi: 0, Ci: 0, Vi: 0, Si: 1
Deci 1010, Decoders Result: 10, ALU IN1: 00, ALU Operation: AND, ALU result: 00, Motor speed: 10000000, Flags = Zi: 0, Ci: 0, Vi: 0, Si: 0
Deci 1111, Decoders Result: 11, ALU IN1: 10, ALU Operation: AND, ALU result: 10, Motor speed: 11000000, Flags = Zi: 0, Ci: 0, Vi: 0, Si: 1
Flag Test - C, ALU result: 00, Vi: 1
Flag Test - V, ALU result: 00, Vi: 1
Flag Test - S, ALU result: 00, Vi: 1
Flag Test - Z, ALU result: 00, Vi: 1
** Notes: finnish : i C:\Users\tencor\OneDrive\Documentos\Github\Proyecto2FAC\FAC_F1\th_top_module.evl(139)
1
Break in Module th_top_module at C:\Users\tencor\OneDrive\Documentos\Github\Proyecto2FAC\FAC_F1\th_top_module.evl line 139
```

Fig. 8. Testbench de la ALU y el PWM

20/10/2024

Se contruyó en protoboard el circuito de las entradas de los dedos para el decodificador, se utilizaron leds, fotoresistores, transistores 2N2222, resistencias. Se obtuvieron resultados positivos y se logró que funcionara perfecto. Ademas también

se dejó listo el desacople para el motor de CD necesario para evitar daños a la FPGA.

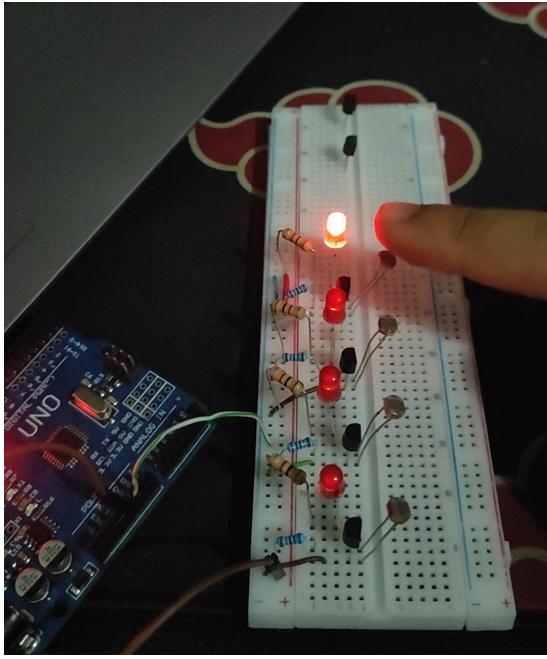


Fig. 9. Circuito con LEDs y fotoresistores Funcional

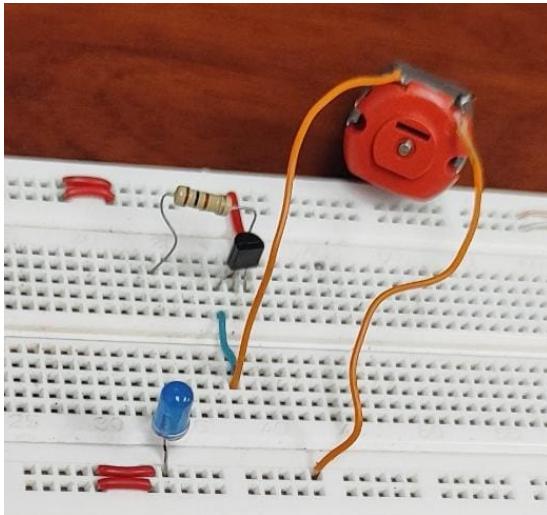


Fig. 10. Circuito con desacople para el motor de CD

21/10/2024

Se desarrolla el módulo del BCD a 7 segmentos de forma estructural, para esto se hizo uso de compuertas lógicas y se construyó cada uno de los segmentos por separado. También se implementó un registro que almacene el resultado de la ALU y que mediante un clock el dato pase tanto al 7 segmentos como al PWM. También se implementó un botón de reset, que reinicia todos los datos y el almacenado en registro lo pone en 0.

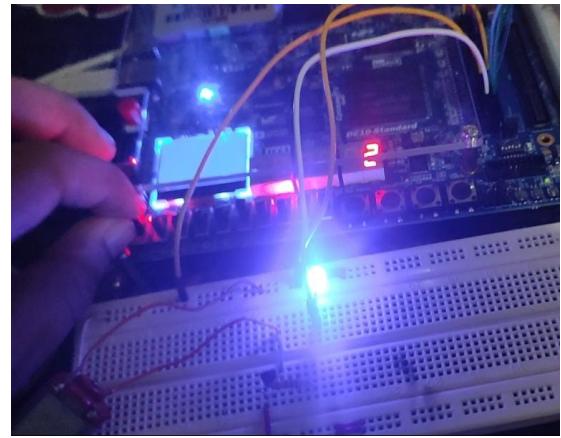


Fig. 11. Swtich de clock y 7 segmentos funcionando correctamente

22/10/2024

Se logró conectar toda la lógica de la comunicación y el recibimiento del segundo dato de la ALU por medio del arduino. Una vez listo esto se pincelaron a conveniencia todos los pinos GPIO que se iban a utilizar. Fue posible probar todo el proyecto en su totalidad, se obtuvieron resultados exitosos, el único inconveniente es que la comunicación UART se mantuvo secuencial debido a que no nos fue posible pasarla a estructural por múltiples bugs que afectaban a las demás partes del sistema, por lo que se optó por mejor utilizar la secuencial que funciona perfecto.

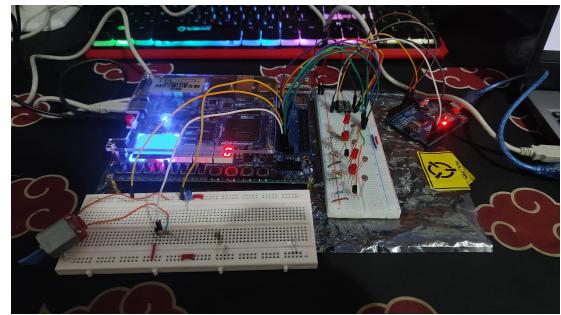


Fig. 12. Sistema del Proyecto totalmente implementado y funcional

23/10/2024

Corregimos algunos fallos que contenía el PWM, además se cambió un poco la lógica implementando un registro dentro de este módulo, para que de esta forma cumpliera con lo especificado en el proyecto. Hicimos una revisión general y detallada de todos los módulos, se procedió a realizar el último commit con el proyecto finalizado en su totalidad. Empezamos y terminamos la construcción de los diagramas necesarios del sistema.

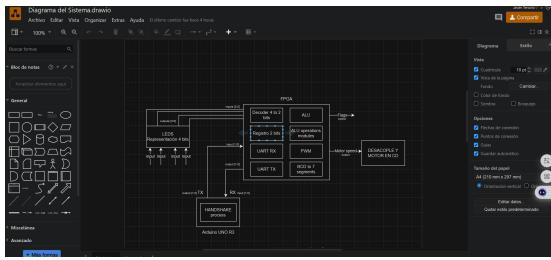


Fig. 13. Construcción del diagrama general del sistema

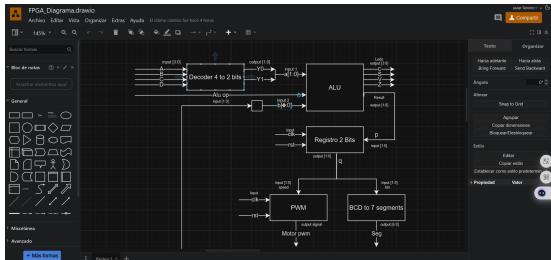


Fig. 14. Construcción del diagrama de módulos en la FPGA