**PID control for DC motor based on FPGA**

Viviana Mero1, Jean Zamora2

Escuela Superior Politécnica del Litoral, ESPOL Facultad de Ingeniería en Electricidad y Computación,

Campus Gustavo Galindo Km 30.5 Vía Perimetral, Guayaquil, Ecuador

{vmero1, jpzamora2}@espol.edu.ec

*Abstract*— Este trabajo presenta el diseño e implementación de un controlador proporcional – integral – derivativo (PID) para controlar la velocidad de un motor DC utilizando una FPGA como microcontrolador principal. La implementación técnica usada para lograr un sistema con retroalimentación en tiempo real es la interacción entre interfaces analógicas y digitales. El hardware principal del trabajo consta de una tarjeta D10 Standard, con varios procesadores realizando la lectura, la modulación PWM y el bloque PID. Para obtener la data del motor se utilizó encoder óptico para contrar las revoluciones por minutos del motor DC

Keywords— Controlador PID, motor DC, FPGA, D10 Standard, control de velocidad, PWM

# Introduction

Un motor eléctrico es un artefacto básico para muchas de las aplicaciones que solucionan problemas del diario vivir como de la industria, por ejemplo en los ventiladores, electrodomésticos, unidades de disco, vehículos eléctricos, laminadoras de acero, etc. [1].

El motor DC es el encargado de convertir la energía eléctrica en energía mecánica, mediante un movimiento rotatorio, estos motores son frecuentemente utilizados en aplicaciones donde la velocidad del motor requiere ser controlado externamente. En la implementación de cualquier aplicación en donde se utilice un motor DC es importante diseñar un controlador, para lo cual es necesario tener en consideración la velocidad y el estado del motor [2]. Una de las razones más importantes de controlar la velocidad de un motor es por motivos de mejorar la eficiencia del sistema y para alcanzar la velocidad deseada.

Existen muchas técnicas para controlar la velocidad de un motor DC, pero en el presente trabajo se abordará un controlador PID juntos con un control por ancho de pulso PWM, ya que su estructura es bastante simple y se puede configurar cada parámetro del controlador [3].

Un controlador PID soluciona de forma eficiente el problema de un control preciso de la velocidad ya que este es muy receptivo a los cambios de velocidad y los cambios de carga. La velocidad del motor es proporcional al voltaje de alimentación de este, con el controlador PID se puede modificar el ciclo de trabajo de los pulsos PWM, lo que representan la amplitud de la velocidad que se necesita para llegar a la velocidad que se desea [4].

En la actualidad existen muchos tipos de encoders que son utilizados en el control de un motor DC, como los encoders ópticos, lineales, absolutos, incrementales y de cuadratura, estos dispositivos son frecuentemente utilizados porque convierten el movimiento mecánico del motor en señales o pulsos digitales, el encoder óptico es uno de los más utilizados y su funcionalidad está basada en un detector de luz, este tipo de encoder se lo suele utilizar junto con un disco montado en el eje giratorio del motor.

El sistema que se va a implementar en el presente trabajo consta principalmente del diseño de un controlador PID en una FPGA que por medio de un lazo de retroalimentado nos permita regular la velocidad del motor, eliminando su error y volviéndolo más estable. La adquisición de datos para obtener el modelo matemático del sistema, donde se utilizará un microcontrolador junto con el motor DC y el encoder óptico para la medición de las revoluciones por minutos del motor y por último se utilizará un motor driver L298n para controlar el ciclo de trabajo del motor por medio de una señal PWM y de esta forma obtener la velocidad deseada.

# Related work

Para el control de la velocidad de un motor de corriente continua se tienen varias opciones de microcontroladores como los PIC para modular por ancho de pulso el ciclo de trabajo para obtener la velocidad deseada [5]. Cuando se realiza el control con la FPGA se tienen 2 bloques principales el circuito controlador y el motor DC, en donde el circuito controlador está compuesto por elementos pasivos, mientras para la adquisición de datos se utiliza un encoder de cuadratura [6]. Otra alternativa para desarrollar un controlador PID es utilizando el módulo LabVIEW FPGA e instalándolo en Xilinx Spartan-3E algunas veces sin la necesidad de crear código VHDL con la ventaja de realizar varios procesos a la vez [7]. Implementar un controlador digital PID en FPGA permite evitar los problemas cuando interactúan señales analógicas y digitales en tiempo real [8] [9], en algunos trabajos se muestra la velocidad en un display LCD mientras que en el presente trabajo se mostraran los datos en un servidor web montado en el procesador ARM. Muchos de estos trabajos realizan el control en lazo cerrado, en donde leen el error de un convertidor analógico digital [10].

En el presente trabajo se utiliza varios procesadores para dividir las tareas, el primer procesador es el encargado de la identificación del sistema, el segundo procesador se implementará el controlador PID y finalmente con el procesador ARM se montará un servidor web en el cual se mostrarán los datos en tiempo real.

# DataSet

Para este trabajo, los datos de entrada provienen del encoder óptico, el cual esta conectado a una placa de Arduino. El encoder óptico es el que envía pulsos digitales con lo que se puede contar las revoluciones por minuto del motor. El conjunto de datos tiene una frecuencia de muestreo de 100 Hz, los pulsos que están representados por ceros o unos, serán enviados al primer procesador Nios II y serán almacenados para identificar el modelo matemático de la planta. El motor que se está utilizando para este proyecto es un motor de DVD cuyas RPM oscilan desde las 500 RPM hasta las 2700 RPM. Para poder hacer el control por ancho de pulso se utiliza el motor driver L298N, que controlará el ciclo del trabajo del motor para obtener la velocidad deseada. En el segundo procesador Nios II, se codificará en lenguaje C un controlador PID en un circuito de lazo cerrado. Dado que se requiere mostrar para visualizar los datos en un servidor web, se utilizará el procesador ARM, para visualizar la data y gráficas en tiempo real.

Con la ayuda de un cronómetro se midió el tiempo que el motor DC se demora en llegar a su máxima velocidad (PWM = 100%) y el tiempo que tarda en detenerse (PWM = 0%). Con lo cual se obtuvo un periodo de 52 segundos para su encendido y apagado;

Para saber cuántos datos de muestra en total se debe tomar de la señal del motor DC, se empleó la siguiente formula:

[1]

Donde:

El valor de 100 corresponde a la frecuencia de trabajo [1/s].

T = periodo total de la señal [s].

fn = frecuencia natural del semiciclo [1/s].

#Datos = Cantidad de muestras a tomar.

Reemplazando los valores se obtiene que el número de datos que debemos de tomar serán:

[2]

Esto quiere decir que serán 100 los datos para el ciclo de encendido y 100 para el ciclo de apagado.

Ya conociendo estos parámetros, en el código implementado se estableció un delay de 259. Esto significa que se obtendrá por pantalla el valor de las RPM del motor cada 0.26 segundos, se obtendrá los 100 datos al pasar los 26 segundos en cada ciclo.

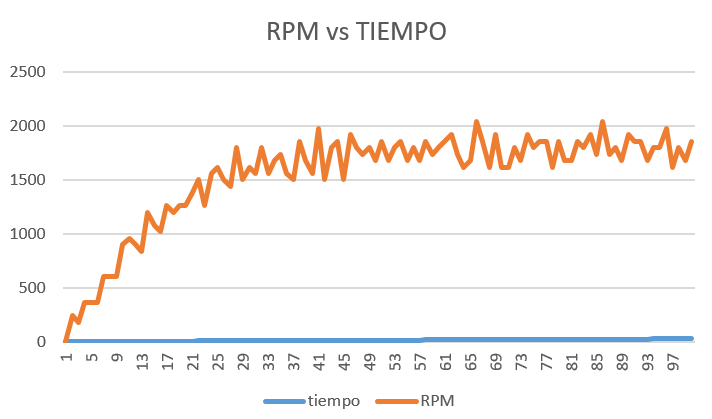


Figura 1: Curva de Datos tomados en el ciclo de encendido.

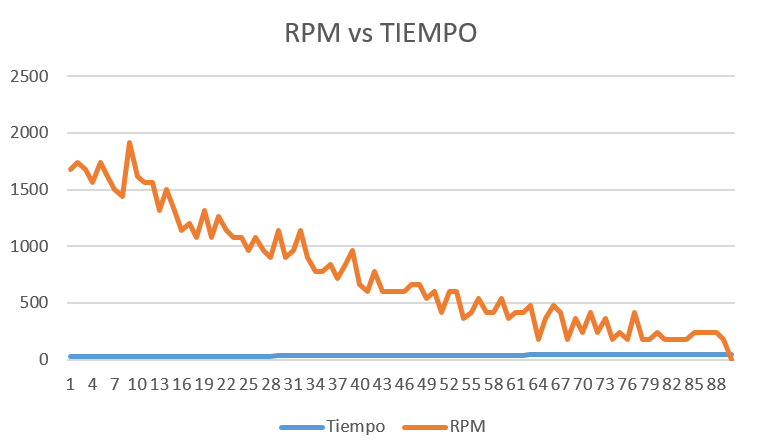


Figura 2: Curva de Datos tomados en el ciclo de apagado.

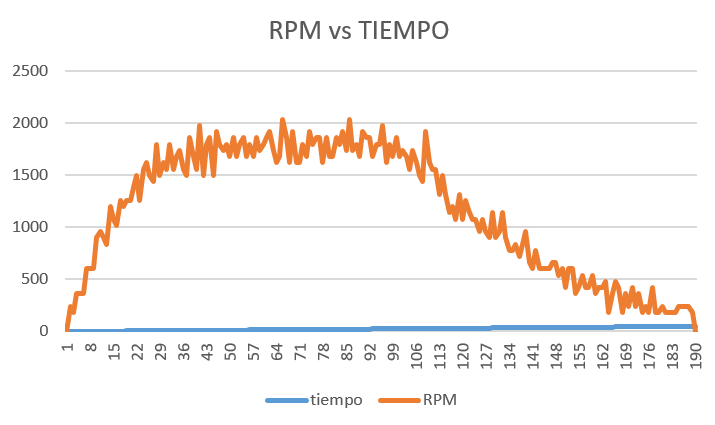


Figura 3: Curva con ambos ciclos de operación.

# Methodology

En esta sección se presenta el método realizado para el diseño de nuestra planta con los datos obtenidos durante la el proceso de adquisición, y a su vez la metodología para el diseño de nuestro controlador PID.

## System Identification

Con los datos registrados de la velocidad (r.p.m) del motor durante la etapa de encendido y apagado se las registró en Excel en forma de lista con 2 columnas, la primera columna representa el estado de operación (0-255) y la segunda columna tiene el valor de las revoluciones por minuto. En la figura 4 y 5 se puede ver la representación en histograma de dichos valores.

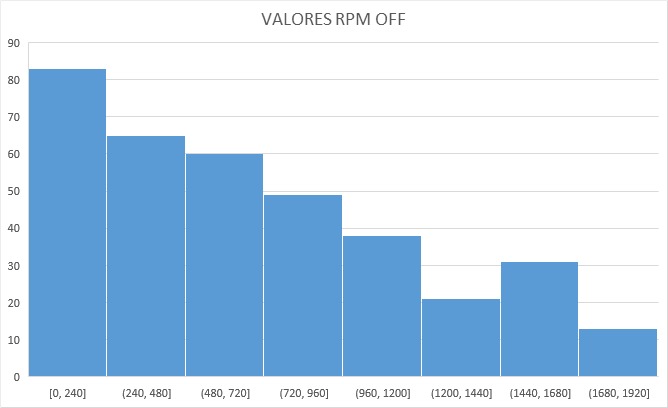


Figura 4: Datos en el estado de operación OFF.

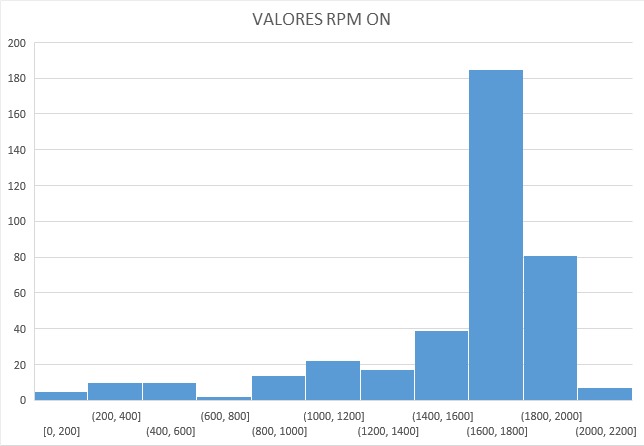


Figura 5: Datos en el estado de operación ON.

Se realiza la lectura de los datos en el programa Matlab y una vez importado se crea una variable con las 2 columnas del archivo. Se obtiene los datos de entrada “in1” que representa el estado de operación del motor y salida de la variable “out1” que representa el valor r.p.m del motor, como lo muestra en la figura 5.



Figura 6: Obtención de los datos de entrada y salida.

Con la herramienta System Identification se procede a estimar el modelo, con los datos reales obtenidos de la planta. Se tiene que seleccionar aquel modelo que presente un mayor el porcentaje de ajuste superior al 80%, como se muestra en la Figura 5.

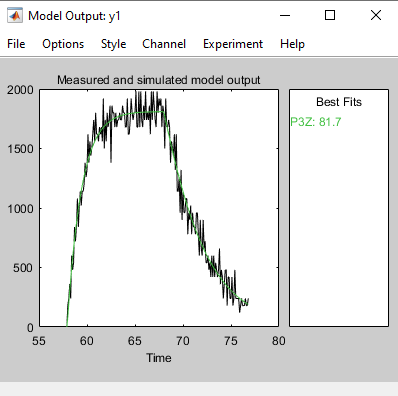


Figura 7: Porcentaje de ajuste del modelo escogido.

Con el modelo matemático escogido se logró obtener las constantes de la función de transferencia que representa al sistema de las revoluciones por minuto del motor

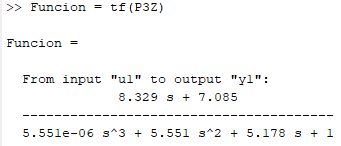


Figura 8: Función de transferencia del modelo de estimación.

Como se muestra en la Figura 7 la función de transferencia consta de grado 3, con un sus respectivos polos y ceros.

## Sisotool()

Para crear el controlador PID se utiliza la herramienta sisotool() con lo cual se obtienen los valores de las constantes

Kp, Ki y Kd del controlador, como se muestra en la Figura 8

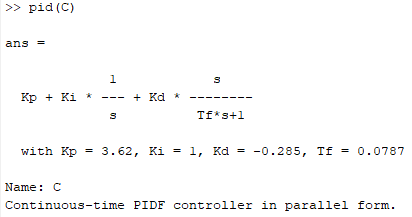


Figure 9: Parámetros del controlador PID.

# CONCLUSIONS

Cuando se tiene un sistema que adquiere datos em tiempo real es importante tener una etapa de acondicionamiento de la señal antes de analizar los datos para realizar el control del sistema, en el caso del presente trabajo se tuvo que filtrar y realizar varias pruebas de adquisición de datos durante un periodo de tiempo para poder seguir con la implementación del sistema.

La adquisición de datos es una de las etapas más importantes de todo el sistema puesto que para el sistema presente se tuvo que primero hallar la representación matemática de las revoluciones por minuto del motor a distintos ciclos de trabajo, como se pudo ver en el proyecto estos eran cuando el motor funcionaba a su voltaje máximo y cuando se dejaba de alimentar completamente el motor, es decir, un ciclo de trabajo igual a 0%.

Se obtuvo el modelo matemático de las revoluciones por minuto del motor gracias a la herramienta de Matlab, llamada System Identification, para luego poder encontrar la función de transferencia del controlador con sus respectivas constantes.

Para poder transferir los datos que se adquirían desde Arduino se planteó utilizar comunicación UART, ya que implementar una comunicación I2C requería más tiempo de implementación.

En el presente proyecto se utilizaron múltiples procesadores, para realizar distintas tareas en uhna misma tarjeta, aumentando así la eficiencia y el uso de recursos compartidos. El primer procesador encargado de realizar la lectura de datos en tiempo real provenientes de la planta, el segundo procesador era el controlador PID hallado inicialmente en la etapa de identificación del sistema y por último un tercer procesador el cual estaba encargado de la visualización de los datos.

Se instalo un servidor de pagina web en el procesador Dual-core ARM Cortex-A9, en el sistema operativo Linux, en el cual se visualiza los datos en tiempo real que son adquiridos desde la planta, es decir, se muestran el funcionamiento de motor en sus dos ciclos de trabajo, funcionando al 100% y al 0%

# Referencias

|  |  |
| --- | --- |
| [1] | A. Jaya, M. Badriatul, G. Prabowo, E. Purwanto, F. Dwi y M. Rizani, «Design of PID-Fuzzy for Speed Control of Brushless DC Motor in Dynamic Electric Vehicle to Improve Steady-State Performance,» *International Electronics Symposium on Engineering Technology and Applications,* pp. 179-184, 2017. |
| [2] | A. Mohamed, M. Mahmoud, L. Said y A. Radwan, «Design of FOPID Controller for a DC Motor using Approximation Techniques,» de *Novel Intelligent and Leading Emerging Sciences Conference*, Cairo, 2019. |
| [3] | R. Shanmugasundram, K. Zakariah y N. Yadaiah, «Implementation and Performance Analysis of Digital Controllers for Brushless DC Motor Drives,» *IEEE/ASME Transactions on Mechatronics,* vol. 19, nº 1, pp. 213-224, 2014. |
| [4] | T. Vamsee kiran y I. Sowjanya, «Speed Performance of a BLDC Motor Employing PWM/PAM Control Techniques,» *International Journal of Application or Innovation in Engineering & Management,* vol. 5, nº 5, pp. 29-38, 2016. |
| [5] | G. Yoshihiko, «DC-motor speed control by PIC-based digital PID-controller,» 2011. |
| [6] | A. Telba, «Motor Speed Control Using FPGA,» de *World Congress on Engineering*, London, 2014. |
| [7] | A. Fakhrulddin, M. Mohammed y I. Sinan, «LabVIEW FPGA Implementation Of a PID Controller For D.C. Motor Speed Control,» de *International Conference on Energy, Power and Control (EPC-IQ)*, Basrah, 2010. |
| [8] | P. Sandeepa y K. K. Praveen, «PID Implementation on FPGA for Motion Control in DC Motor Using VHDL,» *IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) ,* vol. 6, nº 3, pp. 116-121, 2016. |
| [9] | B. Joni y A. Anwarul, «Design and Implementation of a FPGA Based Closed Loop Speed Controller for DC Motor using PWM Technique,» de *International Conference on Advancement in Electrical and Electronic Engineering*, Gazipur, 2018. |
| [10] | I. A. G. Pacheco, E. G. Ramírez, E. O. G. Ramírez y C. L. P. Agüero, «DISEÑO DE EXPERIMENTOS CON FPGA'S EN CURSOS DE INGENIERÍA RELACIONADOS CON MOTORES DE CORRIENTE CONTINUA,» *Dyna,* vol. 90, nº 4, pp. 372-379, 2015. |