DISEÑO EN VHDL DE UNA INTERFAZ DE DATOS SERIE-DAC PARA UN CONTROL BARRA-ESFERA

VHDL DESIGN OF A SERIAL-DAC DATA INTERFACE FOR A BALL AND BEAM CONTROL

Juan Sifuentes Mijares
Tecnológico Nacional de México / Instituto Tecnológico de la Laguna
jsifuentesm@correo.itlalaguna.edu.mx

Jean Philipe Andre Paistel Sanchez
Tecnológico Nacional de México / Instituto Tecnológico de la Laguna
phil.ppe@outlook.com

Daniel Flores Montes
Tecnológico Nacional de México / Instituto Tecnológico de la Laguna
daniyfm5@gmail.com

Resumen

En este documento se presenta la descripción del desarrollo de un circuito interface realizado sobre una tarjeta FPGA SpartanXC3S200. El circuito digital está diseñado en VHDL; las operaciones que realiza el circuito son transferencia y conversión de datos entre dos computadoras. Todo el conjunto implementa un control por visión de un sistema barra-esfera. La interface recibe los datos del procesamiento digital de imágenes “PDI”, realizado en una primera PC. Los datos son la posición del centro geométrico de la esfera. Estos datos son agrupados y convertidos a señales analógicas, para ser transferidas a una tarjeta adquisitora PCI instalada en la segunda PC. Esta segunda computadora es usada para implementar los algoritmos de control y proporcionar la señal del par para el motor. La interface realiza también la decodificación del encoder de posición de la barra y la transforma a señal analógica para enviarla a una tarjeta PCI.

Palabras Claves: barra-esfera, control, FPGA, VHDL.

Abstract

In this paper, we present a description of a developed interface circuit which was created in a SpartanXC3S200 FPGA board. The digital circuit is design in VHDL;

the operations that the circuit develops are the transfer and conversion of data between two computers. All set implement a vision control of a ball & beam system. The interface receives the data from the image processing "PDI" developed by a first PC. The dates are position of geometric center of the ball. These data are grouped and converted to analog signals to be transferred to a data acquisition PCI card installed in a second PC. This second computer is used to implement the control algorithms and to provide the torque signal for the motor. The interface also performs the decoding of position encoder of the beam, and transforms it to analog signal to send to the PCI card.

Keywords: Control, FPGA, VHDL.

1. Introducción

El balanceo de una esfera sobre una barra es un sistema clásico de control sub actuado, ya que no se tiene directo control sobre la esfera, es indirecto atreves de un motor conectado a la barra. El problema de control de este sistema ha sido abordado de muchas formas [Oliveira, 2017], [Srivastava, 2015], [Boumazbar, 2015], [Ezzabi, 2013], [Ogata,2010], [Kim , 2007], y en el presente trabajo se hace mediante una cámara para determinar la posición de la esfera en la barra. Por otra parte también diversos trabajos sostienen la rentabilidad de trabajar con circuitos de lógica programable FPGAs y VHDL para mejorar las prestaciones de carga de trabajo y velocidad [Delgado, 2016], [Hernández, 2016], [Mohammad, 2014], [Pong, 2008], [Maxinez, 2006], [Brown, 2006]. Considerando los trabajos anteriores, se decidió ha repartido la carga de trabajo en varios sistemas y se ha utilizado un dispositivo de lógica programable FPGA. La distribución de tareas es de la siguiente forma (figura 1): En una primera computadora se realiza el procesamiento digital de la imagen y se determina la posición del centro de la esfera. En una segunda computadora se implementa el algoritmo de control que va ha ser probando. Un tercer sistema lo conforma la interface digital implantada en la tarjeta FPGA Spartan XC3S200 y diseñada en VHDL. Esta interface proporciona el enlace de datos de los dos primeros sistemas y también se encarga de leer el encoder de posición de la barra. Con la distribución de trabajo anterior

se logra aligerar la carga de trabajo en las computadoras y aumentar la velocidad de procesamiento haciendo posible el controlar la posición de la esfera en la barra.

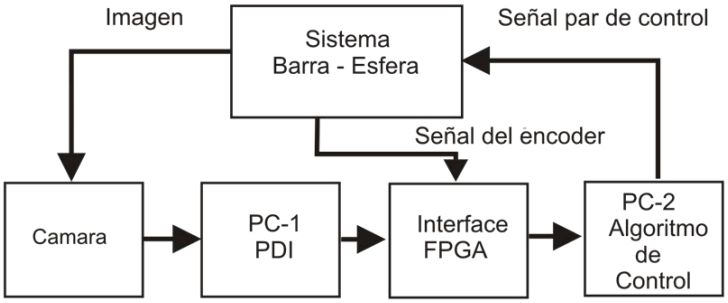


Figura 1 Diagrama del sistema completo de control por visión del sistema barra–esfera.

En si el trabajo que se presenta es: El diseño de la interface que hace posible la transferencia de datos entre el sistema de control y el sistema de PDI; y que en esta configuración de carga de trabajo distribuido, con un circuito de lógica programable FPGA usado para la interface, con esta configuración, si se consigue el funcionamiento del sistema del control barra esfera, ya que en previas pruebas, empleando configuraciones de carga de trabajo; en una sola computadora o dos sin interface, no fue posible lograr su funcionamiento. Adicionalmente la interface también realiza el trabajo del decodificador de encoder para la lectura de la posición del motor que mueve la barra sobre la cual rueda la esfera.

2. Métodos

A continuación, se explica la metodología que se siguió en la configuración para lograr el buen funcionamiento del sistema, así como los procedimientos empleados en los diseños de los diferentes componentes digitales y analógicos realizados para la interface.

Un diagrama a bloques del circuito digital de la interface es mostrado en la figura 2; en donde se muestran los puertos de entrada a la izquierda y puertos de salida a la derecha del bloque, las dimensiones son de 1 bit, 8 bits y 10 bits.

Entre los puertos de entrada se encuentran las terminales ‘A’ y ‘B’, usadas para la lectura del encoder de posición del motor que mueve la barra.

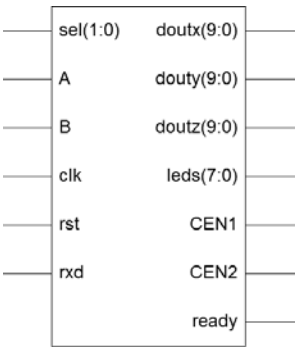


Figura 2 Diagrama externo de la interfaz serie-DAC.

La lectura de los datos de las coordenadas de posición X e Y están en formato serie RS-232 y provienen de la computadora que realiza el procesamiento digital de las imágenes (PDI), estas coordenadas llegan al puerto de entrada ‘rxd’. El puerto ‘rst’ tiene la función de reiniciar el circuito. El puerto ‘clk’ se alimenta con una señal de reloj de 50Mhz que llega a todos los procesos secuenciales y de sincronización. Las terminales ‘sel’ seleccionan algunas señales internas a ser monitoreadas mediante las terminales de salida ‘leds’. Los puertos de salida ‘doutx’, ‘douty’ y ‘doutz’, cada uno, proporcionan los bits de entrada para un convertidor R-2R. CEN1 y CEN2 son dos terminales que siempre tienen una señal fija en ‘1’, para inhibir un circuito de memoria que comparte algunas de las mismas terminales en la tarjeta de desarrollo que se usan para la interfaz.

La tarjeta de desarrollo usada para implantar el circuito digital es la tarjeta Spartan 3, que lleva un dispositivo de lógica programable FPGA XC3S3200, con un oscilador interno de 50 MHz. En los subtemas siguientes se explica el funcionamiento y código de diseño de los circuitos internos de la interface creada.

El circuito decodificador de encoder

Este circuito es el encargado de llevar la cuenta de los pulsos generados por el encoder de posición del motor que mueve la barra del sistema, figura 3. En el procedimiento de lectura del encoder se realiza un continuo muestreo de las señales A y B con dos registros de corrimiento, para almacenar las señales presentes y pasadas, este muestreo se realiza a una frecuencia de 50 MHz. Un circuito comparador se encarga de detectar las diferencias entre los registros

presente y pasado, en cualquiera de las señales A o B. El mismo circuito comparador también produce un incremento o decremento del registro que lleva posición, según corresponda al movimiento del encoder y que está directamente relacionado con la secuencia de cambios en de las señales A y B.

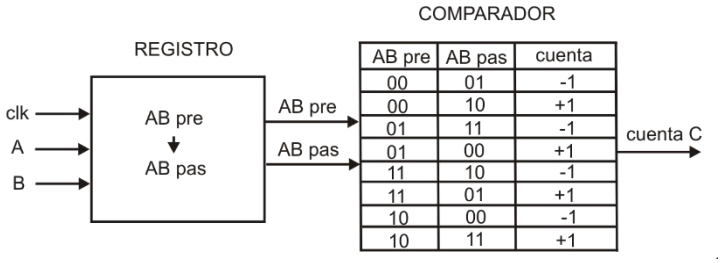


Figura 3 Diagrama funcional del circuito decodificador de encoder.

El aumento en la precisión usando este método se puede ver si consideramos que un pulso completo de la señal A o B implica dos cambios para volver al mismo estado, y puesto que son dos señales del encoder, entonces suman 4 cambios por cada pulso completo, este procedimiento multiplica la precisión del decodificador del encoder por 4. La frecuencia de muestreo limita en velocidad a las señales entrantes A y B del decodificador de encoder, de tal forma que no podría funcionar correctamente si este límite es superado.

Un diagrama funcional del circuito decodificador de encoder es mostrado en la figura 3 y a la vez el código en VHDL del proceso que implementa este circuito es mostrado tabla 1, en este se pueden distinguir las señales ABpre y ABpas, las cuales son las señales A&B muestreadas presentes y pasadas. El funcionamiento es sencillo y es que cuando estas son diferentes estas producen un cambio +1 o -1 en la cuenta, esto se puede observar en la estructura del case de esa misma tabla 1.

El circuito de recepción serie RS-232

La figura 4 muestra un diagrama funcional del circuito de recepción serie. Las características de este circuito de comunicación diseñado son: velocidad de 115200 bits/s con longitud de los datos 8 bits y un ‘1’ en el bit de alto.

El circuito utiliza un pequeño registro de corrimiento en el puerto de entrada de los datos serie rxd en conjunto con una compuerta and de 8 bits para dar la salida. Este registro funciona como un filtro simple para el ruido de alta frecuencia y su funcionamiento es simple.

Tabla 1 Código en VHDL del proceso para crear el circuito decodificador de encoder.

```
process(rst, clk, ABpas, ABpre)
begin
 if rst = '1' then
  cuenta <= (others => '0');
  ABpre <= A&B;
  ABpas <= A&B;
  elsif clk'event and clk = '1' then
  ABpre <= A&B;
  ABpas <= ABpre;
  case ABpre is
  when "00" => if ABpas = "01" then
          cuenta <= cuenta -1;
          elsif ABpas = "10" then
          cuenta <= cuenta +1;
          end if;
  when "01" => if ABpas = "11" then
          cuenta <= cuenta -1;
          elsif ABpas = "00" then
          cuenta <= cuenta +1;
          end if;
  when "11" => if ABpas = "10" then
          cuenta <= cuenta -1;
          elsif ABpas = "01" then
          cuenta <= cuenta +1;
          end if;
  when "10"  => if ABpas = "00" then
          cuenta <= cuenta -1;
          elsif ABpas = "11" then
          cuenta <= cuenta +1;
          end if;
  when others =>
         cuenta <= (others => '0');
end case;
end if;
end process;
```

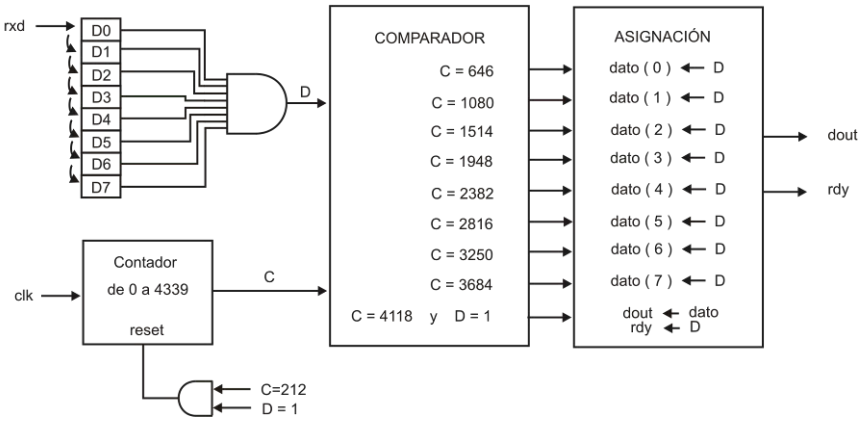


Figura 4 Diagrama funcional del circuito receptor de datos serie.

Para que la salida vaya a ‘1’ se necesita que todos los bits en el registro sean unos, si hay un cambio a cero antes de esto, es decir ruido de una frecuencia alta, tal que no pueda mantenerse un ‘1’ por lo menos por 8 pulsos del reloj, entonces no se producirá el cambió a 1 en la salida, eliminando todos esos cambios de alta frecuencia. El formato de datos serie usado es un bit de inicio ‘0’, seguido de 8 bits de datos iniciando con el menos significativo y al final un ‘1’ como el bit de alto.

El proceso de recepción inicia con un contador que se encuentra detenido, en valor cero, esperando un bit de inicio en el puerto ‘rxd’ para dar inicio a un conteo hasta la cantidad de 4339. El conteo dura el mismo tiempo que tarda una transferencia de datos serie completa, incluidos los bits de inicio y de alto, con una señal de reloj de 50 MHz, siendo entonces la velocidad de comunicación 115200 bits/s. Un circuito comparador es el encargado ir asignando cada dato en registro de almacenamiento en el tiempo correcto, y también se encarga de dar la salida de datos en paralelo cuando se termina la recepción. La terminal de salida ‘rdy’ informa con un ‘1’ cuando los datos recibidos están completos y listos en el puerto paralelo ‘dout’. En tabla 2 se pueden observar dos procesos, uno que implementa dos circuitos; el primero contiene un contador y un registro de corrimiento para los datos de llegada serie, y el segundo implementa circuito un comparador para la asignación y lectura de los datos en serie, además fuera de los procesos, están las líneas de código que implementan el circuito filtro con compuertas AND.

Tabla 2 Código VHDL del receptor serie.

```
process(clk,rst,contador, C)
begin
if clk='1' and clk'event then
   bufer  <=  bufer(6  downto  0)  &
rxd;
  if rst='1'  then contador <= 0;
  else
  if( ((contador <= 212) and ( C =
'1'))
  or (contador >= 4339)) then
        contador <= 0;
   else   contador <= contador + 1;
  end if;
 end if;
end if;
end process;

C  <=  bufer(0)  and  bufer(1)  and
bufer(2) and bufer(3) and bufer(4)
and  bufer(5)  and  bufer(6)  and
bufer(7);
```

```
process(contador,C)
begin
case contador is
when   646 => dato(0) <= C; aux_rdy <=
'0';
when  1080 => dato(1) <= C; aux_rdy <=
'0';
when  1514 => dato(2) <= C; aux_rdy <=
'0';
when  1948 => dato(3) <= C; aux_rdy <=
'0';
when  2382 => dato(4) <= C; aux_rdy <=
'0';
when  2816 => dato(5) <= C; aux_rdy <=
'0';
when  3250 => dato(6) <= C; aux_rdy <=
'0';
when  3684 => dato(7) <= C; aux_rdy <=
'0';
when  4118 =>
  if  C = '1' then dout <= dato;
                        rdy <= C;
  end if;
when others =>
end case;
end process;
```

El circuito de entramado de datos serie

Los datos que llegan en formato serie son los datos provenientes del PDI que calcula el centro de la imagen de la esfera, las coordenadas cartesianas X e Y.

Cuando un dato serie está completo este se pasa a por un circuito que se encarga de identificarlo. El circuito primero espera por el carácter de inicio de la trama de datos. Una vez que el dato de inicio llegó, el siguiente dato corresponde a la parte menos significativa de la coordenada X, esta se ubica completa en la parte baja de un registro de 10 bits, después llega la parte alta de X y esta se recorta a 2 bits y se ubica en la parte alta del mismo registro, completando 10 bits y continuando con la coordenada Y de igual forma. Al terminar se da salida de datos en paralelo.

En la figura 5 se puede observar el diagrama de estados que explica el funcionamiento del circuito de entramando de datos. En este podemos observar que la señal rdy realiza la función como una la señal de reloj. La señal de rst nos lleva a un estado inicial en donde se queda en espera hasta que el carácter de inicio de trama aparezca, este es una "@". Una vez que pasa el primer estado la trama de datos llega en forma consecutiva sin ninguna otra condición. La señal Ds representa los datos de la trama.

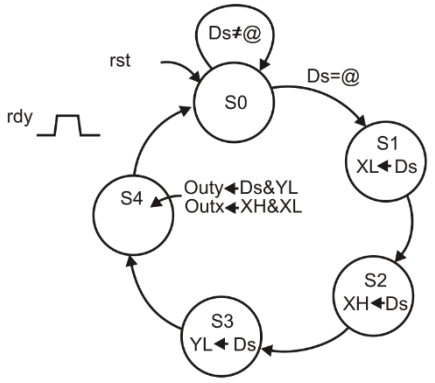


Figura 5 Diagrama de estados del circuito para entramado de datos serie.

En tabla 3 se tiene el código que sintetiza el circuito de entramado de datos. En este se ha implementado una máquina de estados con dos procesos; un primer proceso con un case anidado con la condición de un flanco de subida en la señal rdy es el que implementa las acciones para cada estado, y se observa que solo en primer caso se tiene el condicionante if preguntando por un "01000000" para avanzar al siguiente caso. Mientras que en todos los otros casos solo se guarda el dato y se avanza al siguiente, volviendo caso inicial al terminar. El segundo

proceso de la tabla 3 es el circuito que lleva la actualización del estado presente por el estado siguiente de la máquina de estados usando la señal clk para la actualización de estos.

Tabla 3 Código VHDL del circuito de entramado de datos X e Y.

```
process(edopre, rdy, dout )
begin
 if rdy = '1' and rdy'event then
   case edopre is
      when s0 =>
AX <= "00000000";
if dout = "01000000" then
   edosig <= s1;
else
   edosig <= s0;
end if;
      when s1 =>
AX <= "00000001";
XL <= dout;
edosig <= s2;
   when s2 =>
AX <= "00000010";
XH <= dout(1 downto 0);
edosig <= s3;
      when s3 =>
AX <= "00000011";
YL <= dout;
edosig <= s4;
      when s4 =>
AX <= "00000100";
doutx <= XH & XL;
douty <= dout(1 downto 0)& YL;
edosig <= s0;
      when others =>
  end case;
 end if;
end process;

process(clk,rst)
begin
if rst = '1' then  edopre <= s0;
elsif clk'event and clk='1' then
  edopre <= edosig; end if;
end process;
```

El circuito de monitoreo de señales internas

La tabla 4 muestra el código para el diseño del circuito selector de datos de salida a los leds de la tarjeta de desarrollo Spartan 3.

Tabla 4 Código VHDL para monitoreo de señales internas.

```
Leds      <=        cuenta(7 downto 0)              when sel ="00"
   else   ABpre & "0000"&cuenta(9 downto 8)   when sel = "01"
   else   dout                                when sel ="10"
   else   AX ;
doutz <= cuenta;
```

El circuito es implementado con la finalidad de revisar el buen funcionamiento de los procesos internos, ya implementados físicamente, este incluye la selección de la cuenta, mostrando los primeros 8 bits del contador de pulsos del decodificador de encoder con sel = “00”, después con sel =”01” se seleccionan las señales AB actuales leídas del encoder, y los dos bits más altos del resto de la cuenta del decodificador de encoder, luego con sel = “10” se da salida al dato serie recibido y finalmente con sel = “11” se muestra AX que es un indicador del estado presente en la máquina de estados del circuito de comunicación, que se tiene en tabla 3.

El circuito convertidor de señal digital a señal analógica

Tres circuitos de conversión digital a señal analógica DAC (Digital Analogo Conversión) tipo R-2R se han implementado en los puertos de salida digital de la tarjeta Spartan3. Los datos que alimentan a los tres circuitos DAC R-2R son las coordenadas de posición de la esfera X, Y y la cuenta del decodificador del encoder que lleva la posición de la barra.

La precisión del convertidor está dada por la calidad y cantidad de voltaje con el que se alimenta el circuito y la cantidad de bits que se han usado como datos de entrada. En este caso se ha utilizado el voltaje de la tarjeta 3.3 VDC y 10 bits de datos para la resolución, esto proporciona incrementos de 3.3 / 2^10 VDC = 3.22 mV, es decir 1024 posibles cambios en la escala de 0 a 3.3 VDC. Adicionalmente se han colocado dos capacitores a tierra para reducir el ruido; uno en el voltaje de alimentación de 100 uF y otro de 0.1 uF en la terminal de salida de conversión del DAC.

3. Resultados

El circuito diseñado e implementado sobre una tarjeta Spartan 3 XC3S200 encapsulado ft 256 y grado de velocidad -4; presenta como datos de consumo los resultados siguientes:

Número de Slices:	*93*	*out of*	*1920*	*4%*
Número de Slice Flip Flops:	*102*	*out of*	*3840*	*2%*
Número de 4 input LUTs:	*138*	*out of*	*3840*	*3%*
Número de IOs:	*48*			
Número de bloques IOBs usados:	*48*	*out of*	*173*	*27%*
IOB Flip Flops:	*2*			
Número de GCLKs:	*2*	*out of*	*8*	*25%*

Los datos anteriores permiten ver que los recursos utilizados de la tarjeta de desarrollo fueron menores al 30%. También se muestra el resumen de resultados en consumo de tiempo y velocidad en la implementación, siendo los siguientes:

- Periodo mínimo: 8.130ns (Frecuencia máxima: 123.001 MHz).
- Tiempo mínimo de llegada antes del reloj: 5.422 ns.
- Salida máximo tiempo requerido después del reloj: 9.559 ns.

- Máximo retardo de ruta combinacional: 9.686 ns.

Un controlador tipo PID fue implementado en el sistema de control para probar el funcionamiento del sistema completo. La esfera fue colocada a 10 cm del punto de equilibrio al iniciar el experimento y se le aplicó una perturbación a los 50 s. Los valores usados en las ganancias fueron: kp = 0.131 Nm, Kv = 0.00355 Nm-s/rad y Ki = 0.05 Nm/s-rad. La figura 6 muestra la gráfica de posición de la esfera.

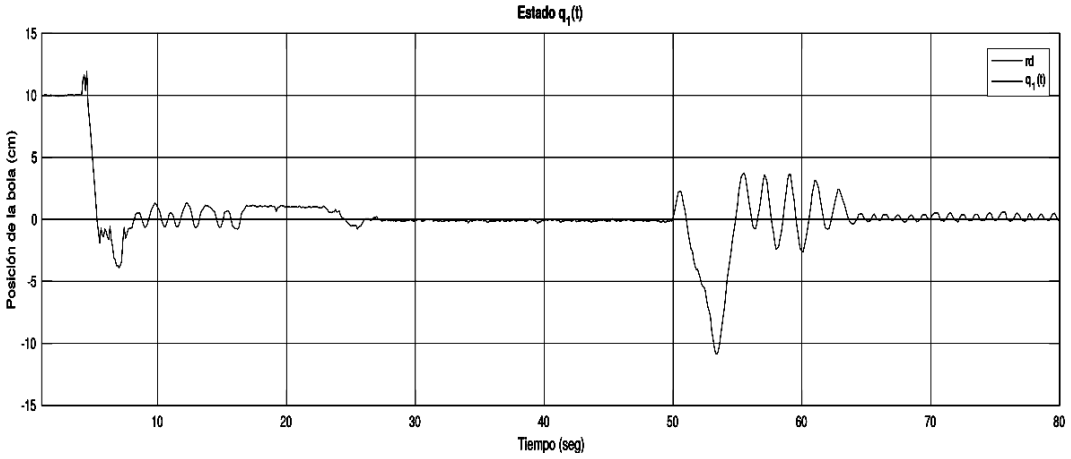


Figura 6 Gráfica de la posición de la esfera para un control PID, con perturbación.

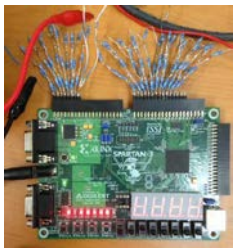
En figuras 7a y 7b se muestran dos fotografías de la tarjeta Spartan 3 en donde se implementó todo el circuito de la Interfaz Serie–DAC y los convertidores R-2R, en figura 7c se puede observar el sistema físico barra esfera, por último, en figura 7d se presenta una pantalla de la computadora con el algoritmo de control en simulink Matlab, utilizado para probar el funcionamiento del control por visión para un sistema barra–esfera,

4. Discusión

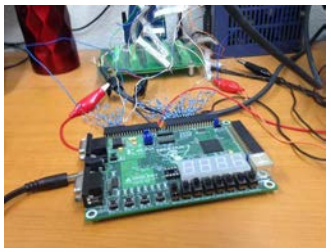
En primera instancia de forma global se diseñado e implantado una interface que realiza la transferencia de información para dos sistemas que realizan el control por visión de un sistema barra-esfera.

Hay que mencionar que todos los circuitos digitales que se han diseñado para este trabajo tienen prestaciones que cumplen adecuadamente con las exigencias del

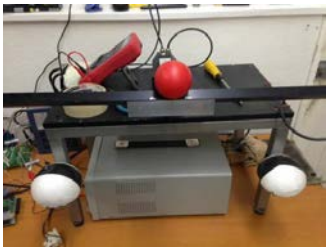
sistema y en algunos casos como el decodificador de encoder superan por mucho las exigencias mínimas de funcionamiento. Todos los circuitos diseñados en VHDL, son diseños propios en VHDL, con los cuales es posible reproducir totalmente todos los circuitos digitales de la interface.



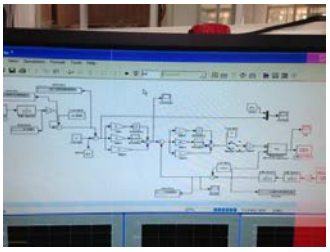
a) Tarjeta SPARTAN3 XC3S200.



b) Interfaces conectadas.



d) Sistema barra–esfera.



d) Diagrama de control en Matlab.

Figura 7 Fotos del sistema físico implementado y del sistema de control barra esfera.

5. Conclusiones

Se diseñó e implementó un circuito de una interface de datos serie a DAC, con decodificador de encoder, para ayudar en reducción de la carga de trabajo distribuido entre dos computadoras y de esta forma con todos los elementos en conjunto se consigue el buen funcionamiento del control por visión del sistema barra – esfera. Demostrando así que los dispositivos de lógica programable, en este caso la tarjeta de desarrollo Spartan3 XC3S200 y el lenguaje de descripción de hardware VHDL presentan facilidad para diseñar e implementar circuitos digitales con la versatilidad y buen desempeño que se ha mostrado. Revisando las evidencias del funcionamiento del sistema de control completo presentas, los resultados obtenidos son adecuados y satisfactorios. Las prestaciones de desempeño y velocidad proceso de control se han aumentado al redistribuir entre más elementos la carga de trabajo de los sistemas, esta configuración funcional fue posible gracias a la interface diseñada.

6. Bibliografía y Referencias

[1] Boumazbar, S., Bouallégue, S., Haggége, J. Co-simulation and rapid prototyping of fuzzy supervised PID controllers based on FPGA-Nexys2 board, pp. 1-6, Tunes, 2015.

[2] Brown, S. D., Vranesic, Z. G. Fundamentos de lógica digital con VHDL, McGraw Hill, 2006.

[3] Delgado J., García M., Posada R., Aguilar I. Implementación de un sistema de adquisición de imágenes embebido en un FPGA, Pistas Educativas No. 120, 1578-1595, México, 2016.

[4] Ezzabi, A. A., Cheok K. C., Alazabi F. A. A nonlinear backstepping control design for ball and beam system. International Midwest Symposium on Circuits and Systems (MWSCAS), 56th, 1318 - 1321, USA, 2013.

[5] Hernández S., Sifuentes J., Paistel J. Diseño digital de un control de posición para un motor en VHDL y LABVIEW usando FPGA, Pistas Educativas No. 120, 227-243, México, 2016.

[6] Kim J. y Jung, J. S. Hardware implementation of nonlinear PID controller with FPGA based on floating point operation for 6-DOF manipulator robot arm, Automation and Systems, Corea del sur, 2007.

[7] Maxinez D. G. VHDL el arte de programar sistemas digitales, CECSA, 2006.

[8] Mohammad A. Zare; Rajesh G. Kavasseri; Cristinel Ababei, FPGA-based design and implementation of direct torque control for induction machines, 2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14), pp. 1-6, 2014.

[9] Ogata, K. Ingeniería de control moderna, Madrid: PEARSON, 2010.

[10] Oliveira J., Balula, S., Fernandes, H. Ball & beam experiment control with current sensing, 4th, Experiment@International Conference (exp.at'17), 258 - 263, Portugal, 2017.

[11] Srivastava, A., Pratap B. Nonlinear robust observers for ball and beam system: A comparative analysis, 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS), 1 - 6, India, 2015.