Fecha de entrega 11 de Diciembre de 2023.

Juego de Gato en tarjeta Booster MK II

JOAQUÍN DUNNER[1], CLAUDIO ROJAS [2]
[1]Pontificia Universidad Católica de Chile (e-mail: primerintegrante@uc., segundointegrante@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 consiste en la creación del popular juego 'Gato' haciendo uso de la tarjeta ZYBOZ7 y la 'Booster Pack MKII', en el que al presionar el botón del joystick de esta ultima se imprime una 'X' o un 'O' en la pantalla y el primer jugador en hacer una linea horizontal, vertical o diagonal gana el juego. El diseño de este proyecto tiene una parte de software y otra de hardware. Para la parte de software se ha implementado una lógica principal en Vitis se hace lectura y escritura sobre los distintos periféricos para que se impriman en la pantalla LCD los elementos que conforman al juego. Se hace uso del botón del joystick para iniciar una jugada de cada jugador, se implementan interrupciones en el acelerómetro, micrófono y sensor de luz, se usan los botones de propósito general para reiniciar el juego y se hace sonar el 'Buzzer' durante cada turno. Respecto al Hardware, se implementó en Vivado un block design que considera al 'ZYNQ' como modulo central y se utilizan módulos GPIO, AXI Timer, AXI IIC y AXI SPI para realizar la comunicación con los distintos periféricos y poder ser controlados de forma correcta. Se logra una correcta implementación proyectando en la pantalla LCD un juego que funciona de manera correcta y se generan interrupciones bajo las condiciones planteadas correctamente..

INDEX TERMS Gato, Vivado, Vitis, FPGA, ZYBOZ7-10, Booster Pack MKII, IP-Cores, Máquina de Estados, Comunicación AXI, Buzzer, Pantalla LCD, Comunicación SPI, Comunicación I2C.

I. ARQUITECTURA DE HARDWARE Y SOFTWARE

EL proyecto realizado consiste en el popular juego de 2 jugadores conocido como 'Gato' en el que el objetivo es hacer una linea horizontal, vertical o diagonal , para ello se usan los potenciómetros de la tarjeta Booster para determinar la coordenada horizontal o vertical donde se colocará un símbolo. Se está haciendo uso del botón del 'joystick' para colocar el símbolo y se está haciendo uso de la pantalla para mostrar el progreso de este. Además se esta haciendo uso.

Al iniciar el juego hay una interacción a través de la terminal serial para elegir la frecuencia del sonido de cada jugador la cual se escucha a través del Buzzer. Para ello se ha diseñado un IPCore que realiza la comunicación entre el PL y el Buzzer para que este ultimo emita un sonido con una frecuencia igual a le enviada en la consola.

Durante el desarrollo del juego, en la pantalla se muestra la ubicación en el eje X, la ubicación del eje Y y el turno del jugador. Al presionar el joystick se imprimirá en la pantalla un carácter 'X' o 'O' en la ubicación determinada dependiendo si es el jugador 1 o el jugador 2. A lo largo del desarrollo del juego se ha hecho uso de interrupciones usando 3 periféricos los cuales son el micrófono, el acelerómetro y el sensor de luz. Se muestra un mensaje en pantalla para mostrar que se está dentro de la interrupción.

Al finalizar el juego o llegar a un empate se muestra en pantalla el mensaje de finalización existe un mensaje diferente en el caso de que sea algún jugador haya ganado o en el que se de un empate. Al usar el botón 0 de propósito general (el superior de la Tarjeta Booster) se reinicia el juego.

En la Figura 1 se puede ver un diagrama de bloques con el funcionamiento del software descrito con las principales funciones del software. En negro se ve la lógica principal del juego, en azul se ve la lógica de las interrupciones y en rojo se ve la lógica de cambio de turno.

Para el desarrollo de este proyecto se implementó un hardware en Vivado y software en Vitis. Para Vivado se hizo uso del 'Zynq' , dicho IP-Core esta conectado a un 'Axi Interconnect' y este ultimo esta conectado a los distintos periféricos. Para ello se hace uso de 4 módulos 'AXI_GPIO' (0-3), un módulo 'AXI_IIC', 2 módulos 'AXI_Timer', 2 módulos 'AXI_Quad_SPI' y un módulo de creación propia

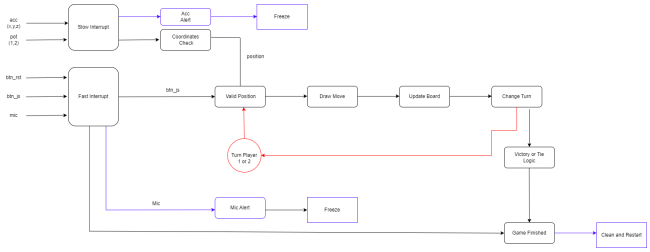

FIGURE 1: Este es un ejemplo de como incorporar una figura de un esquema.

'AXI_Buzzer'.

Los 'AXI_GPIO' son utilizados en orden para la lectura de señales de control de la pantalla, la lectura del botón del joystick, la lectura del botón 0 y la lectura del pin de interrupción del sensor de luz respectivamente. Los 2 módulos 'AXI_Timer' son hechos para generar interrupciones: AXI_Timer 0 es para la interrupción rápida (20ms) y AXI_Timer 1 para la interrupción rápida (100ms). Los 2 módulos 'AXI_Quad_SPI' se encargan de la comunicación con la pantalla LCD y el módulo 'AXI_IIC' de la comunicación con el sensor de luz. El módulo 'AXI_Buzzer' se encarga de hacer la comunicación con el Buzzer este recibe de entradas frecuencias especificados por el usuario desde la terminal de Vitis y hace sonar el Buzzer a esa frecuencia.

En la Figura 2 se puede ver un diagrama de bloques con la arquitectura de Hardware y los principales IP-Cores de esta. En Azul se puede observar la comunicación AXI entre módulos.

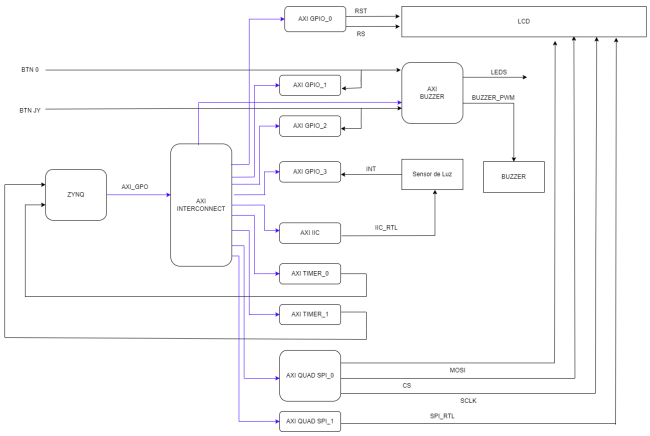

FIGURE 2: Este es un ejemplo de como incorporar una figura de un esquema.

II. ACTIVIDADES REALIZADAS

Describa el nivel de avance que obtuvo en cada una de las 10 actividades planteadas. Sea breve y guíese por el siguiente formato:

AO1 (100%): *Descripción:* Respecto a la pantalla, todo el juego se despliega en el LCD: se muestra el tablero del gato, se actualizan los turnos y se imprimen los mensajes especiales dependiendo de si se gana, empata o se muestran mensajes relaciones a interrupciones. Además se hace uso de otros 3 periféricos como lo son el joystick, potenciómetros y acelerómetro.
Nivel de Logro: 100%. Completamente logrado, el código implementado en Vitis compila correctamente y se observa de manera esperada el despliegue en la pantalla .

AO2 (100%): *Descripción:* Se generaron 2 timers uno rápido (20ms aproximadamente) y uno lento (100ms) aproximadamente. La interrupción rápida fue usada para la interrupción del micrófono y los botones mientras que la interrupción lente fue usada para la interrupción del botón del joystick, potenciómetros y acelerómetros. Con estos timers se leen periódicamente los periféricos utilizados. Por otro lado se implementó una interrupción del sensor de luz en la que se configuró en los registros del sensor un límite inferior de 1000 y se implementó un sistema de reinicio del sensor. Tanto las interrupciones por timers como la del sensor de luz son desplegadas al usuario a través de un mensaje en pantalla.

Nivel de Logro: 100%. Completamente logrado, se implementó en el block design, 2 IPCORES del tipo Axi Timer los cuales se conectaban al Zynq. El block design generó un bitstream adecuado y se pudo observar como en Vitis se ejecutó un código que cumplía con lo esperado y se desplegaba el mensaje correcto. Además fue posible observar el correcto funcionamiento de la interrupción del sensor de luz en los momentos adecuados y el despliegue en pantalla correcto.

AO3 (100%): *Descripción:* Se diseñó el IP-Core conocido como AXI_Buzzer. Este módulo recibe como entrada AXI 2 frecuencias las cuales son configuradas por el usuario, cada una de estas frecuencias representa el sonido que se escuchará cuando el determinado jugador presione el botón del Joystick. Luego este módulo interactúa con el Buzzer para hacerlo sonar con las frecuencias enviadas. En Vitis se implementan funciones con punteros a lo largo de todo el 'main()' tanto en la inicialización como en la configuración del hardware. En particular la función encargada de realizar esta comunicación se llama 'comm_with_PL'

Nivel de Logro: 100%. Completamente logrado, el IP-Core tiene un funcionamiento adecuado y no genera complicaciones al correr el 'bitstream'. Al ejecutar el programa se le pide al usuario escribir la frecuencia que desea para cada jugador y estos suenan adecuadamente comprobando así tanto el funcionamiento del IP-Core como de las funciones implementadas en Vitis.

AC1 (100%): . *Descripción:* El IP-Core ya mencionado fue escrito como una máquina de estados. Este controla el sonido que emite el parlante dependiendo del turno del

jugador.

Nivel de Logro: 100%. Completamente logrado, al correr el bitstream y ejecutar el programa en Vitis se observa un correcto funcionamiento de este módulo.

AC2 (100%): . *Descripción:* Se implementaron estructuras, arreglos, punteros, funciones y macros a lo largo de todo el código. Dando algunos ejemplos: se implementó una estructura llamada 'list_9' para saber que posiciones del tablero ya estaban ocupadas; se implementaron funciones para las interrupciones como 'timerInterruptHandler'; se implementaron punteros tal cual como fue descrito en la actividad obligatoria 3; se implementaron macros al inicio del código como los 'XGpio' para referirnos a los GPIO de manera global.

Nivel de Logro: 100%. Completamente logrado, cada una de los aspectos mencionados compila correctamente en Vitis y se obtienen resultados esperados al correr el programa.

AC3: (100%): *Descripción:* Además de los periféricos descritos en la actividad obligatoria 1, se usó el sensor de luz, los botones de propósito general, los LEDs y botones de la ZYBO, el Buzzer y el micrófono.

Nivel de Logro: 100%. Completamente logrado, cada uno de los periféricos funciona correctamente al correr el programa.

AC4: (100%): *Descripción:* Se hizo el 'booteo' de la ZYBO tal cual como se mostró en la ayudantía correspondiente. Al encender la Zybo el programa carga automáticamente al seleccionar el modo adecuado. Es importante mencionar que este programa no carga hasta que el usuario escribe en la terminal los valores de frecuencias, por lo que hay que asegurarse de que este abierta al momento de encender la tarjeta.

Nivel de Logro: 100%. Completamente logrado, el booteo fue implementado correctamente y el programa inicia al encender la tarjeta.

AC5: (100%): *Descripción:* Se hizo uso de VIO e ILA en el block design del hardware del proyecto. El VIO se usó para verificar que las interrupciones de los timers estuvieran funcionando correctamente. Por otro lado ILA se utilizó para medir la comunicación con el IP-Core desarrollado para insertar las frecuencias al inicio del juego, de manera que al configurar como trigger el bus '... RDATA' se pudiera observar que se enviara el dato correspondiente.

Nivel de Logro: 100%. Completamente logrado, se pudo observar las señales deseadas tanto por VIO para las interrupciones como con ILA para la comunicación AXI.

AC5: (100%): *Descripción:* Se hizo uso de VIO e ILA en el block design del hardware del proyecto. El VIO se usó para verificar que las interrupciones de los timers estuvieran funcionando correctamente. Por otro lado ILA se utilizó para medir la comunicación con el IP-Core desarrollado para insertar las frecuencias al inicio del juego, de manera que al configurar como trigger el bus '... RDATA' se pudiera observar que se enviara el dato correspondiente.

Nivel de Logro: 100%. Completamente logrado, se pudo observar las señales deseadas tanto por VIO para las interrupciones como con ILA para la comunicación AXI.

AC6: (100%): *Descripción:* Se utilizó el Buzzer para especificar la frecuencia del sonido que se emitirá cuando cada jugador presiona el joystick y hace su jugada. Para ello se implementó el ya mencionado IP-Core y la función en Vits para hacer la comunicación. Al iniciar el juego se debe escribir la frecuencia que se le asignará a los movimientos del jugador 1 y el jugador 2. Por ejemplo: 500 y 1500Hz.

Nivel de Logro: 100%. Completamente logrado,al iniciar el juego en la terminal el usuario debe escribir las frecuencias y al presionar el joystick se escucha un sonido del Buzzer que se corresponde con la frecuencia seleccionada demostrando un correcto funcionamiento del IP-Core como de las funciones de Vitis.

AC7: (0%): *Descripción:* No se hizo uso de la tarjeta SD.

Nivel de Logro: 0%. No logrado, no fue posible hacer uso de la tarjeta SD..

III. RESULTADOS

Se puede comprobar el correcto funcionamiento del IP-Core diseñado 'AXI Buzzer' a través del ILA de Vivado. Con esto se observa que en el bus '... WDATA' se envían las 2 frecuencias seleccionadas por el usuario. En la Figura 3 se muestra lo explicado.

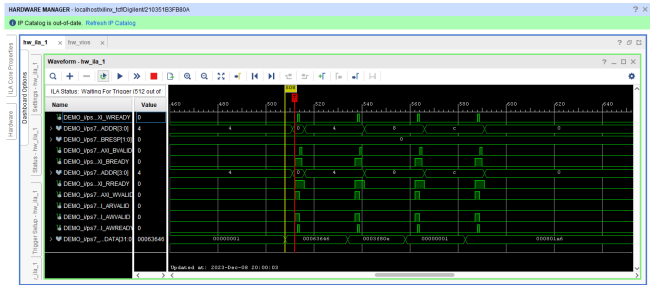

FIGURE 3: Este es un ejemplo de como incorporar una figura de un esquema.

Por otro lado se comprueba el correcto funcionamiento del código de Vitis a través del modo Debugging. Se comprueba como se entra en una interrupción al presionar el botón de

proposito general este cambia su valor y además se observa como el códgio se de tiene en el módulo de interrupciones. En la Figura 4 se muestra lo descrito.

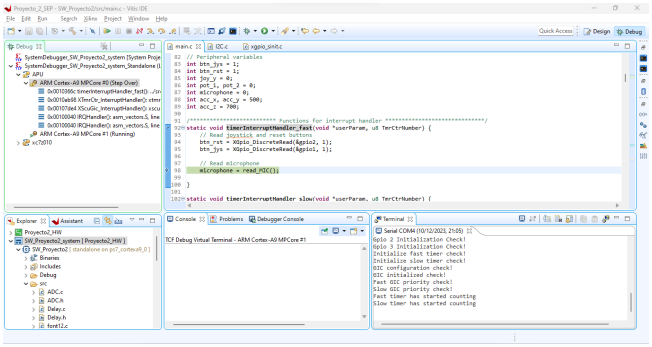

FIGURE 4: Este es un ejemplo de como incorporar una figura de un esquema.

Por último se muestra el debugeo simultaneo entre Vitis y Vivado para ello se ha entrado nuevamente en la misma interrupción descrita a través de Vitis y en Vivado se hace uso de VIO para observar el valor de las interrupciones que se conectan al ZYNQ, se observa que al momento de entrar en el módulo de interrupciones, el valor de las señales medidas a través de VIO cambian a 1. En la Figura 5 se muestra lo descrito.

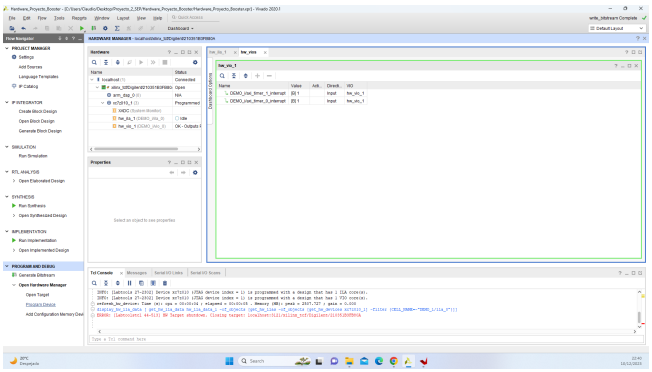

FIGURE 5: Este es un ejemplo de como incorporar una figura de un esquema.

IV. RESULTADOS IMPLEMENTACIÓN

Describa brevemente, en no más de dos párrafos, como le resultó su implementación. Los principales problemas que tuvo, cuales pudo resolver y cuales no.

La implementación de este proyecto considera a la pantalla, potenciómetros y botón del joystick y botón de propósito general para la lógica principal del juego y se consideran al acelerómetro, micrófono y sensor de luz para las interrupciones. A excepción del sensor de luz, todos los elementos mencionados fueron originalmente planeados para la implementación del programa y en todos se obtuvo un resultado esperado y exitoso al correr el programa.

La principal complicación que surgió fue la interrupción del sensor de luz. Originalmente se planeo utilizar el sensor de temperatura pero el 'datasheet' daba poca información al respecto. Respecto al sensor de luz, fue complejo entender el como configurarla para la interrupción. Este problema se solucionó leyendo el datasheet varias veces y después de una lectura pausada fue posible entender como configurarlo.

V. CONCLUSIONES

Entre las conclusiones más relevantes se encuentran.

- A través de ILA se esperaba obtener 2 resultados correspondientes a las 2 frecuencias enviadas. Se obtienen los datos 50 y 150 en un total de 4 ciclos de reloj.
- Se logra capturar una interrupción en una ciclo de loop después de presionar el botón. El programa se mantiene en la interrupción mientras se cumplan las condiciones requirdas.
- Se observa una actualización en un ciclo de loop a través del VIO de las variables asociadas a la interrupción.

VI. TRABAJOS FUTUROS

Como trabajo futuro se tiene:

- Implementar interrupciones del sensor de temperatura, ya que no fue posible descubrir como configurar el sensor para realizar la interrupción de forma simular al sensor de luz.

- Hacer uso de la tarjeta SD, ya que en el tiempo de trabajo no fue posible descubrir como implementar un código en Vitis que haga uso de ella. Queda pendiente la investigación de como implementarla y un código que cumpla esta función. Surge como idea escribir un txt en la tarjeta con todos los 'prints que se hacen en consola'.

- Hacer uso de los otros periféricos de ambas tarjetas como lo son los switches de la ZYBO, el otro botón de propósito general, el joystick en si y el LED RGB tanto de la tarjeta ZYBO como de la tarjeta Booster. De momento no hay pensada una utilidad para estos.

REFERENCES

[1] [7] H. Rosenfeldt, "ZYNQ: Adding an AXI timer to trigger periodic interrupts – harald's embedded electronics", Harald-rosenfeldt.de. [En línea]. Disponible en: https://www.harald-rosenfeldt.de/2018/01/15/zynq-adding-an-axi-timer-to-trigger-periodic-interrupts/. [Consultado: 11-dic-2023]. 4.