CONTROL DE AUTO MEDIANTE PLD EN LENGUAJE VHDL

José Luis Pérez Justo
joseurp2015@hotmail.com

Escuela Profesional de Ingeniería Electrónica
Universidad Ricardo Palma
Taller de Electrónica I

Resumen: *El presente artículo muestra el diseño e implementación de un robot móvil seguidor de línea, producto del desarrollo de la práctica de laboratorio correspondiente a la asignatura de Taller 1 del quinto semestre académico del Programa de Ingeniería Electrónica de la Universidad Ricardo Palma. El objetivo general del proyecto final es identificar y aprender la importancia del desarrollo de aplicaciones basadas en circuitos digitales programables PLD, abordando los conceptos de Lenguaje de Descripción de Hardware VHDL y herramientas de desarrollo como ISPLever Starter de la empresa Lattice Semiconductors. Los resultados obtenidos permitieron desarrollar habilidades para el diseño de aplicaciones basadas en lógica combinacional para una aplicación en particular del área de Robótica, y además comprender las ventajas con respecto a la implementación de aplicaciones digitales utilizando circuitos digitales cableados convencionales. El dispositivo lógico programable utilizado para el control del robot móvil fue la GAL22V10, el sensor CNY70 fue utilizado para la detección de línea.*

ABSTRACT: *This article shows the design and implementation of a mobile robot following the line, product of the development of the laboratory practice corresponding to the Workshop 1 subject of the fifth academic semester of the Electronic Engineering Program of the Ricardo Palma University. The general objective of the project is to identify and learn the importance of development in the development of applications in programmable digital circuits, addressing the concepts of VHDL Hardware Description Language and development tools such as ISPL start the startup of the company Lattice Semiconductors. The results were allowed. The results were allowed. The results were reduced. The programmable logic device used to control the mobile robot was the GAL22V10, the CNY70 sensor was used for line detection.*

1. Introducción

Si bien la robótica es un área que actualmente está siendo utilizada en un sin número de aplicaciones industriales, domésticas y educativas, trabajar con robots exige un conocimiento multidisciplinar de la electrónica, la informática y hasta la mecánica. Conocimiento de sensores, comunicaciones, motores e incluso inteligencia artificial hace de esta disciplina un excelente elemento formativo para estudiantes y profesionales, colocando en práctica los conocimientos en cada una de las temáticas anteriormente mencionadas, además de generar el deseo y la motivación de querer aprender nuevas técnicas y métodos que hagan del robot un sistema más autónomo e inteligente.

2. Presentación del Problema

El desarrollo de circuitos con lógica digital combinacional y secuencial con los diferentes integrados se hace muy tedioso y costoso, debido al espacio y cableado realizado para circuitos muy complejos, entonces. ¿cómo se podría simplificar y poder ser más eficiente?

3. Descripción de la Solución

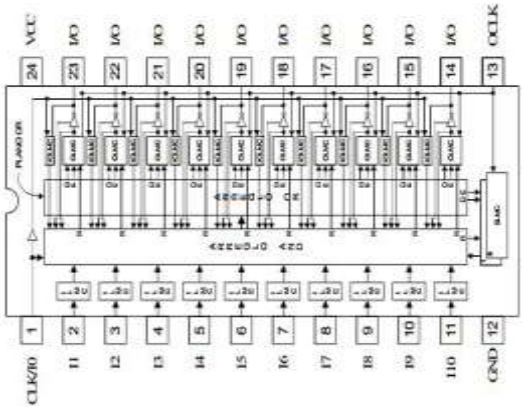
Cuanto más complejo es un circuito, más probabilidades hay de que alguna de sus partes falle. Puesto que los PLDs reducen el número de chips en los sistemas, la probabilidad de un fallo disminuye.
Los circuitos impresos con menor densidad de CI son más fáciles de construir y más fiables. Las fuentes de ruido también se reducen.

4. Fundamento Teórico

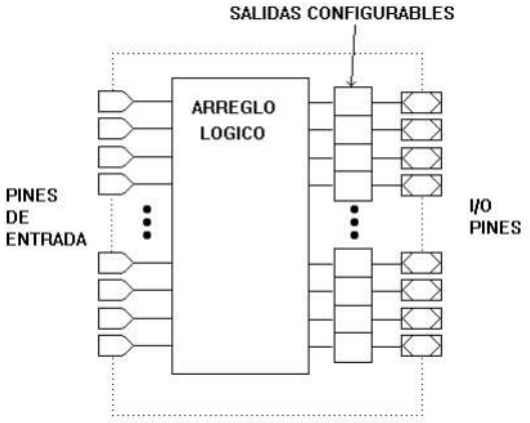
Un Dispositivo Lógico Programable (PLD) es un Chip LSI que contiene una estructura de circuito “regular”, pero que permite al diseñador adecuarlo para una aplicación específica.
Un dispositivo programable por el usuario es aquel que contiene una arquitectura general predefinida en la que

el usuario puede programar el diseño final del dispositivo empleando un conjunto de herramientas de desarrollo.
Las arquitecturas generales pueden variar, pero normalmente consisten en una o más matrices de puertas AND y OR para implementar funciones lógicas.
Muchos dispositivos también contienen combinaciones de flip-flops y latches que pueden usarse como elementos de almacenaje para entrada y salida de un dispositivo.
Los dispositivos más complejos contienen macro celdas, que permite al usuario configurar el tipo de entradas y salidas necesarias en el diseño.

En determinadas aplicaciones, un PLD puede sustituir desde unos pocos hasta unas decenas de CI de función fija, mientras que los grandes ASICs pueden sustituir a cientos e incluso miles de CI.
En ocasiones, los PLD se utilizan para realizar prototipos que posteriormente se llevarán a un ASIC más complejo.
El trabajo con PLDs proporciona: facilidad de diseño, prestaciones, fiabilidad, economía y seguridad.



Las herramientas de soporte al diseño con PLDs facilitan enormemente este proceso. Las hojas de codificación que se utilizaban en 1975 han dejado paso a los ensambladores y compiladores de lógica programable (PALASM, AMAZE, ABEL, CUPL, OrCAD/PLD, VHDL etc).
Estas nuevas herramientas permiten expresar la lógica de los circuitos utilizando formas variadas de entrada tales como: ecuaciones, tablas de verdad, procedimientos para máquinas de estados, esquemas, etc.



La simulación digital posibilita la depuración de los diseños antes de la programación de los dispositivos.
Todo el equipo de diseño se reduce a un software de bajo costo que corre en un PC, y en un programador.

5. Diagrama Circuital

En la Fig. 1 se muestra el esquema general de la arquitectura del robot móvil implementado.

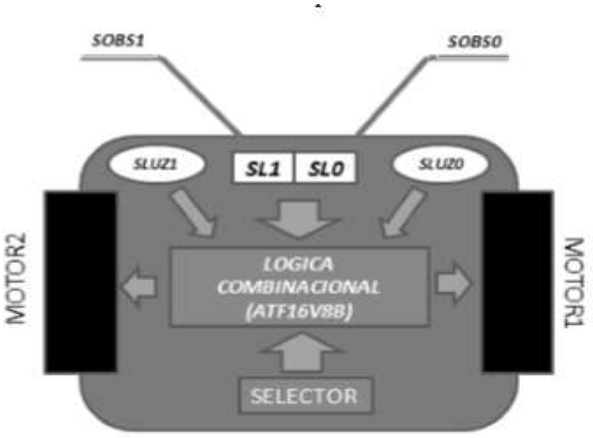
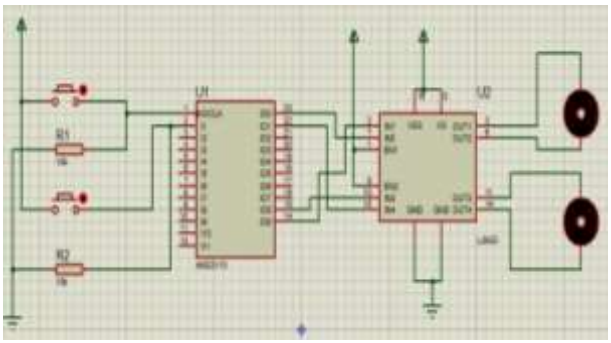


Fig. 1. Estructura del robot móvil



Simulación en Proteus

Programación del PLD, con lenguaje VHDL:

```
LIBRARY ieee;
LIBRARY generics;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
USE generics.components.ALL;

ENTITY testbench IS
END testbench;

ARCHITECTURE behavior OF testbench IS

        COMPONENT seguidor
        PORT(
             E : IN std_logic;
             S : OUT std_logic
             );
        END COMPONENT;

        SIGNAL E :  std_logic;
        SIGNAL S :  std_logic;

BEGIN

        uut: seguidor PORT MAP(
             E => E,
             S => S
        );


-- *** Test Bench - User Defined Section ***
   tb : PROCESS
   BEGIN
      wait; -- will wait forever
   END PROCESS;
-- *** End Test Bench - User Defined Section ***

END;
```

Tabla lógica de control de motores para seguidor de línea ENTRADAS SALIDAS SL1 SL0 MOTOR1 MOTOR 2.

ENTRADAS		SALIDAS			
SL1	SL0	MOTOR1		MOTOR 2	
0	0	1	0	1	0
0	1	0	1	1	0
1	0	1	0	0	1
1	1	0	1	0	1

Seguidor de Línea La función de seguidor de línea solo tendrá en cuenta los estados lógicos digitales entregados por los sensores ópticos dispuestos para tal fin. El acondicionamiento electrónico realizado a cada sensor óptico para que entregue dichos estados digitales se muestra en la Fig. 2.

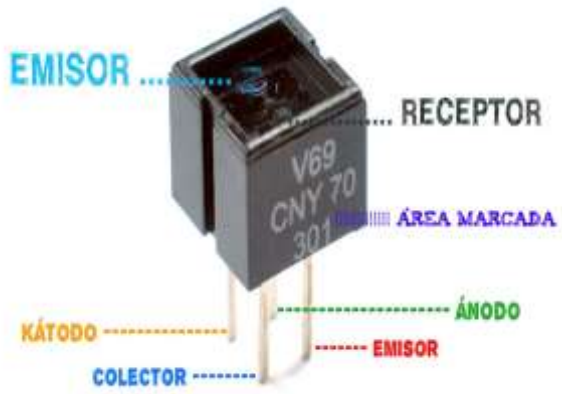


Figura 2.

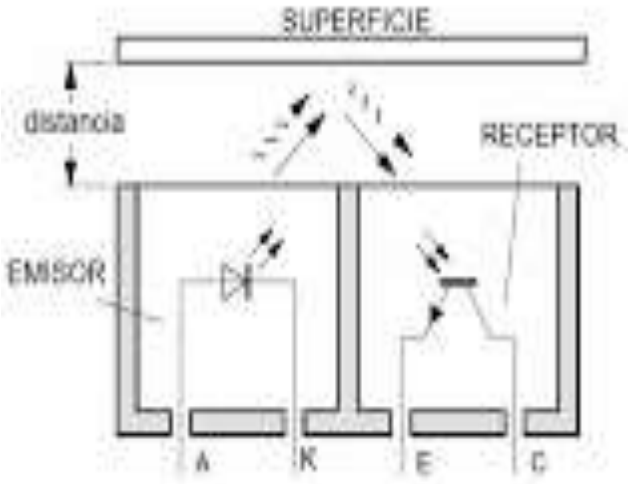


Figura 3.

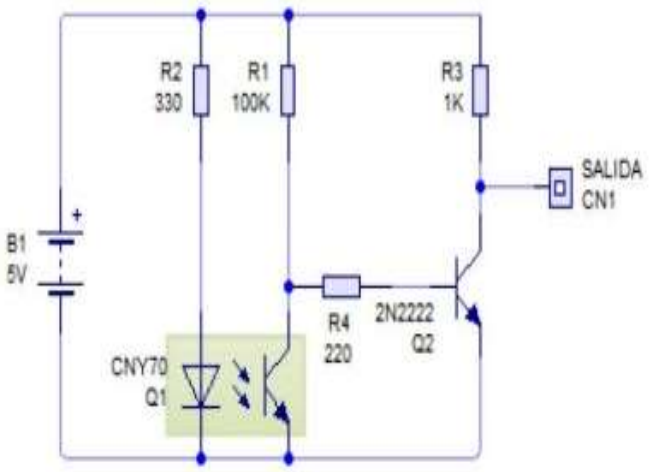
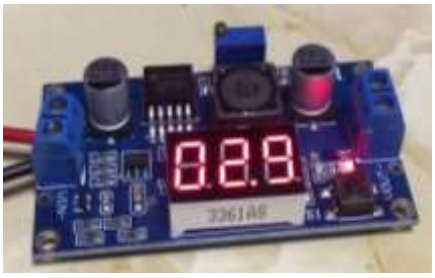
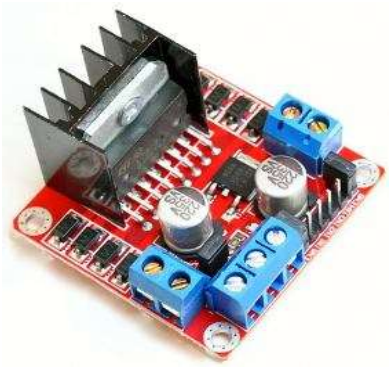


Figura 4.

6. Componentes y Costos

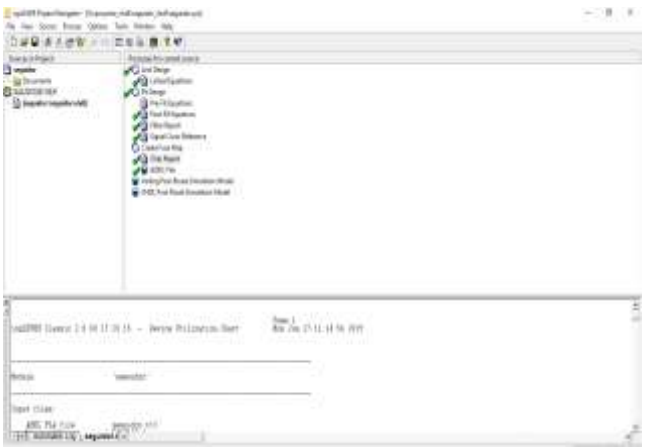
Componente	Unidad	Costo Unitario S/.	Costo S/.
Modulo cny70	1	4.00	4.00
Driver de motor	1	7.00	7.00
Regulador de voltaje	1	8.00	8.00
GAL 22V10	1	20.00	20.00
Motorreductores y llantas	2	7.50	15.00
Rueda deslizable	1	10.00	10.00
Batería de 9V	1	8.00	8.00
Costo total S/.			72.00





7. Procedimiento

Primero desarrollamos la programación en ispLEVER Project Navigator.



Segundo paso, grabar el PLD.

Tercero, armar y montar el circuito con el driver de motor, los módulos CNY 70, regulador de voltaje, gal22v10, etc.

Figura 5.

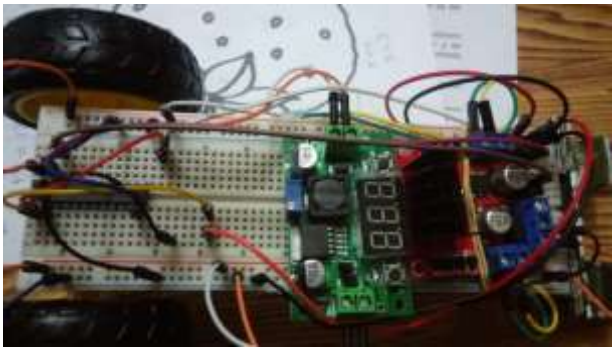


Figura 6.

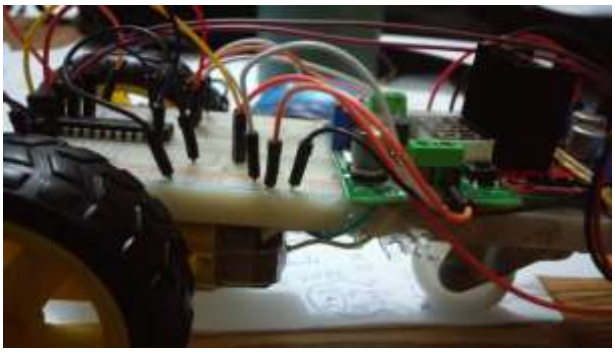


Figura 7.

8. Resultados

La corrección lateral de desplazamiento en el estado 10 de la variable “Seguidor de Línea” señala que es necesario activar el motor de la llanta de la derecha sin que se active el motor de la llanta de la izquierda. Con este paso se obtiene un desplazamiento en giro hasta lograr la corrección deseada. Una vez finalizada la corrección lateral, se registrará un cambio de estado con lo que se reactivará el motor de la llanta inactiva por la corrección.
El mismo esquema se aplica para el desplazamiento en giro contrario. Este mismo esquema o comportamiento se aplica del mismo modo en el sentido contrario para la corrección lateral de la izquierda. Cuando se presente el estado 00, el móvil tendrá que retroceder lo suficiente en dirección recta hasta hallar la última posición en la que se detectó línea sobre el trayecto. Cuando el carro se encuentre sobre la línea, se mantiene el curso hacia el frente (avanzando), por lo tanto, el estado descrito ideal corresponde a 11 en el vector (variable) mencionado.

9. Conclusiones

- VHDL es un lenguaje de gran versatilidad y potencial que permite describir de manera sencilla y eficaz funciones básicas para el manejo de proyectos como el móvil previamente descrito.

- Es indispensable probar cada función por separado y luego conformar en un solo bloque todo el contenido de programación para poder depurar el programa en caso de errores.

- La programación de dispositivos, en base a aplicaciones de software, permite generar programas de proyectos de manera más rápida y eficiente, puesto que el diseño es más orientado al algoritmo desarrollado a nivel de software, y los inconvenientes de implementación y cableado pasan a ser problemas menos tediosos y difíciles de corregir.

10. Bibliografía

[1] J. Garza, Sistemas digitales y electrónica digital, Ed. Pearson, 2006.

[2] D. Maxinez, J. Alcala, Ed. VHDL, El arte de programar sistemas digitales, Ed. Cecsa,2002.

[3] A. Gradiaga, J. Perez, Diseño de procesadores con VHDL, Ed. Universidad Alicante,2007.

[4] T. Floyd, Sistemas digitales, Ed. Pearson, 2016.

[5] R. Tocci, N. Widmer, G. Moss, Sistemas Digitales, Ed. Pearson, 2010.

[6] F. Pardo, J, Boluda, VHDL, Lenguaje para síntesis y modelado del circuito, Ed. Alfaomega, 2008.

[7] R. Cayssials, Sistemas embebidos en FPGA, Ed. Alfaomega, 2014.