Fecha de entrega 09 de mayo de 2023.

RGB LED Controler

VICENTE JASEN[1], ANDRES MENDOZA[1]
[1]Pontificia Universidad Católica de Chile (e-mail: vijasen@uc.cl, aemendoza@uc.cl)

SI Autorizo que mi proyecto (tal como ha sido entregado, sin nota ni comentarios de evaluación) sea publicado en un repositorio para pueda servir de guía y ser mejorado en proyectos de futuros estudiantes.

Este proyecto ha sido desarrollado bajo el curso IEE2463: Sistemas Electrónicos Programables.

ABSTRACT El proyecto RGB LED controller, se realizo para la tarjeta zybo z7 – 20, en vivado 2020.1, en lenguaje VHDL. Este proyecto permite controlar los colores del led rgb de la tarjeta de distintos modos, los cuales son 4; el primer modo permite cambiar los colores del led de manera manual, esto mediante el uso de los switches y botones, el segundo modo, es una secuencia de colores, hecha con una maquina de estados, a la cual se le puede ajustar la velocidad mediante los botones, el tercer y cuarto modo son cargar colores rgb predefinidos desde una memoria, de dos formas, una mediante protocolo AXI lite, y otro mediante AXI full, estos modos se seleccionan dependiendo de la combinación de los switches, y cada uno tiene distintas funcionalidades con los botones, los leds comunes indican el modo en el que se encuentra la tarjeta.

INDEX TERMS RGB, LED, AXI4, VHDL, Zybo Z7 -10, maquina de estados, AXI Lite, Vivado 2020.1, Block desing.

I. ARQUITECTURA DE HARDWARE (1 PUNTO)

La arquitectura de nuestro programa se divide en 3 partes:

- La primera es el control manual de color mediante el uso de botones y switches
- La segunda es la secuencia de luces a traves de la maquina de estados
- La tercera es la carga de colores predeterminados desde una memoria a traves de protocolo AXI.

Para el control manual se realizó un IP(Colorchange0) que recibe como entrada el botón 0 un vector dado por los 4 switches, 3 bits de colores, uno para cada color.Para la entrada se separa el vetor de 3 bits RGB, en 3 bits,utilizando el IP Slice de Vivado 3 veces , uno para cada entrada.Como salida tiene un vector de 3 bits que contiene los bits de colores rojo, verde, y azul respectivamente.Con esto cada bit de color es asociado a un color del LED RGB si el SuperMultiplexor esta funcionando en modo manual.

El IP Colorchange0 le cambia el valor de uno o varios bits de color si se cumple que se aprieta el botón 0 y se tiene el valor de switches asociado. Por ejemplo si se tienen todos los switches en 0 y se aprieta el botón 0, se cambia el bit del color rojo.

En la siguinete tabla se muestra los cambios que relizan cada configuración de switches al apretar el botón 0.

sw	bit de color que cambia
"0000"	Rojo
"0001"	Verde
"0010"	Azul
"0011"	Rojo y Verde
"0100"	Rojo y Azul
"0101"	Verde y Azul
"0110"	Rojo,Verde y Azul
otro	ninguno

Para la siguiente parte, la máquina de estados, esta se realizo creando 2 IP(ClockModv1.0 y SMachinev1.0), una que contiene la maquina y realiza los cambios de color segun los flancos de subida del clock, y otra IP que permite modificar la velocidad del clock, esto mediante 3 botones, el primer botón permite (botón 0) acelerar la velocidad del clock mediante una suma, es decir aumenta la velocidad del clock de manera ligera cada vez que se apreta el botón, el segundo botón aumenta la velocidad del clock con una multiplicación, esto aumenta la velocidad del clock de manera mas rápida que con el botón 0, tambien se puede disminuir la velocidad del clock, esto mediante el tercer botón(botón 2), este disminuye la velocidad por medio de una resta, los numeros por los cuales suma,resta y multiplica son configurables en el diagrama de bloques, los valores default son suma = 10, resta = 10 y multiplicador = 2, tambien en el último botón (botón 3), esta el reset, el cual al presionarlo, hace que vuelva el clock a la velocidad default (1 cambio por segundo), y estas modificaciones en el clock son las que permiten manejar la

velocidad de la maquina de estados, (nota este bloque solo afecta la velocidad del clock para los cambios en la maquina de estados, no afecta los demas bloques, esto queda mas en claro en el diagrama, figura 1), esta maquina se activa en la combinación de switches 1001.

Luego para la tercera parte, la carga de 6 colores predeterminados desde una memoria a traves de protoco AXI, esta se realizo basandose en las ayudantias 4 y 5, en este modo se utilizan los switches para seleccionar el color a cargar, los cuales estan especificados en la siguiente tabla.

Color	Combinación de switches
1	1010
2	1011
3	1100
4	1101
5	1110
6	1111

Luego de escribir la combinación en los switches se puede presionar el botón 2 (el tercer botón de la placa) el cual corresponde a un write in, para cargar el color desde la memoria al led, todo esto se realiza a traves de un bloque atg en test mode (AXI traffic generator, en el diagrama), el cual en sus archivos coe, tiene escrito el procedimiento y los colores preterminados a cargar, el cual esta conectado a un periferico creado por nosotros, el cual recibe la información del atg y la entrega dependiendo de la combinacion de los switches (RGB peripheral en el diagrama).

Tambien para este modo se tenia planeado realizar el mismo procedimiento pero con un bloque atg en modo advance, para lo cual se necesitaban 2 bloques atg (1 test mode y uno advance mode) y un periferico, pero este ultimo no se logró crear, por lo cual sus combinaciones de switches quedaron sin uso.

Color	Combinación de switches
sin uso (apagado)	0111
sin uso (apagado)	1000

Ademas mencionar otros 2 bloques que estan en el diagrama y cual es su funcionalidad, el bloque Mode indicator, enciende un led para indicar el modo en el que se encuentra la placa, esto segun la siguiente tabla

modo	Led encendido
manual	led 1
maquina de estados	led 2
AXI full	led 3
AXI lite	led 4

Finalmente el bloque Supermultiplexor permite conectar todos los bloques al led rgb, leyendo las misma combinaciones de los switches es cual modo deja pasar, es decir si se utiliza una combinación de switches perteneciente al modo AXI, deja pasa solamente al led rgb la salida entregada por el rgb peripheral, si se usa una combinación perteneciente al modo manual, solamente pasa la salida de Manual change, y el mismo procedimiento es aplicado para la maquina de estados.

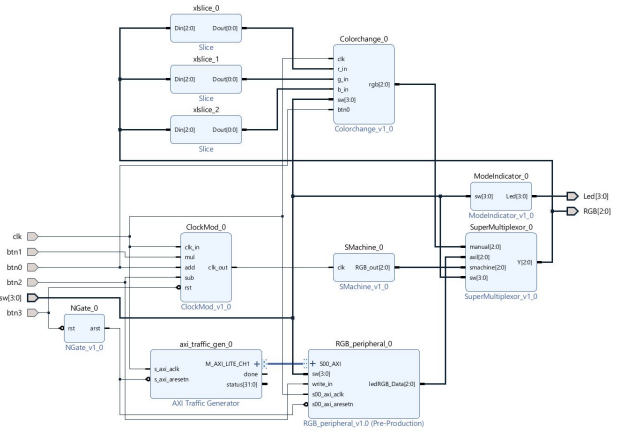

FIGURE 1: Diagrama de bloques del proyecto en vivado block desing

II. ACTIVIDADES REALIZADAS (1.5 PUNTOS)

AO1 (100%): *Descripción:* El código cumple con tener al menos 3 components colorchange,Slice,SMachine, SuperMultiplexor,etc.Estos se pueden ver dentro del wrapper del proyecto general. De acuerdo a la arquitectura presentada en la sección I, esto se implementó en los bloques colorchange,Slice,SMachine, SuperMultiplexor.Colorchange se utiliza para el control manual de colores. SLice se utiliza para manupular vectores y pasarlos a bits.Smachine se utiliza para crear la secuencia de colores.SuperMultiplexor se utiliza para seleccionar el estado de operación. *Nivel de Logro:* 100%. Completamente logrado, el código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitsteram y cargarlo en la ZYBOZ7.

AO2 (100%): *Descripción:* El código cumple con las funciones que habíamos planteado en nuestro proyecto las cuale son: La maquina de estados (cambiar los colores en orden segun el clock, esta hecha con 2 IP, Smachine y ClockMod) un periferico AXI Lite, (que entrega la información transmitida por AXI al led RGB), un bloque multiplexor (seleccionar la salida adecuada dependiendo de la combinación de switches), un indicador de modo (enciende los leds para indicar en que modo se encuentra operando la tarjeta) y finalmente el cambiardor de colores manual (el cual permite generar los colores con combinaciones de los switches y botones). De acuerdo a la arquitectura presentada en la sección I, los IP core creados por nosotros son ClockMode, ModeIndicator, Smachine, SuperMultiplexor, RGBperipheral y ManualChange, el bloque ClockMod, tiene parametros genericos los cuales pueden ser modificados *Nivel de Logro:* 100%. Completamente logrado, el código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitstream y cargarlo en la ZYBOZ7.

AO3:(75%) *Descripción:* El código cumple con las funciones 1 de las funciones que habíamos planteado en nuestro proyecto las cual es: usando un ATG en test mode poder cargar colores predefinidos al led RGB, la otra función que no se logró implementar fue cargar colores predetermindos usando

un bloque ATG en advance mode (solo falto la creación del periferico AXI full, los archivos coe y las configuraciones del ATG estan listas) . De acuerdo a la arquitectura presentada en la sección I, esto se implementó en los bloques AXI traffic generator y RGBperipheral, cuya funcionalidad fue cargar los colores predefinidos al led rgb. *Nivel de Logro:* 75%. la parte del ATG en test mode funciona al 100% (se genero el bitstream y se cargo a la tarjeta), y para la parte del ATG en advance mode solo falto terminar de crear el periferico que se comunicaria con este

AC1(100%): *Descripción:* El código cumple con las funciones que habíamos planteado en nuestro proyecto las cuale son: crear una maquina de estados la cual tenga un patron de luces, al cual se le puede modificar la velocidad con los botones. De acuerdo a la arquitectura presentada en la sección I, esto se implementó en los bloques ClockMod y SMachine, cuya funcionalidad fue para ClockMod, es cambiar la velocidad del clock que recibe la maquina de estados (este bloque tiene parametros configurables) y SMachine es la maquina de estados que tiene el patron de colores. *Nivel de Logro:* 100%. Completamente logrado, el código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitstream y cargarlo en la ZYBOZ7

AC2(100%): *Descripción:* El código cumple con las funciones que habíamos planteado en nuestro proyecto las cuale son: los botones, son para cargar colores predeterminados, cambiar la velocidad de la maquina de estados, y cambiar de manera manual los colores del led RGB, los 4 leds, se utilizan para indicar el modo en el que se encuentra trabajando la tarjeta (manual, Maquina de estados, AXI full y AXI lite). De acuerdo a la arquitectura presentada en la sección I, esto se implementó en todos los bloques , ya que en total algunos bloques leen los switches (casi todos), otros se conectan a los leds (Mode indicator), y otros al led RGB (Super multiplexor, aunque este solo deja escoger cual es la salida que va aplicar, dado que los bloques conectados al multiplexor entregan la salida al led rgb, el multiplexor solo escoge cual pasa). *Nivel de Logro:* 100%. Completamente logrado, el código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitstream y cargarlo en la ZYBOZ7, y se ocupan los 4 botones, 4 switches, 4 leds y el led RGB.

AC3(60%): *Descripción:* El código cumple con algunas de las funciones que habíamos planteado en nuestro proyecto las cuale son: usar operadores y atributos en algunos bloques De acuerdo a la arquitectura presentada en la sección I, esto se implementó en los bloques ClockMod, el cual ocupa los operadores +, - y *, el bloque color change ocupa el operador not, ademas la mayoria de bloques ocupan operadores de asignación como lo son el <= y el :=, tambien ocupan operadores de comparacion como el =,>,<,>=, , el atributo rising edge se ocupa en todos los bloques que reciben un clock, lo cual da en total 5 operadores y 1 atributo de un total de 10

AC4(100%): *Descripción:* El código cumple con las funciones que habíamos planteado en nuestro proyecto las cuale son: actualizar de manera inmediata el numero del contador dentro del bloque Clock Mod (este numero al variar puede disminuir o aumentar la velocidad del clock de la maquina de estados, por lo cual acelera la velocidad del patron de luces). De acuerdo a la arquitectura presentada en la sección I, esto se implementó en los bloques ClockMod, cuya funcionalidad fue en el caso de las señales poder enviar el numero de un proceso a otro y conectar la salida, y para la variable, la cual corresponde al numero que se le suma al contador, esta se utilizo para poder actualizarlo de manera inmediata *Nivel de Logro:* 100%. Completamente logrado, el código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitstream y cargarlo en la ZYBOZ7

AC5(100%): *Descripción:* El código cumple con las funciones que habíamos planteado en nuestro proyecto las cuale son: crear la maquina de estados, ya que para crearla nos basamos en la plantilla entegada en la capsula de video de maquinas de estados, la cual contiene una parte codigo secuencial y codigo combinacional. De acuerdo a la arquitectura presentada en la sección I, esto se implementó en los bloques SMachine, cuya funcionalidad fue la del codigo secuencial es actualizar el estado cada vez quer hay un flanco de subida del clock, y el codigo combinacional mediante el uso de case y when contiene las definiciones de que hacer en cada estado. *Nivel de Logro:* 100%. Completamente logrado, el código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitstream y cargarlo en la ZYBOZ7.

AC7(100%): *Descripción:* El código cumple con utilizar una parte de Hardware no vista en clases que es utilizar el led 6 RGB de la tarjeta Zybo z7-10, ademas de utilizar el IP Slice de Vivado, para cortar vectores. De acuerdo a la arquitectura presentada en la sección I, esto se implementó en los bloques Slice en la entrada del bloque ColorChange, cuya funcionalidad fue la de manipular vectores cuando se nesecitaba trbajar con bits.Ademas el Led RGB fue utilizado en la salida RGB que es asociada al led 6 RGB de la tarjeta. *Nivel de Logro:* 100%. Completamente logrado, el código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitstream y cargarlo en la ZYBOZ7.

III. RESULTADOS DE SIMULACIÓN (3 PUNTOS)

Para los resultados de simulación, estos se probaron en el proceso de enviar el color predefinido desde el periferico AXI (RGBperipheral) al led RGB, La simulación post implementación de comportamiento se ve de la siguiente manera

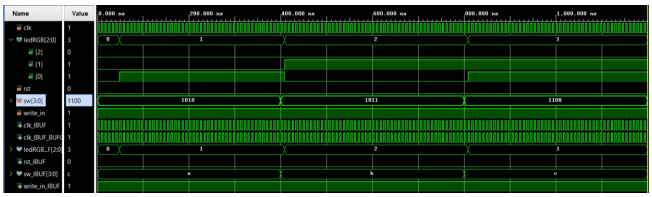

FIGURE 2: Scope de simulacion funcional post implementacion.

Esta simulación, se realizaron 3 combinaciones con los switches (1010, 1011 y 1100) para colocar 3 colores predeterminados en el led rgb (001, 010 y 110), en esta simulación como es de tipo funcional, solo es para ver si el codigo cumple con la función con la que fue escrito, en este caso entregar colores al led rgb, y se ve que realiza todo bien, al principio se nota que el color no carga de manera inmediata, y esto se debe a que durante ese tiempo el atg en test mode se encuentra escribiendo los colores en la memoria del periferico, los cuales una vez escritos pueden ser leidos y entregados al led.

Luego para la simulación post implementación para el timming (que corresponde a la simulacion real), se utilizaron las mismas combinaciones de switches y botones, la misma velocidad de clock y los mismos tiempos de activación y duración de los switches y botones, la simulación se ve de la siguiente manera

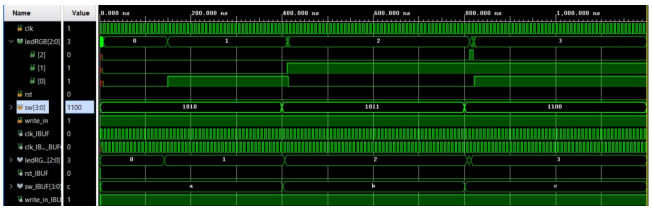

FIGURE 3: Scope de simulacion timming post implementacion.

Esta simulación a pesar de que se realizo de la misma manera que la anterior hay zonas a destacar donde se puede observar la influencia del delay, las cuales son las siguientes

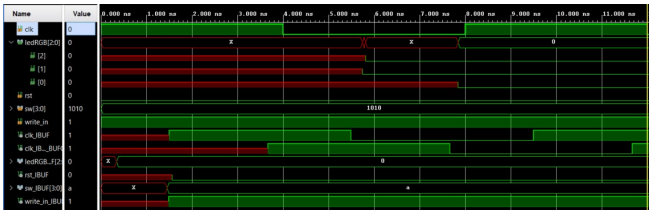

FIGURE 4: Zona delay 1.

En esta zona el delay produce que durante unos instantes los valores de las variables sean desconocidos, y que los buffers no obtienen la información de inmediato

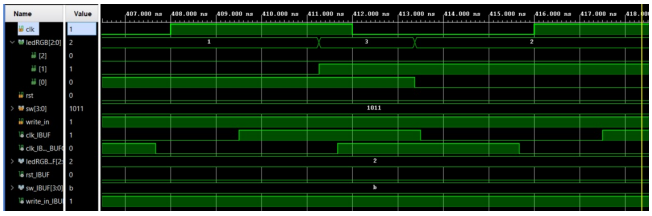

FIGURE 5: Zona delay 2.

La segunda zona donde afecta el delay es en el cambio de los colores, ya que las señales no se actualizan de manera "instantanea", por lo cual se produce un pequeño traslape entre los valores, lo cual genera durante unos instantes (demasiado pequeño para que sea notorio) el color equivocado, esto se produce en todos los cambios de color.

Tambien mencionar que la diferencia entre las señales y las variables, esta en que las señales se actualizan solamente 1 vez por cada ejecución del proceso , no como las variables, las cuales se actualizan de manera inmediata dentro del proceso (pero las variables solo son locales de ese proceso, las señales son generales de la arquitectura, es decir las variables solo se pueden ocupar dentro de ese proceso y en ningun otro, pero las señales se pueden ocupar en cualquier proceso)

Luego para ver los HandShakes del protocolo AXI Lite se realizo una simulación de funcionamiento post implementación, la cual arrojo el siguiente scope

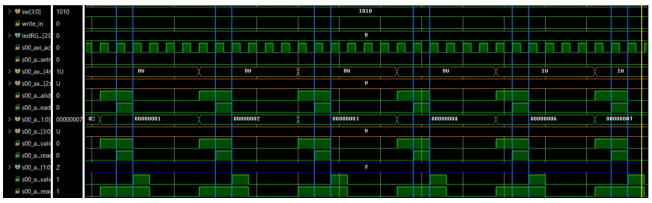

FIGURE 6: Scope protocolo AXI lite

En este scope los handshakes son los que se encuentran entre los marcadores azules, estas zonas corresponde al momento donde los canales Valid y Ready (ya sea de escritura o lectura, en este caso solo se realizaron directamente operaciones de escritura) se encuentran activos

Para observar mejor, se tiene la siguiente imagen donde se muestra con mas zoom un handshake

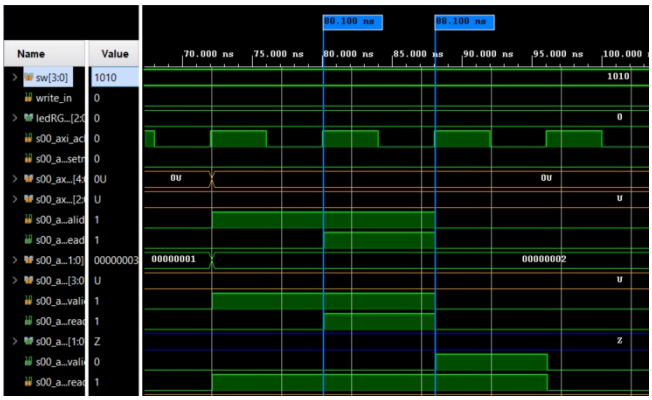

FIGURE 7: Scope de un handshake

IV. RESULTADOS IMPLEMENTACIÓN (0.1 PUNTOS)

Durante la implementación, los principales problemas que tuvimos fueron, el bloque clockmod funcionaba pero no cambiaba la velocidad de la maquina, el bloque colorchange no arrojaba ningun error pero no cambiaba los colores del led, de estos errores, el bloque colorchange fue arreglado por completo y quedo con su funcionalidad al 100%, pero el bloque clockmod solo se pudo arreglar de manera parcial, esto mediante el uso de una variable, aun asi hay botones que no tienen el funcionamiento correcto, otro error que es importante mencionar es que en el modo manual, los botones quedaron sensibles, entonces se tienen que presionar por poco tiempo para su correcto funcionamiento (que no apage y prenda el canal al tocar 1 vez)

Tambien mencionar que el problema mas grande fue el no poder crear el periferico que se comunicara por AXI full al bloque atg en advance mode, esto ya que al usar el IP packager de vivado para crear un periferico AXI full, no entendimos como leer los datos que este recibia y por lo tanto no se pudo terminar su creación, mencionar tambien que la configuración para este procedimiento estaba pensada en 1 bloque atg en test mode conectado a un bloque atg en advance mode (conectados mediante el bloque AXI protocol converter), ambos bloques y los archivos coe si lograron crearse y configurarse correctamente.

V. CONCLUSIONES(0.2)

- Se logró cumplir con el objetivo de crear un controlador RGB que presenta 3 funcionalidades,un ajuste manual de color, la carga de un color desde una memoria y crear una secuencia de colores en el tiempo atraves de una maquina de estado.
- Se logró crear al menos 3 components y generar al menos 3 packages en Vivado,se logró comunicar un ATG maestro con un IP esclavo utilizando Test Mode.
- No se logró comunicar un ATG Maestro con un IP esclavo utilizando Advance Mode, se logró configurar el periférico AXI Full y los archivos coe, pero no se logró comunnicar el periférico AXI Full con el IP esclavo.
- Se logró generar una máquina de estado hacerla funcionar en la implementación.
- Se logró utilizar los 4 switches, 4 botones , y 4 Leds de la tarjeta.
- Se utilizó código secuencial y concurrente..

VI. TRABAJOS FUTUROS (0.2)

Este proyecto en particular esta pensado para el led RGB de la tarjeta Zybo Z7-10, sin embargo se puede hacer una versión más compleja del controlador si se asume un Led RGB que reciba 3 vectores de 8 bit, uno bit para el color rojo, uno para el vere y uno para el azul.Así se podría obtener un color más complejo.

Tambien se podría modificar para que la máquina de estado se pueda configurar para poder elegir la secuencia de colores de forma manual, en vez de una secuencia determinada, o tambien se podría exportar una secuencia de una memoria utilizando el protocolo AXI, de forma que se exporte una secuencia de colores previamente guardada en una memoria.

Tambien se podria hacer una configuración que mediante AXI permita guardar en una memoria el color mostrado actualmente o la secuencia de colores configurada actualmente, de forma que se pueda dejar guardado una secuencia deseada o un color deseado.

Ademas se puede lograr completar la implementación del modo AXI full en la tarjeta para lograr cargar colores predefinidos al led, todos los espacios para agregarlo al proyecto estan.

REFERENCES

[1] Sebastian Castro, “Ayudantıa 5: AXI Interconnect” IEE2463, Primer semestre 2023.
[2] Reimundo Alcalde, “Ayudantıa 4: Comunicacion AXI” IEE2463, Primer semestre 2023.
[3] Sebastian Castro, “Ayudantıa 3” IEE2463, Primer semestre 2023.
[4] Felix Rojas, PecLab “State Machines,Modeling Sequential Logic Circuits” IEE2463, Primer semestre 2023.
[5] Felix Rojas, PecLab “PACKAGES and COMPONENTS, SYSTEM DESIGN” IEE2463, Primer semestre 2023.
[6] Felix Rojas, PecLab “Signals and Variables, When and where to use them” IEE2463, Primer semestre 2023.
[7] Felix Rojas, PecLab “LAB04, AXI Traffic Generation (ATG)” IEE2463, Primer semestre 2023.
[8] Felix Rojas, PecLab “LAB04, AXI Traffic Generation (ATG)” IEE2463, Primer semestre 2023.