



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Manual de prácticas para motores de corriente directa

MATERIAL DIDÁCTICO

Que para obtener el título de

INGENIERO MECATRÓNICO

P R E S E N T A

Francisco Michell Salinas González

ASESOR

Dr. Edmundo Gabriel Rocha Cózatl



Ciudad Universitaria, Cd. Mx., 2018

Índice

1	Objetivos.....	1
2	Resumen	1
3	Introducción.....	2
3.1	Identificar las partes del banco de pruebas.	3
3.2	Arquitectura abierta	7
3.3	Conexión Bluetooth	10
3.4	Estados del microcontrolador.....	11
4	Interfaces de usuario.....	12
4.1	Instalación	12
4.2	Descripción de las interfaces	14
4.2.1	Interfaz_1	14
4.2.2	Interfaz_2.....	17
4.2.3	Errores comunes de conexión.....	19
5	Descripción de las prácticas.	20
5.1	Codificador (encoder) de cuadratura	20
5.2	Objetivos de Control (Estabilización).....	20
5.3	Objetivos de Control (Regulación)	20
5.4	Modelado Matemático	20
5.5	Diseño de Controladores con Base en la Función de Transferencia.....	20
5.6	Comparación de Controladores	20
5.7	Sintonización de Controladores PID, Método Ziegler- Nichols (Oscilaciones continuas)	21
5.8	Relación de prácticas con diversas materias.....	21
6	Prácticas.....	22
6.1	“Codificador (<i>encoder</i>) de Cuadratura”	23
6.2	“Objetivos de Control (Estabilización)”	27
6.3	“Objetivos de Control (Regulación)”.....	30
6.4	“Modelado Matemático”.....	34
6.5	“Diseño de Controladores con Base en la Función de Transferencia”	44
6.6	“Comparación de Controladores”	55
6.7	“Sintonización de Controladores PID, Método Ziegler-Nichols (Oscilaciones Continuas)”	59
7	Conclusiones.....	64

8	Fuentes de consulta.....	67
9	Anexos Generales.....	69
9.1	Códigos de programación.....	69
9.1.1	Variables.....	69
9.1.2	Configuraciones.....	71
9.1.3	Programa principal.....	72
9.1.4	Funciones.....	77
9.2	Código de programación de las interfaces.....	82
9.3	Especificaciones de material eléctrico o electrónico.....	85
9.3.1	Metal Gearmotor 37Dx57L mm con Encoder 64 CPR.....	85
9.3.2	Micro Metal Gearmotor 100:1.....	86
9.3.3	ACS714 Current Sensor Carrier -30A to +30A.....	86
9.3.4	VNH5019 Motor Driver Carrier.....	86
9.3.5	Bluetooth HC-06.....	87
9.3.6	Tiva C Series TM4C123G. EK-TM4C123GXL.....	87
9.4	Especificación de tarjeta PCB del banco de pruebas grande.....	88

1 Objetivos

- Elaborar prácticas enfocadas al control y empleo de las señales de motores de corriente directa, como material de apoyo donde el usuario aplicará conocimientos adquiridos en clase.
- Crear interfaz de usuario que permitirá la adquisición de información gráfica que se aplicará y analizará para comprobar y reforzar conocimiento adquirido en clases.

2 Resumen

El banco de pruebas elaborado por *Salgado, O. Q. (2015)*. Se componen de diversas partes como son chumaceras, ejes y elementos electrónicos como el decodificador, el sensor de corriente entre otros. Además cuenta con un sistema llamado arquitectura abierta que le brinda la posibilidad de interactuar con otros microcontroladores, sin depender del instalado internamente. Para comunicarse con la computadora emplea conexión alámbrica (Cable DB15); sin embargo también está la opción para comunicarse con conexión inalámbrica (*bluetooth*), dando versatilidad de poder trabajar alejados del sistema; a una distancia no mayor de 10 metros.

Como forma de interacción con el usuario, la programación hace uso del led indicador del microcontrolador para mostrar estados o fases que tome el proceso de alguna práctica, entre los colores que puede tomar están el azul, blanco, rojo y verde.

Se tiene una interfaz de usuario que se puede instalar en cualquier computadora (que cumpla con los requisitos de instalación) mediante un archivo ejecutable (.exe). Se puede analizar la respuesta del motor ante una determinada entrada, es decir, se observan los datos obtenidos en una gráfica, que se puede guardar como imagen, además de que los datos se almacenan en un archivo para ser utilizados en otra ocasión por el mismo banco de pruebas.

El manual contempla se contempla para abarcar distintas asignaturas de forma directa o indirecta, entre las materias directas esta Control Automático y Modelado de Sistemas, mientras que las indirectas están Diseño Mecatrónico e Instrumentación.

Las prácticas elaboradas cubren aspectos como el uso del decodificador (encoder), Objetivos de control (Estabilización y Regulación), Diseño de Controladores, Comparación de Controladores. Donde se emplea la Función de Transferencia del motor, mientras que la práctica de Modelado Matemático se obtiene la FT, al caracterizar el motor al conocer su velocidad y el tiempo de la constante Tau.

3 Introducción

Durante muchos años los temas relacionados a motores de corriente directa, en clase se han transmitido de forma teórica y de forma empírica, como si el conocimiento se pasase de generación en generación de forma oral, lo cual no es malo, sin embargo la comprensión de ciertos temas o conceptos se asimilan mejor cuando estos se realizan o ejecutan de forma experimental.

El presente manual de prácticas se realiza con base en el banco de Pruebas grande (Figura 3-1), elaborado en el trabajo de Salgado, O. Q. (2015).

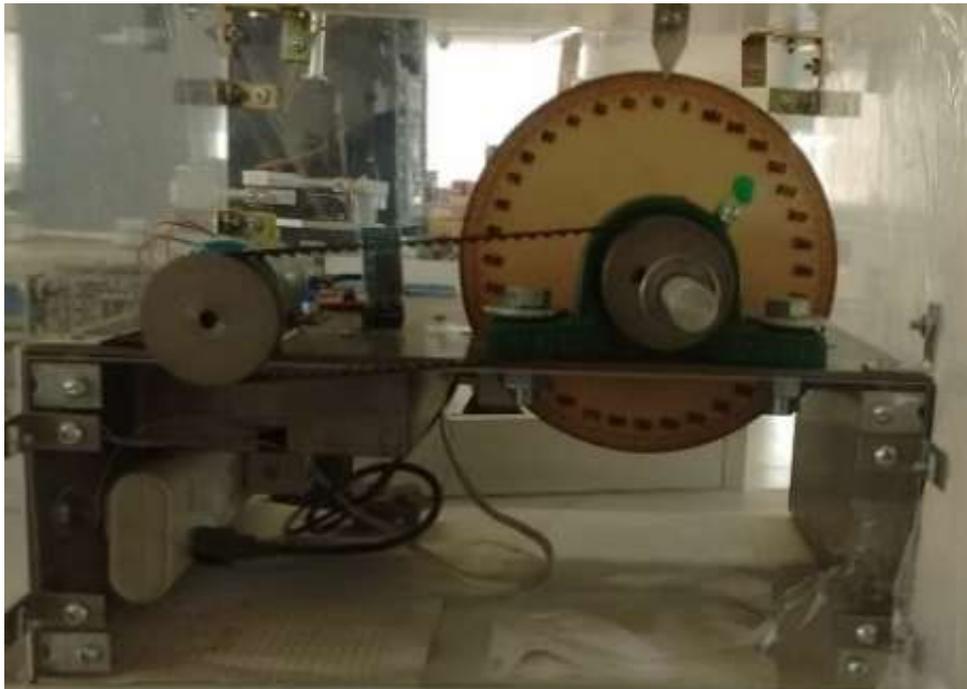


Figura 3-1 Banco de prueba Grande

Dicho banco de pruebas está destinado para realizar prácticas relacionadas con distintas asignaturas tales como Control Automático, Instrumentación, Diseño Mecatrónico, Modelado de Sistemas, entre otras asignaturas de la carrera de Ingeniería Mecatrónica o afines. En el presente trabajo, el alcance de las prácticas elaboradas es para la asignatura de control automático, pero no significa que descarten por completo su uso en alguna otra asignatura.

Además, se tendrá acceso a las señales fundamentales de los dispositivos (arquitectura abierta) que conforman el control habitual de un motor de corriente directa, como las entregadas por el encoder, el sensor de corriente, el módulo bluetooth y el driver (puente H) para el motor.

Los códigos de programación además de funcionar con el microcontrolador (uC) del banco de pruebas, se pueden usar en la arquitectura abierta con un uC similar.

En el caso de querer realizar las prácticas en su computadora, será necesario instalar la interfaz de la que se necesite, además de instalar la **IDE Energia** para poder cargar los códigos de las prácticas a la tarjeta **Tiva C Series TM4C123G. EK-TM4C123GXL**.

También se podrá ver la programación de la interfaz para usarla como base para crear su propia interfaz, en dicho caso, se recomienda tener instalado **Ni LabVIEW 2017** o una versión superior con la paquetería **VISA** (para la comunicación Serial). La carpeta que contiene las prácticas de los bancos de pruebas, se puede solicitar al profesor con el que se realice el laboratorio. Los códigos de programación para el microcontrolador, las aplicaciones, instaladores y complementos correspondientes de cada interfaz, vendrán separadas por carpetas.

Todos los programas que se empleen en el presente manual de prácticas, como LabVIEW y Matlab, de no contar con licencia oficial, se recomienda usar la versión de prueba o de estudiantes.

El uso del banco de pruebas y el presente manual son opcionales, son un material que el profesor puede emplear como apoyo a su respectiva clase, el orden de aplicación de las prácticas es una sugerencia; por lo que el profesor tiene la libertad de realizar una o más prácticas del manual y en el orden que crea necesario o conveniente.

Por ejemplo la asignatura de control automático no tiene laboratorio, entonces el manual puede servir como una opción inicial para la creación de un laboratorio, pero como se ha mencionado pueden surgir más y diferentes prácticas referentes a las asignaturas mencionadas, obteniendo un compendio o conjunto de prácticas aplicables al motor del banco de pruebas.

El tamaño de brigadas para trabajar se sugiere que sea de dos personas, pero el queda a criterio el profesor.

Para trabajar con el banco de pruebas es en el departamento de mecatrónica ubicado en el edificio CIA. Si se desea utilizar en otra ubicación, está a disposición del profesor encargado.

3.1 Identificar las partes del banco de pruebas.

Los componentes principales que conforman al banco de pruebas son:

- | | |
|----------------------|------------------------|
| 1. Motor | 5. Sensor de corriente |
| 2. Chumaceras | 6. Poleas dentadas |
| 3. Microcontrolador | 7. Banda dentada |
| 4. Driver (Puente H) | 8. Modulo bluetooth |

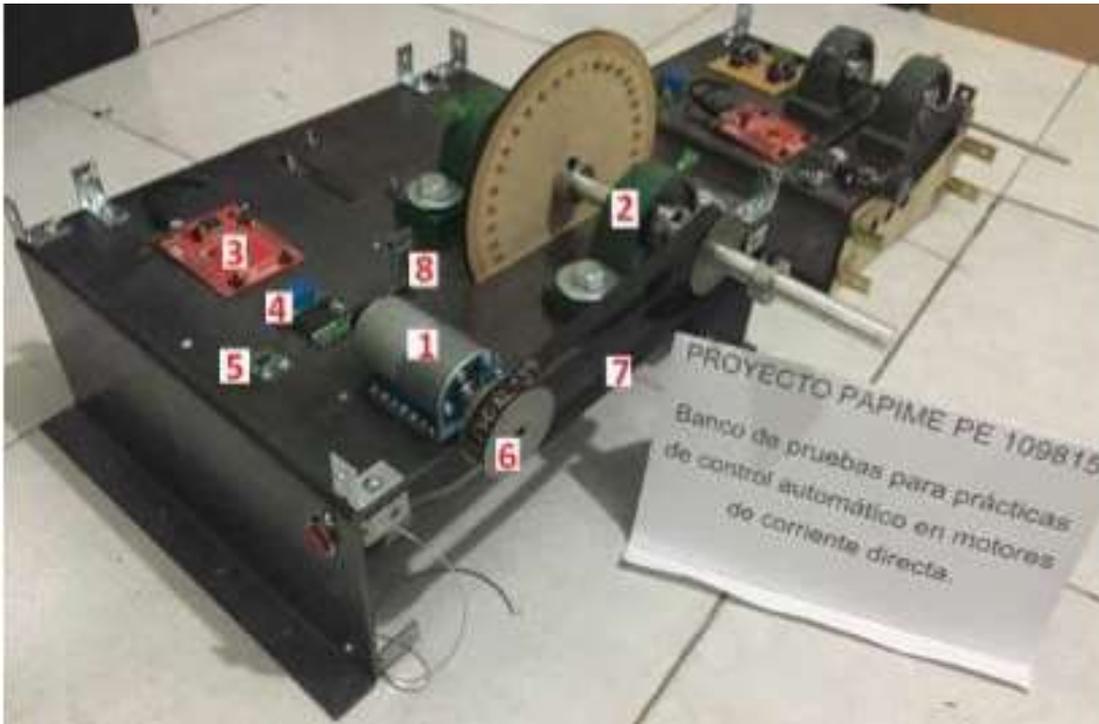


Figura 3-2 Elementos del banco de pruebas. Tomado de Salgado, O. Q. (2015), PP. 37.

El cable de alimentación para suministrar energía a la fuente de poder del banco de pruebas, proveniente del multicontacto con botón de encendido/apagado “switch” (**¡Error! No se encuentra el origen de la referencia.**) ubicado en la parte trasera del dispositivo, el led indicador deberá encenderse, de lo contrario revise que la toma de corriente proporcione voltaje o consulte con el profesor asignado. En dicho conector se conecta la fuente de alimentación del banco de pruebas y el cable USB para dar energía al uC. Al presionar el “switch”, algunos elementos electrónicos del banco de pruebas deberán encender sus leds indicadores (uC y módulo bluetooth).



Figura 3-3 Multicontacto con Switch de led indicador.

Tiene la capacidad de cambiar la fuente interna por una externa de voltaje variable, con la ayuda de conectores tipo banana, según la práctica a realizar.



Figura 3-4 Conectores banana, Rojo (positivo) y Negro (negativo).



Figura 3-5 Botones: Azul para posicionar el motor. Rojo para reinicio del microcontrolador.

Cuenta con una flecha (*Figura 3-6*) y un disco grabado en grados. La función de esta flecha es indicar visualmente la posición en grados del motor, sin embargo su uso se puede omitir, dado que el programa del uC al presionar el botón rojo del banco de pruebas, establece la posición en cero. Así mismo se dispone de discos de carga para agregar perturbaciones en los sistemas de control de

alguna práctica de este manual o las que deseen desarrollar los usuarios. Para agregar o quitar los discos de peso, se tiene una llave allen, que se ubica en un orificio a un costado del motor.



Figura 3-6 Flecha con indicador en grados.



Figura 3-7 Flecha con discos de carga.

Las señales provenientes del motor se interconectan y direccionan al microcontrolador; y de éste también las señales de control se dirigen hacia al motor a través de la tarjeta PCB, manteniendo la

organización y el orden en las conexiones.

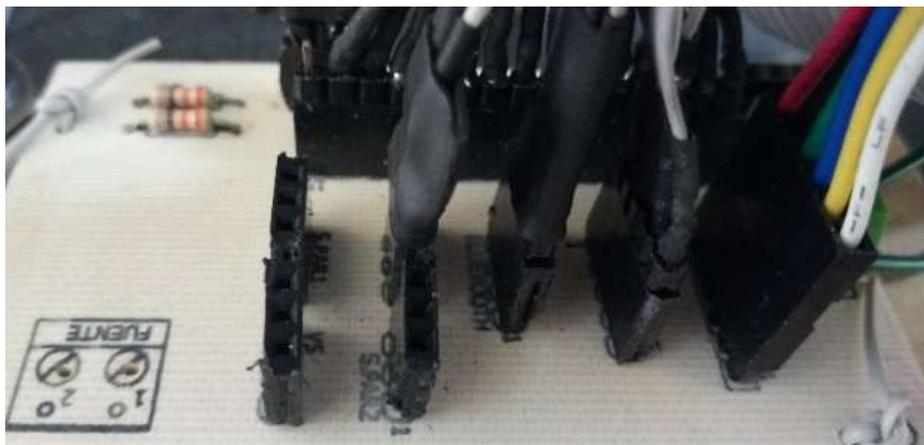


Figura 3-8 Tarjeta PCB y conexiones.

Finalmente la conexión de los conectores DB15 es de gran utilidad para desconectarse del microcontrolador interno y hacer uso de la arquitectura abierta del sistema.

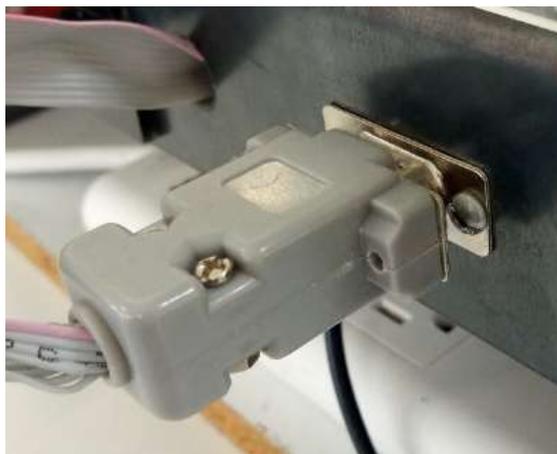


Figura 3-9 Conexión de cable DB15.

3.2 Arquitectura abierta

Es el sistema que permite conectar y desconectar microcontroladores según sea necesario, porque permite disponer de las señales (de encoder, sensores de corriente, comunicación bluetooth, driver, etc.) fundamentales que permiten el control de posición, velocidad y par de un motor de corriente directa.

En la parte trasera se ubica un conector DB15. En la Figura 3-10 se muestran las señales

correspondientes a cada pin para el conector DB15.

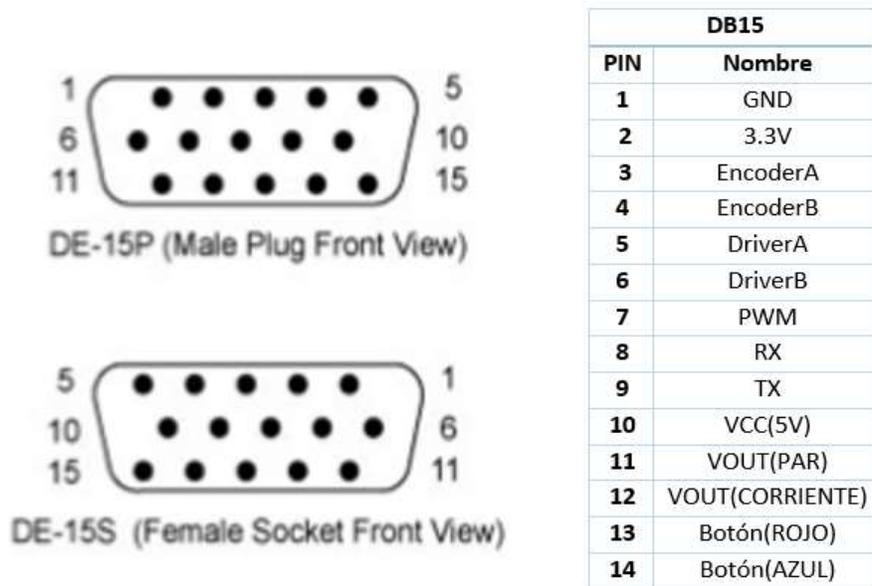


Figura 3-10 Señales correspondientes a cada PIN del conector DB15 para su versión macho y hembra.



Figura 3-11 Cable DB15

Para la realización de las prácticas solo se contará con un único cable (o la disposición por cables donados), por lo que el usuario deberá construir su propio cable en cuyo extremo se encuentre el

DB15 (hembra o macho dependiendo del banco de pruebas) y en el otro extremo se encuentren las terminales que se conectarán de manera directa al microcontrolador (ya sea Texas Instruments, Atmega, etc.).

La terminal 15 no conduce señal alguna.

Los 14 pines mostrados en la Figura 3-10 deberán conectarse al dispositivo de control. Las señales de entrada al microcontrolador, tienen amplitud promedio de 3.3 V, ideal para los uC de bajo consumo, no obstante, pueden ser leídas correctamente por uC que manejan señales de 5V de amplitud promedio.

Tabla 3-1 Señales de pines

Pines	Señales correspondientes
3 y 4	Señales cuadradas con desfase de 90° correspondientes al encoder acoplado al motor reductor.
5 y 6	Para el sentido de giro del motor
7	Señal de control (PWM) que envía el uC hacia el driver de potencia.
8 y 9	Comunicación bluetooth.
11	Señal libre, se puede modificar la Tarjeta para ser Voltaje, GND u otra señal.
12	Señal analógica del sensor de corriente

Se sugiere la construcción de suficientes cables DB15 (con cable plano) según se necesiten por grupo de clase. El cable plano puede ser sensible (fatiga) a dobleces repetitivos o extremos, como todos los elementos electrónicos, debe ser tratado con cuidado. Los cables elaborados se pueden donar para que se usen posteriormente en el banco de pruebas, con el fin de reducir la cantidad de materiales a comprar o los desechos que estos puedan ocasionar.

Implementación de algoritmos de control elaborados por los usuarios.

Con un controlador externo y el cable DB15 construido, está todo listo para implementar algoritmos (independientes a los de este manual) para controlar posición, velocidad y par en los motores. Para ello desconectar el cable DB15 del microcontrolador incluido en el banco de pruebas, al cual solo se le cargaran los algoritmos de las prácticas del manual que le permite controlar al motor y a su vez comunicarse con la interfaz instalada. Se sugiere que el usuario cree su propia interfaz, o de alguna otra forma muestre en tiempo real los valores medidos.

Sugerencias:

Cualquier tarjeta que se desee conectar al banco de pruebas deberá primero pasar por el visto bueno

del profesor encargado. Evitar dañar la tarjeta y/o el (los) banco(s) de prueba. Antes de conectar un nuevo uC, revisar el cable construido (probar continuidad con multímetro) para evitar cualquier corto o desconexión que puedan derivar en daños permanentes al uC y a los componentes de los bancos de pruebas. Si hay dudas, comentarlas con el profesor en turno. Si el banco de pruebas no funciona con el algoritmo de control del usuario, desconectar el cable db15 del usuario y conectar el que viene con el sistema, realizar una prueba y si funciona con normalidad, entonces el uC externo y/o el algoritmo del usuario posiblemente contengan fallas. Intentar primero con algoritmos de control sencillos.

3.3 Conexión Bluetooth

El dispositivo cuenta con conexión remota *bluetooth*, en el caso de no querer usar esta conexión, solo tiene que desconectar el Bluetooth HC-06 del banco de pruebas y conectar el cable USB a la computadora.

Para vincular o emparejar la computadora con el banco seguir las instrucciones de la página de Microsoft <<https://support.microsoft.com/es-mx/help/15290/windows-connect-bluetooth-device>>, en dicha página se muestran las formas de emparejar cualquier dispositivo *bluetooth* a una PC con Windows 7, 9.1 y 10. Para poder terminar la conexión; el sistema pide una contraseña para acceder/conectarse al *bluetooth* del banco de pruebas, la clave de acceso es **1234**.

El nombre del dispositivo es **BancoPruebas**, éste puede no aparecer cuando el emparejamiento no se haya realizado.

Al encender el banco de pruebas, el led indicador del *bluetooth*, estará parpadeando, lo que significa que ésta esperando para ser conectado. Una vez conectado el led dejará de parpadear.

Para conocer el Puerto COM, seguir los siguientes pasos.

Ir a **Panel de control** después a **Dispositivos e impresoras** donde se puede observar la etiqueta de **BancoPruebas**, dar clic secundario del *mouse* para acceder a las propiedades del dispositivo.

Posteriormente ir a la pestaña **Servicios** para observar el puerto COM asignado, recordar tener el *bluetooth* de la computadora encendido, en caso contrario no se mostraran las propiedades del **BancoPuebas**.



Figura 3-12 Propiedades: BancoPruebas (Puerto COM).

3.4 Estados del microcontrolador.

Para saber el estado en que se encuentra el microcontrolador se utiliza el *led RGB* para separar o identificar cada estado mediante colores, o secuencias de encendido y apagado.

Al cargar un nuevo programa o al hacer *reset* en el uC el *led* presenta un parpadeo con luz roja.

El estado de espera o de paro, el *led* se iluminará de color verde.

Cuando se envían los comandos de operación el *led* prenderá de color rojo mientras lee y procesa el comando recibido, es decir, no toma más de un segundo.

Si el comando leído fue para iniciar la secuencia del PID o para enviar datos a la computadora el color que se encenderá es el azul y en algunos casos (Control de velocidad) luz blanca, este estado tiene una duración aproximadamente de 10 a 20 segundos, según este establecido en los códigos de programación.

Tabla 3-2 Estados del uC.

 	Secuencia en ejecución.
	Secuencia detenida.
	El uC se reinició (parpadea).

4 Interfaces de usuario

Las siguientes interfaces han sido desarrolladas para la realización de algunos experimentos de este manual de prácticas empleando el banco de pruebas, Figura 3-1; experimentos en los cuales se procesan señales tales como: posición y velocidad.

Su objetivo es auxiliar al profesor y al alumno en la definición de parámetros del sistema a utilizar para realizar las prácticas y apreciar visualmente lo que se estudia en clase, y así poder analizar los resultados de una forma más sencilla.

Proporcionando un entorno visual sencillo para permitir la comunicación entre el banco de pruebas y la computadora, a través de un microcontrolador.

Las interfaces se desarrollaron en LabVIEW debido al conocimiento y experiencia del programador en este software para crear interfaces de usuario, pero podrían ser desarrolladas en otro tipo de software.

4.1 Instalación

- I. Si la computadora en la que desea utilizar las interfaces está instalada una versión de LabVIEW 2017 o superior y que el sistema operativo sea Windows 7 o superior, sólo bastará con entrar a la carpeta Aplicación (correspondiente a la interfaz deseada) y abrir el archivo ejecutable (.exe).
- II. En caso de no contar con el software de LabVIEW, ir a la carpeta **Instalador** de la práctica que desea realizar. Y seguir los siguientes pasos.
 1. Dar doble clic al archivo setup.exe o ejecutar como administrador, aceptar la petición del anuncio emergente del sistema emergente. Aparecerá el instalador del programa, dar clic en *Next*, aceptar términos y condiciones (Figura 4-1), dar clic en *Next*
 2. Una vez terminado el proceso de la **¡Error! No se encuentra el origen de la referencia.**, dar clic en *Next*, y finalmente dar clic en *Finish*.
 3. Reiniciar la computadora para terminar la instalación.
 4. Ir a **Archivos de Programa** del disco local C, abrir la carpeta con el nombre de la interfaz instalada, dar clic derecho para agregar un acceso directo en el escritorio.
 5. La interfaz esta lista para ser utilizada.

Los iconos de las interfaces se muestran en las **¡Error! No se encuentra el origen de la**

referencia.¡Error! No se encuentra el origen de la referencia..

Nota: Una vez instalada una interfaz, la instalación de la siguiente interfaz será más rápida y sencilla.

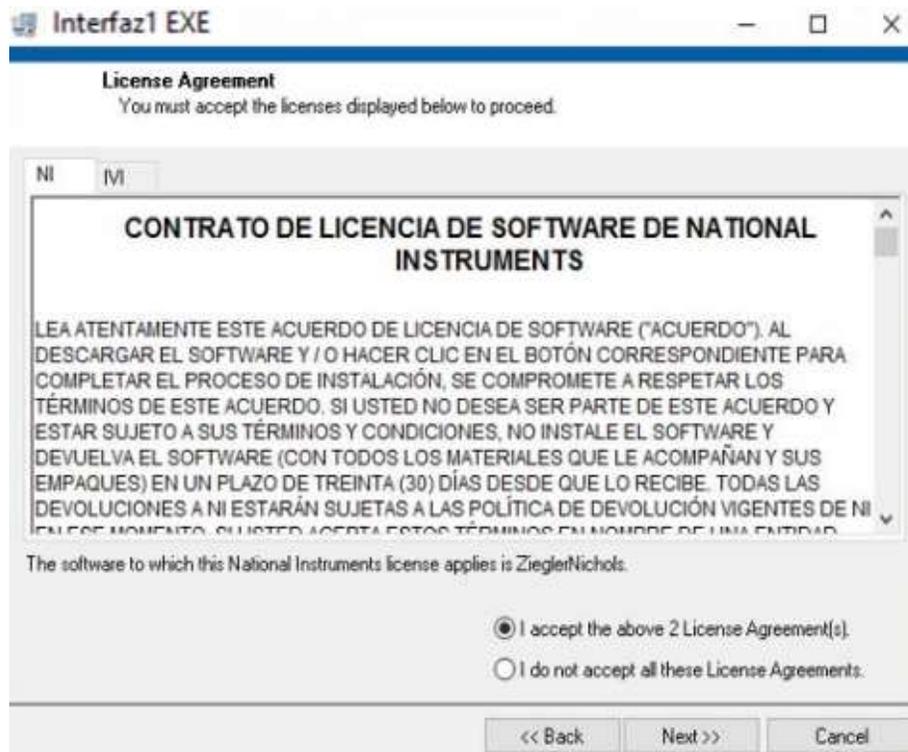


Figura 4-1

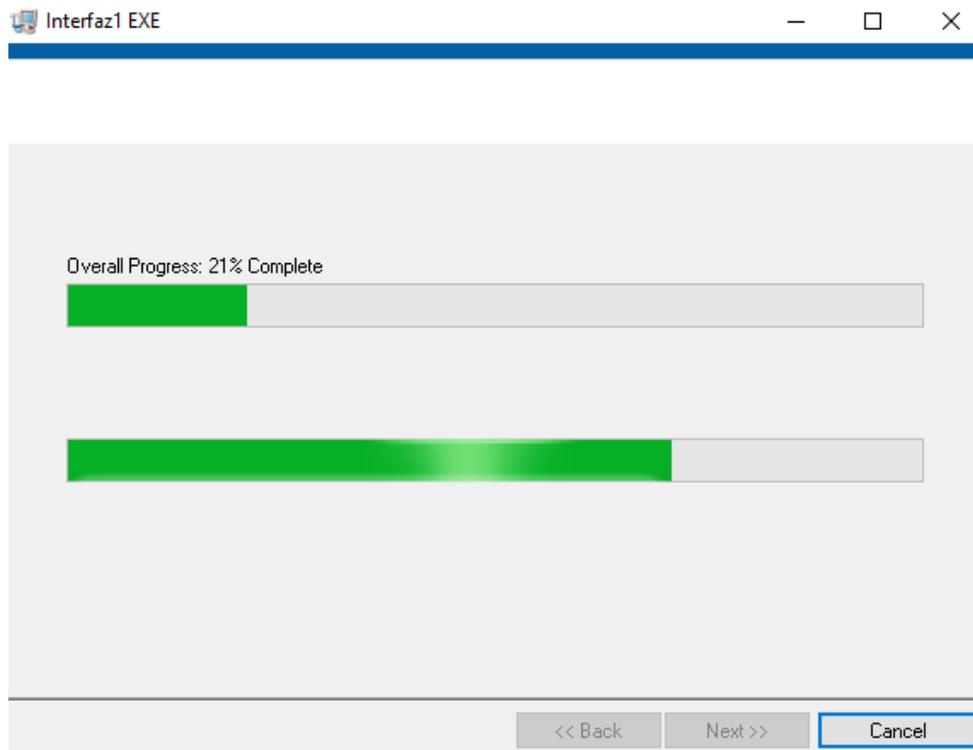


Figura 4-2



Figura 4-3 Interfaz_1.exe



Figura 4-4 Interfaz_2.exe

4.2 Descripción de las interfaces

4.2.1 Interfaz_1

Para comunicarse con esta interfaz de usuario (IU) se emplea una velocidad de 115200 *baudios*.

Los elementos que la componen son:

1. Botón RUN para activar la interfaz.
2. Cuadros de texto para ingresar una dirección para crear la carpeta donde se guardarán los experimentos realizados en la práctica. El nombre del o los archivos que se generaran. Y la selección del puerto COM correspondiente al microcontrolador a emplear.
3. Botones para elegir las secciones de Configuraciones (Figura 4-5), Prueba (Figura 4-6) y Graficar (Figura 4-7).
4. Botones para elección de la práctica a realizar (Activos cuando la interfaz está en la sección de Prueba): Estabilización, Regulación, Comparación de controladores, Diseño de controladores y Ziegler-Nichols.
5. Botón *Power* que sirve para finalizar la interfaz.



Figura 4-5 Interfaz_1 Configuraciones

6. Cuadros de texto que muestran las cadenas de caracteres que envía y recibe del microcontrolador.
7. Cuadros en los que se introducen los valores de las constantes del PID, en el apartado de Objetivo Ctrl. y Comparación Ctrl. los cuadros mostrarán los valores de las constantes predefinidas.
8. Perilla con cuadro de entrada para seleccionar el valor deseado del sistema.
9. Botones de INICIO y ALTO, para los experimentos.
10. Botón que limpia las pantallas de los gráficos desplegados.
11. Caja de botones, los botones cambian acorde a la práctica seleccionada.
12. Gráficas para el valor Y_m y el U (salida de la ley de control)



Figura 4-6 Interfaz_1 Prueba

- 13. Cuadro de texto para ingresar o buscar la dirección del archivo (.csv) para ser graficado.
- 14. Lugar donde se desplegara la gráfica de los datos obtenidos del archivo seleccionado.
- 15. Botones que permiten cambiar las etiquetas de la gráfica si los datos elegidos son para Posición o Velocidad.
- 16. Botón que permite importar la gráfica como imagen
- 17. Mando que habilita o deshabilita el mallado de la gráfica en la imagen que será exportada.

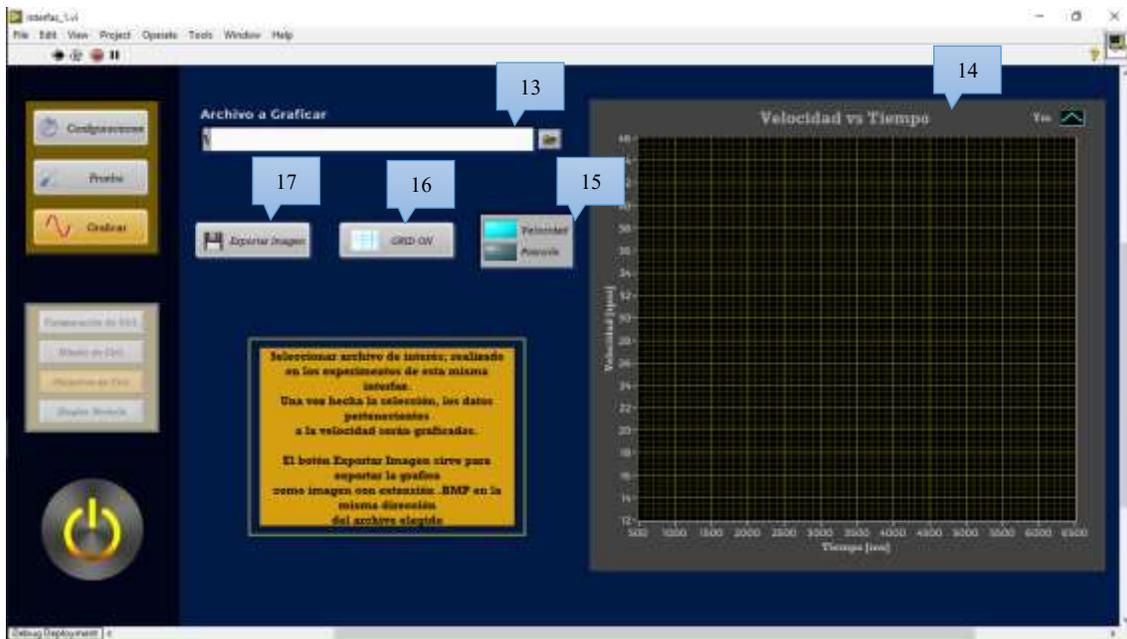


Figura 4-7 Interfaz_1 Graficar

4.2.2 Interfaz_2

Esta segunda IU cuenta con pequeñas diferencias respecto a la interfaz_1. Por lo que no es necesario enumerar las características que contiene esta misma, excepto que la velocidad que utiliza es de 9600 baudios.

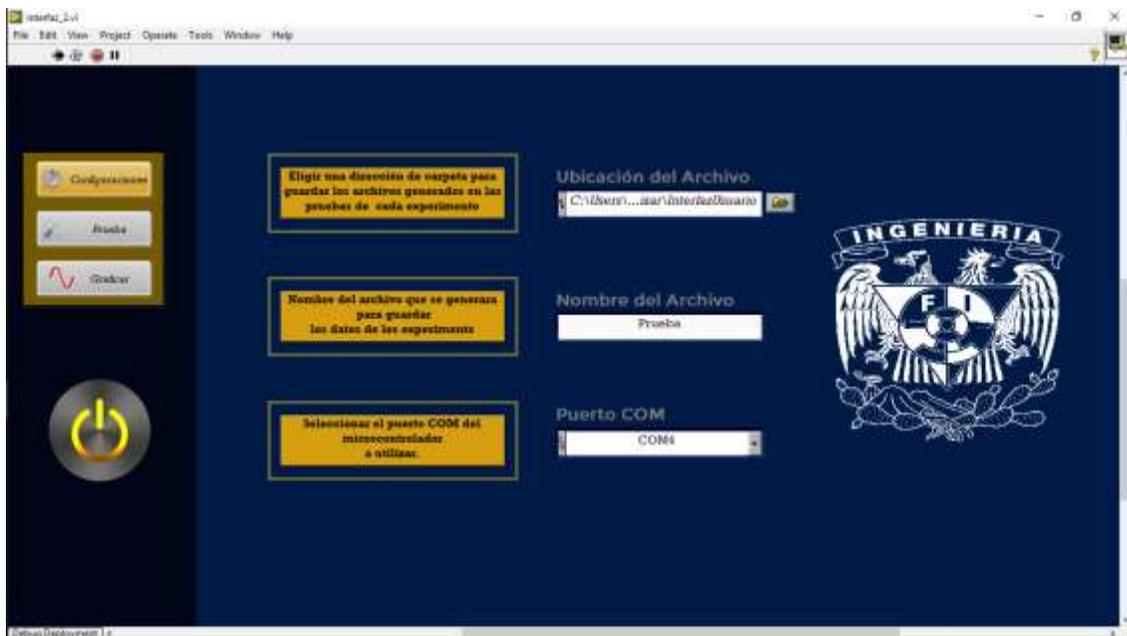


Figura 4-8 Interfaz_2 Configuraciones



Figura 4-9 Interfaz_2 Prueba

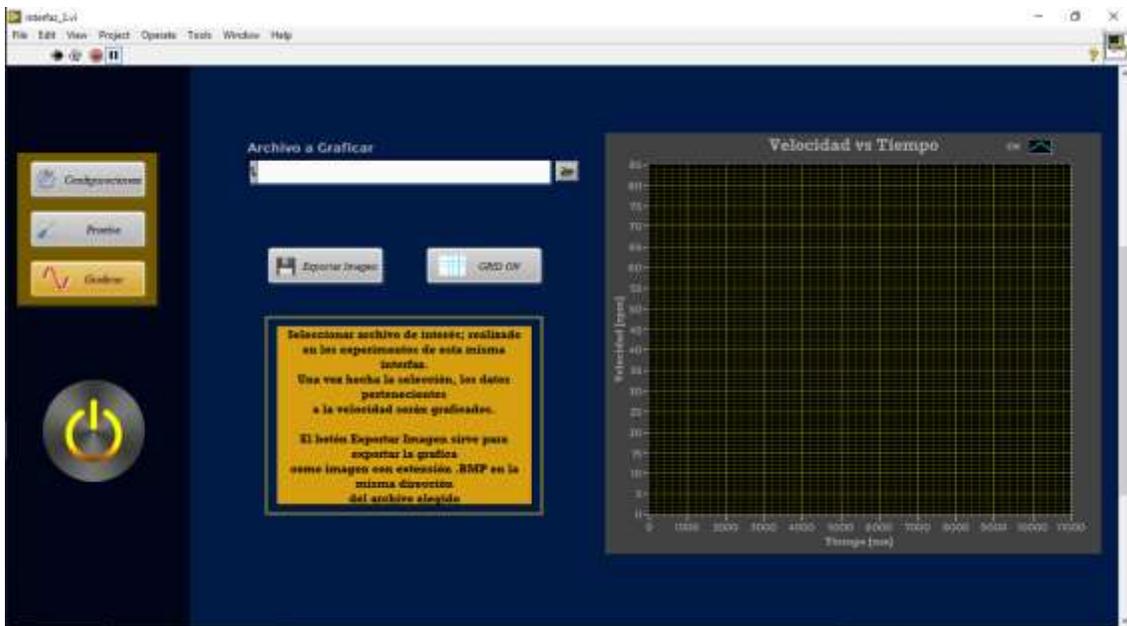


Figura 4-10 Interfaz_2 Graficar

Las gráficas que se encuentran en la sección de prueba de ambas interfaces tienen una barra de herramientas (Figura 4-11) para interactuar con la gráfica.



Figura 4-11 Herramientas de gráficos. (Graph Palette)

El botón de la izquierda sirve como puntero, el de en medio despliega un submenú para realizar distintos tipos de Zoom (Figura 4-12), y el último botón sirve para mover la gráfica a través de la pantalla.

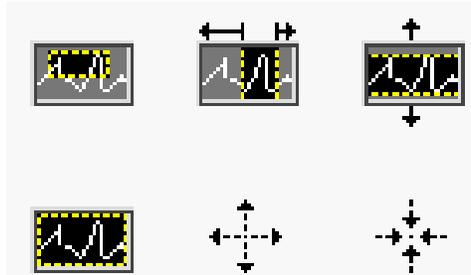


Figura 4-12 Submenú de Graph Palette

4.2.3 Errores comunes de conexión.

En el uso de la interfaz pueden suscitar algunos errores, entre los más comunes es debido a una mala conexión con la tarjeta de control **¡Error! No se encuentra el origen de la referencia.:**

- Puerto COM equivocado.
- No está conectada la tarjeta.
- Iniciar la interfaz sin haber seleccionado el puerto COM.
- El dispositivo se está comunicando con otro software.

Para solucionar estos problemas solo es necesario verificar la correcta localidad del dispositivo, cerrar todos los softwares que pudieran tener una comunicación activa con la tarjeta de control. El emparejamiento de la computadora y el uC existe pero con dificultades que impiden usar la IU correctamente. Para resolver este inconveniente solo basta con detener la interfaz y reiniciar el uC.

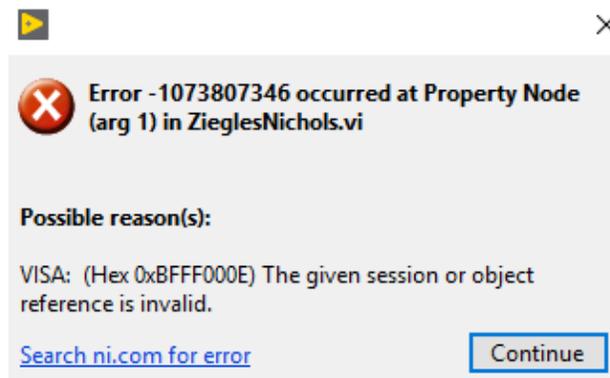


Figura 4-13 Mensaje de error en la IU.

5 Descripción de las prácticas.

5.1 Codificador (encoder) de cuadratura

Se muestra cual es el funcionamiento del desfase de 90 grados que tiene el encoder de cuadratura. La relación entre la frecuencia de los pulsos y la cantidad de pulsos por revolución, para así obtener la velocidad del eje de un motor, sin olvidar el hecho de que el motor puede contar con una caja de reducción de engranes, donde la amplitud de la velocidad cambia.

5.2 Objetivos de Control (Estabilización)

Identificar y comprender el primer concepto de objetivos de control (estabilización), mediante la ayuda del banco de pruebas. (Práctica demostrativa)

Con la capacidad de mover manualmente el eje de del motor para llevarlo a una condición inicial diferente de cero (0), para que posteriormente el sistema con control (PID), alcance el valor cero.

5.3 Objetivos de Control (Regulación)

Demostrar y comprender el segundo concepto de objetivos de control (regulación). Moviendo el eje del motor manualmente a una condición inicial arbitraria o mantener la posición en cero. Donde el sistema con control (PID) debe alcanzar un valor deseado constante. (Práctica demostrativa)

5.4 Modelado Matemático

Verificar el proceso del modelado de un motor DC, y mediante pruebas con el banco de pruebas comprobar que el resultado se acerca al realizado en el modelado.

Los experimentos son una pequeña introducción a la caracterización de un motor DC.

5.5 Diseño de Controladores con Base en la Función de Transferencia

Conocer la Función de Transferencia del banco de pruebas, y aplicar el proceso para encontrar los parámetros para un controlador tipo P, I y PI. Según un desempeño deseado. Así también observar el comportamiento del sistema dependiendo la elección de los parámetros.

5.6 Comparación de Controladores

Observar las diferencias del comportamiento del banco de pruebas cuando se utilizan los distintos controladores tipo PID (P, PI, PD y PID), y de alguna manera comprender el efecto que tiene el agregar una constante K_i o K_d en un control P, y finalmente observar la aplicación del control P con ambas constantes.

5.7 Sintonización de Controladores PID, Método Ziegler- Nichols (Oscilaciones continuas)

Incrementar la constante K_p hasta observar oscilaciones continuas y aplicar el método de oscilaciones continuas, para diseñar un controlador P, PI, PD o PID.

5.8 Relación de prácticas con diversas materias

La mayoría de las prácticas realizadas tienen principal relación con la materia de Control Automático, pero el uso de conceptos u aplicaciones se pueden aplicar en otras materias.

Como ejemplo las siguientes relaciones entre las materias está el uso del encoder, en las materias de Instrumentación en la cual se aprende a hacer el uso de este mismo, así como en Diseño Mecatrónico; que a la vez se emplea en aplicaciones de medición de velocidad angular o posición, dependiendo del problema que se desea resolver.

Otro ejemplo en común es la práctica Diseño de Controladores, en la cual se realizan operaciones para encontrar parámetros para un controlador PID esto para la asignatura de Control Automático, mientras que para Instrumentación y Diseño Mecatrónico es aplicado para los sistemas que se desarrollen en clase, tal como el control de luz o el control de humedad en un invernadero.

Materias / Prácticas	Control Automático	Instrumentación y Control	Instrumentación	Modelado	Diseño Mecatrónico
1.-Encoder		*	*		*
2.Estabilización	*				
3.-Regulación	*				
4.-Modelado Matemático	*	*		*	
5.-Diseño Controladores	*	*			*
6.-Comparación de Controladores	*	*			*
7.-Ziegler Nichols	*		*		*

Tabla 5-1 Relación de prácticas con algunas de las asignaturas de la carrera de Ingeniería Mecatrónica

6 Prácticas



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA



Práctica 01

6.1 “Codificador (*encoder*) de Cuadratura”

Objetivos

- Conocer el funcionamiento de un codificador (*encoder*).
- Calcular la posición angular de un motor con ayuda de un codificador.
- Calcular la velocidad de un motor con ayuda de un codificador

Introducción

Encoder

Es un sensor electromecánico que genera señales digitales en respuesta al movimiento.

Capaz de dar información concerniente a la posición, velocidad y dirección.

Los codificadores pueden ser utilizados en una gran variedad de aplicaciones como retroalimentación para el control de la velocidad en motores, sensores para medición de corte y de posición [13].

Ejemplos de aplicación [13]

- Dispositivo de control de puertas
- Máquinas de lente demoledor
- Soldadura ultrasónica
- Máquinas etiquetadoras
- Indicación x/y
- Máquinas taladradoras
- Robótica
- Plotter
- Máquinas de ensamblaje
- Dispositivos de análisis
- Equipo médico
- Máquinas mezcladoras

Existen dos tipos de *encoder*: Movimiento lineal y rotacional. El primero de estos responde al movimiento a lo largo de un camino, mientras que el segundo responde al movimiento rotacional.

También se pueden categorizar por el tipo de respuesta entregada:

Absoluto: Generan multi-bits que indican directamente su posición actual.

Incremental: Genera un tren de pulsos, el cual es empleado para determinar posición y velocidad. El codificador de cuadratura es un caso especial del incremental; se conoce con este nombre debido a

que sus señales digitales de salida están desfasadas 90°. Dicho desfase permite conocer el sentido de la dirección (horaria o anti-horaria).

El encoder de cuadratura o incremental es el instalado en el banco de pruebas, el cual cuenta con dos señales A y B. La resolución está dada por el CPR, que es la suma de flancos de subida y bajada de las señales A y B

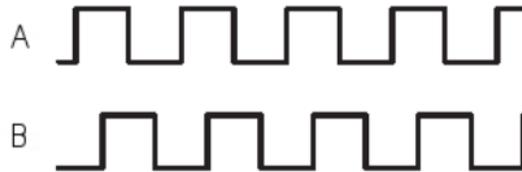


Figura 6-1 Señales con desfase de 90°.

Material o equipo a utilizar

- Banco de Pruebas.
- Cable DB15.
- Microcontrolador. (A elección del alumno)
- Computadora.
- Osciloscopio y cable BNC.

Desarrollo

PARTE UNO

Profesor

Actividad 1. Explicar a los alumnos que cada 360° de rotación del eje del motor equivalen a 64 flancos de subida y bajada contados en ambas señales, A y B; si sólo se consideran los flancos de subida y de bajada de una de las señales, entonces se tienen 32. Por lo tanto, 16 flancos de subida de alguna de las dos señales del codificador (*encoder*) equivalen a una revolución del motor del banco de pruebas. Si se considera que sólo hay un flanco de subida por periodo, entonces la frecuencia de cualquiera de las señales ($f_{encoder}$) es $K_{flancos}$ veces (flancos de subida de una sola señal) la frecuencia de rotación del eje del motor (f_{motor}).

Notas:

Es necesario hacer énfasis en $K_{flancos} = 16$. Los alumnos requerirán este dato para cálculos posteriores.

Alumno

Actividad 1. Realizar un código de programación, en el microcontrolador de su preferencia, que le permita hacer girar el motor del banco de pruebas a su máxima velocidad, por ejemplo en uC de 8 bits de resolución en pwm, se utilizaría un valor de 255, para alcanzar la velocidad máxima. (Utilizar el cable DB15 para conectar su microcontrolador con el banco de pruebas y verificar que el código funcione correctamente).

Actividad 2. Con ayuda de un osciloscopio, obtener la frecuencia de una de las señales del codificador que posee el motor del banco de pruebas, ($f_{encoder}$).

Actividad 3. Con el dato obtenido en la actividad anterior, calcular la frecuencia del motor (f_{motor}) y la velocidad del motor en revoluciones por minuto (ω_{motor} [rpm]). Para ello, utilizar las siguientes ecuaciones.

$$f_{motor} = \frac{f_{encoder}}{K_{flancos}}$$
$$\omega_{motor} = 2 * \pi * f_{motor} \left[\frac{rad}{s} \right]$$

$$1[rpm] = \frac{360^\circ}{60[s]} = \frac{2\pi[rad]}{60[s]}$$

Actividad 4. El valor ω_{motor} [rpm] corresponde a la velocidad del motor en revoluciones por minuto antes de la etapa de reducción (*factor de reducción* : $K_{reduccion}$). Con base en el enunciado anterior, utilizar el factor de reducción del motor empleado (véase en la descripción del material) para obtener la velocidad que entrega la flecha del motor en revoluciones por minuto, (ω_{flecha} [rpm]).

$$\omega_{motor} = K_{reduccion} \omega_{flecha}$$

Actividad 5. ¿Cuál es el error relativo de la velocidad ω_{flecha} obtenida respecto a la mostrada en las especificaciones del motor empleado?

$$E_{relativo} = \frac{valor_{real} - valor_{medido}}{Valor_{real}} \times 100$$

PARTE DOS

Profesor

Actividad 1. Explicar a los alumnos el procedimiento para determinar la posición angular del eje del motor (zona de interés).

El codificador envía $Encoder_{pulsos}$ pulsos cada vez que el motor ha dado una vuelta, pero la flecha del motor después de la reducción solo ha dado $\frac{1}{K_{reduccion}}$ pulsos (factor de reducción del motor que se esté utilizando). Entonces los pulsos que da la flecha se obtienen de la siguiente manera.

$$Flecha_{pulsos} = K_{reduccion} * Encoder_{pulsos}$$

Una vez que se tienen los pulsos que corresponden al punto de interés, la relación de pulsos y posición angular queda de la siguiente manera.

$$\frac{Flecha_{pulsos}}{360^\circ} = \frac{x \text{ pulsos}}{1^\circ}$$

Nota: Los datos necesarios para realizar los cálculos se encuentran en la sección Descripción de material eléctrico o electrónico de este manual.

Alumno

Actividad 1. Responder las siguientes preguntas con base en las expresiones vistas previamente.

$$Flecha_{pulsos} = K_{reduccion} * Encoder_{pulsos}$$

$$\frac{Flecha_{pulsos}}{360^\circ} = \frac{x \text{ pulsos}}{1^\circ}$$

- 1.- ¿Cuántos pulsos del encoder equivalen a un grado de desplazamiento angular del eje de interés?
- 2.- Con base en la respuesta anterior, ¿cuántos grados del eje de interés equivalen a un pulso del codificador?

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO



FACULTAD DE INGENIERÍA



Práctica 2

6.2 “Objetivos de Control (Estabilización)”

Esta práctica solamente es demostrativa.

Objetivos

- Identificar y comprender el concepto del primer objetivo de control (estabilización).

Introducción

El objetivo de control “estabilización” consiste en que la respuesta del sistema con control debe regresar al cero, cuando su condición inicial sea un valor diferente de cero.

Material o equipo a utilizar

- Banco de Pruebas.
- Computadora con la “Interfaz_1” instalada, y la IDE de ENERGIA; para el código de esta práctica.
Si la práctica se decide hacer con la arquitectura abierta
- Cable DB15
- Microcontrolador¹: Tiva C Series TM4C123G. EK-TM4C123GXL
- Cables tipo jumper mínimo 15 piezas (Hembra- hembra, hembra-macho)
- Protoboard

Desarrollo

En caso de no usar la arquitectura abierta del banco de prueba, omitir el paso 1.

1.- Conectar el cable DB15 al banco de pruebas, el microcontrolador; y este conectarlo a la computadora. Ver los pines de conexión del microcontrolador y el banco de pruebas; en el programa

¹ El código proporcionado se realizó en la *IDE Energia* (para microcontroladores TI). Sin embargo, se puede utilizar cualquier otro microcontrolador, haciendo los cambios pertinentes de lenguaje, pero conservando la misma estructura de la cadena de datos que el microcontrolador envía a la computadora.

correspondiente a esta práctica.²

2.-Energizar el banco de pruebas (switch apagado).

3.- Abrir la Interfaz de LabVIEW “Interfaz_1”. Cargar el programa “P2_P3_P6” al microcontrolador que se encuentra en la capeta de códigos, encender el banco de pruebas (switch encendido).

4.- Presionar el **botón azul del banco de pruebas**; hasta que se aprecie que la carátula se encuentra en 0 grados (no es necesario ser exactos), presionar el **botón rojo del banco de pruebas** (o el reset del microcontrolador), para establecer el origen tanto en el microcontrolador como en el motor.



Figura 6-2 Banco de Prueba Carátula

5.- En la interfaz_1, llenar los cuadros de texto con la información solicitada (Ubicación de carpeta, nombre de archivo y puerto COM) y presionar el botón RUN. El led indicador del bluetooth, dejará de parpadear, lo que significa que la conexión se realizó correctamente.

6.-Presionar el botón Prueba, elegir la opción Objetivos de Ctrl. Figura 6-4. Posteriormente elegir la opción Estabilización (Figura 6-3)



Figura 6-4 Botones para selección de prácticas

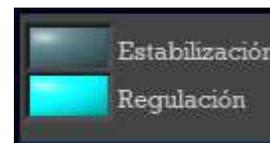


Figura 6-3 Estabilización/Regulación

7.- Presionar el **botón azul del banco de pruebas** para colocar al motor en una posición arbitraria (dentro del intervalo de 0 a 360 grados). Presionar el **botón Play** que se encuentra en la interfaz_1, una vez terminada la secuencia (el motor dejó de moverse) **presionar botón Stop**. El controlador

² Ver conexiones del cable DB15 en *Descripción de tarjeta PCB PinOut* y las características del microcontrolador en *Descripción Material Electrónico*.

empleado es un PID con las siguientes constantes $k_p=3$, $k_i=0.001$ y $k_d=0.01$ las constantes se sintonizaron a prueba y error, el programa no permite modificar dichos valores.

8.-Limpiar la pantalla del *display* de la gráfica dando clic en el botón *Reset Graph*, para asegurar que la pantalla este limpia.

9.- Si se desea realizar otra prueba, presionar el botón rojo del banco de pruebas y repetir los pasos siete y ocho. Intentar varias pruebas con valores dentro del rango permitido (0° a 360°).

10.- Si se desea graficar algún experimento anterior; presionar el botón Graficar y seguir las instrucciones de la interfaz para exportar la imagen de la gráfica del experimento seleccionado.

El programa por seguridad despliega un mensaje de sobre escritura al momento de exportar la imagen, solo dar **OK** si es la primera vez que se exporta la imagen del mismo archivo seleccionado.

11.-Para finalizar el experimento, presionar el botón *Power* (Figura 6-5) de la interfaz, cerrarla, apagar el banco de pruebas y desconectarlo de la toma de corriente.



Figura 6-5 Botón Power

Conclusión:

Se presentó el objetivo de control de estabilización para la posición de la flecha del banco de pruebas, cuya planta es la siguiente:

$$\frac{\Theta(s)}{E(s)} = \frac{30.2}{s(s + 4.76)}$$

Se aprecia que este objetivo de control se cumple ya que sin importar la posición en la que se coloque la flecha, ésta regresa a la posición aproximada de cero.

Nota:

Se sugiere que la diferencia entre la posición cero y la posición arbitraria en que se coloque la flecha debe ser mayor a 10 grados; para apreciar el efecto en la gráfica y evitar hacer grandes ampliaciones.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO



FACULTAD DE INGENIERÍA



Práctica 3

6.3 “Objetivos de Control (Regulación)”

Objetivos

- Identificar y comprender del segundo concepto de objetivo de control (regulación) mediante la demostración con el banco de pruebas.

Introducción

El objetivo de control “regulación” consiste en que la respuesta del sistema con control debe alcanzar un valor deseado constante cuando su condición inicial tome un valor arbitrario.

Con base en lo anterior, se concluye que la “estabilización” es un caso particular de la “regulación” debido a que el estado cero también es considerado un valor deseado para algunos sistemas de control.

Material o equipo a utilizar

- Banco de Pruebas.
- Computadora con la “Interfaz_1” instalada, y la IDE de ENERGIA; para el código de esta práctica.

Si la práctica se decide hacer con la arquitectura abierta

- Cable DB15
- Microcontrolador³: Tiva C Series TM4C123G. EK-TM4C123GXL
- Cables tipo jumper mínimo 15 piezas (Hembra- hembra, hembra-macho)
- Protoboard

³ El código proporcionado se realizó en la *IDE Energia* (para microcontroladores TI). Sin embargo, se puede utilizar cualquier otro microcontrolador, haciendo los cambios pertinentes de lenguaje, pero conservando la misma estructura de la cadena de datos que el microcontrolador envía a la computadora.

Desarrollo

En caso de no usar la arquitectura abierta del banco de prueba, omitir el paso 1.

1.- Conectar el cable DB15 al banco de pruebas, el microcontrolador; y este conectarlo a la computadora. Ver los pines de conexión del microcontrolador y el banco de pruebas; en el programa correspondiente a esta práctica.⁴

2.-Energizar el banco de pruebas (switch apagado).

3.- Abrir la Interfaz de LabVIEW “Interfaz_1”. Cargar el programa “P2_P3_P6” al microcontrolador que se encuentra en la capeta de códigos, encender el banco de pruebas (switch encendido).

4.- Presionar el **botón azul del banco de pruebas**; hasta se aprecie que la carátula se encuentra en 0 grados (no es necesario ser exactos), presionar el **botón rojo del banco de pruebas** (o el reset del microcontrolador), para establecer el origen tanto en el microcontrolador como en el motor.



Figura 6-6 Banco de Pruebas carátula

5.- En la interfaz_1 llenar los cuadros de texto con la información solicitada (Ubicación de carpeta, nombre de archivo y puerto COM) presionar el botón RUN. El led indicador del bluetooth, dejará de parpadear, lo que significa que la conexión se realizó correctamente.

6.-Presionar el botón Prueba, elegir la opción Objetivos de Ctrl. Figura 6-8. Posteriormente elegir la opción Regulación (Figura 6-7).



Figura 6-8 Botones para selección de prácticas



Figura 6-7 Estabilización/Regulación

⁴ Ver conexiones del cable DB15 en *Descripción de tarjeta PCB PinOut* y las características del microcontrolador en *Descripción Material Electrónico*.

7.- Presionar el **botón azul del banco de pruebas** para colocar al motor en una posición arbitraria (dentro del intervalo de 0 a 360 grados). Ingresar el valor deseado (Figura 6-9) de posición. Como primera iteración se sugiere empezar con la flecha en la posición cero y para la posición deseada empezar con valores pequeños, por ejemplo 30 grados e ir incrementando. Presionar el **botón Play que se encuentra en la interfaz_1**, una vez terminada la secuencia (el motor deje de moverse) presionar el botón Stop. El controlador empleado es un PID con las siguientes constantes $k_p=10$, $k_i=0.007$ y $k_d=0.9$, el programa no permite modificar dichos valores.



Figura 6-9 Perilla de Valor Deseado.

8.-Limpiar la pantalla del *display* de la gráfica dando clic en el botón *Reset Graph*, para asegurar que la pantalla este limpia.

9.- Si se desea realizar otra prueba, presionar el **botón rojo del banco de pruebas** y repetir los pasos siete y ocho. Intentar varias pruebas con valores dentro del rango permitido (0° a 360°).

10.- Si se desea graficar algún experimento anterior; presionar el botón Graficar y seguir las instrucciones de la interfaz para exportar la imagen de la gráfica del experimento seleccionado. El programa por seguridad despliega un mensaje de sobre escritura al momento de exportar la imagen, solo dar **OK** si es la primera vez que se exporta la imagen del mismo archivo seleccionado.

11.-Para finalizar el experimento, presionar el botón *Power* (Figura 6-10) de la interfaz, cerrarla, apagar el banco de pruebas y desconectarlo de la toma de corriente.



Figura 6-10 Botón Power

Conclusión:

Se presentó el objetivo de control de regulación para la posición de la flecha del banco de pruebas, cuya planta es la siguiente:

$$\frac{\Theta(s)}{E(s)} = \frac{30.2}{s(s+4.76)}$$

Se aprecia que este objetivo de control se cumple ya que es posible posicionar la flecha en el valor que se desee sin importar la posición previa en la que esta se coloque. Además se aprecia que si el valor ingresado para posición deseada fuera cero, se podría cumplir también con el objetivo de control de estabilización. Comprobando así que la estabilización es un caso particular de la regulación.

Nota:

La diferencia entre el valor deseado y el medido deber ser mayor a 10 grados para que se aprecie el efecto de regulación claramente, para evitar hacer ampliaciones.

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO FACULTAD DE INGENIERÍA



Práctica 4

6.4 “Modelado Matemático”

Objetivos

- Obtener la función de transferencia para el control de posición y velocidad del sistema (banco de pruebas) y verificar el resultado con el sistema físico.
- Las pruebas que se proponen realizar sobre el motor cumplen además con el objetivo de introducir al alumno en el proceso de caracterización de motores de CD

Introducción

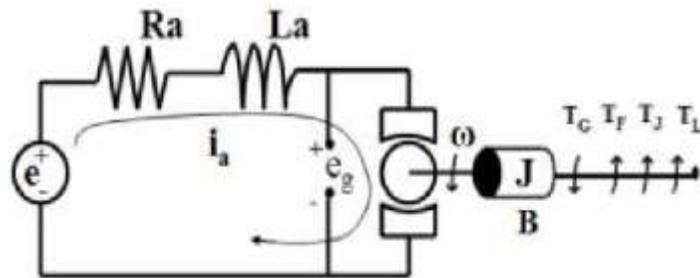


Figura 6-11 Diagrama de motor CD

Tabla 6-1 Nomenclatura

Nomenclatura	
Ra [Ω]	Resistencia de armadura
La [H]	Inductancia de armadura
ia [A]	Corriente de armadura
e [V]	Voltaje de entrada
eg [V]	Voltaje generado
ω [rad/s]	Velocidad angular
TG [Nm]	Par generado
TF [Nm]	Par debido a la fricción
TJ [Nm]	Par debido a la inercia
TL [Nm]	Par debido a la carga
J	Inercia
B	Coefficiente de fricción viscosa

Material o equipo a utilizar

- Banco de Pruebas.
- Computadora con la "Interfaz_2" instalada, y la IDE de ENERGIA; para el código de esta práctica.
- Microcontrolador⁵: Tiva C Series TM4C123G. EK-TM4C123GXL

Si la práctica se decide hacer con la arquitectura abierta

- Cable DB15.
- Cables tipo jumper 15 piezas (Hembra- hembra, hembra-macho).
- Protoboard
- Fuente de Voltaje variable.

Desarrollo

Profesor

Actividad 1. Realizar y explicar el procedimiento para la obtención del modelo matemático de un motor de corriente directa (Figura 6-11).

I.- Aplicando la ley de voltajes de Kirchhoff

$$e(t) = R_a i_a(t) + L_a \frac{di_a(t)}{dt} + e_g(t) \rightarrow (1)$$

Donde

$$e_g(t) = K\phi(t)\omega(t) \rightarrow (2)$$

Por efectos de linealidad

$$e_g(t) = K_e \omega(t) \rightarrow (3)$$

K_e , constante de velocidad del motor.

II.- Considerando un flujo magnético constante, el par generado es proporcional a la corriente y está dado por:

⁵El código proporcionado se realizó en la IDE Energia (para microcontroladores TI). Sin embargo, se puede utilizar cualquier otro microcontrolador, haciendo los cambios pertinentes de lenguaje, pero conservando la misma estructura de la cadena de datos que el microcontrolador envía a la computadora.

$$T_G = K_t i_a(t) \rightarrow (4)$$

K_t , constante de par del motor.

III.- El modelado de la parte mecánica queda de la siguiente manera

$$T_G = TF(t) + TJ(t) + TL(t) \rightarrow (5)$$

Donde

$$TF = Tf(t) + B\omega(t) \rightarrow (6)$$

$$TJ = J \frac{d\omega(t)}{dt} \rightarrow (7)$$

IV.- Por lo que al sustituir la ecuación 6 y 7 en la 5, se obtiene:

$$T_G = Tf(t) + B\omega(t) + J \frac{d\omega(t)}{dt} + TL(t) \rightarrow (8)$$

V.- Las ecuaciones fundamentales del modelado de un motor DC queda representado por las ecuaciones 1, 3, 4 y 8.

Por facilidad algebraica se utiliza la transformada de Laplace, teniendo así:

$$E(s) - E_g = (R_a + sL_a)I_a(s) \rightarrow (10)$$

$$E_g(s) = K_e \Omega(s) \rightarrow (11)$$

$$T_G(s) = K_t I_a(s) \rightarrow (12)$$

$$T_G(s) - T_F(s) - T_L(s) = (B + sJ)\Omega(s) \rightarrow (13)$$

VI.- Con base en las ecuaciones 10, 11, 12 y 13 se obtiene la función de transferencia (FT), que relaciona la velocidad angular y el voltaje aplicado en las terminales del motor. Ya que $T_F(s)$ y $T_L(s)$ son variables que se controlan discretamente en el banco de pruebas para observar la respuesta del controlador ante perturbaciones, se pueden simplificar las funciones de transferencia $T_F(s) = 0$ y $T_L(s) = 0$.

$$\frac{\Omega(s)}{E(s)} = \frac{K_t}{R_a B \left(\frac{L_a}{R_a} s + 1 \right) \left(\frac{J}{B} s + 1 \right) + K_t K_e} \rightarrow (14)$$

Donde $\frac{L_a}{R_a}$ y $\frac{J}{B}$ son las constantes de tiempo eléctrica y mecánica respectivamente del sistema.

VII.- Normalmente los valores de L_a son de magnitud menor a R_a , por lo que, para valores pequeños de L_a , $L_a \rightarrow 0$; la constante de tiempo eléctrica es despreciable respecto a la constante mecánica, por ello la FT (Función de Transferencia) al simplificarse queda:

$$\frac{\Omega(s)}{E(s)} = \frac{K_t}{R_a(Js+B)+K_tK_e} \rightarrow (15)$$

VIII.- La forma simplificada de la FT de velocidad angular entre el voltaje de un motor DC, es la siguiente:

$$\frac{\Omega(s)}{E(s)} = \frac{K}{s+P} \rightarrow (16)$$

Donde:

$$K = \frac{K_t}{R_aJ}; \quad P = \frac{BR_aJ+K_tK_e}{R_aJ}$$

IX.- Con el conocimiento de la FT donde la variable de interés es la velocidad angular, se puede obtener una FT donde la variable de interés es el desplazamiento angular.

$$\frac{\Omega(s)}{sE(s)} = \frac{K}{s(s+P)} \rightarrow (17)$$

$$\frac{\Theta(s)}{E(s)} = \frac{K}{s(s+P)} \rightarrow (18)$$

Nota 1. Explicar al alumno que al estar en el dominio de Laplace y usar un integrador sobre esta misma, se obtiene la FT donde la variable de interés es el desplazamiento.

X.- Con una entrada escalón de amplitud Ae entonces $E(s) = \frac{Ae}{s}$. La ecuación 16 queda de la siguiente forma:

$$\Omega(s) = \frac{K * Ae}{s(s+P)} \rightarrow (16.1)$$

Al separarse en fracciones parciales:

$$\Omega(s) = \frac{M}{s} + \frac{N}{s+P} \rightarrow (16.2)$$

Donde

$$M = \frac{K \cdot A e}{P} = -N \rightarrow (16.3)$$

XI.- Regresando al dominio del tiempo:

$$\omega(s) = M(1 - e^{-P \cdot t}) * u(t) \rightarrow (19)$$

Actividad 2. Obtener los parámetros del motor de DC.

I.- Explicar a los alumnos que la resistencia de armadura (R_a) se puede determinar midiendo directamente en las terminales del motor con un óhmetro.

$R_a = 2.4$ OHMS. Este valor de resistencia es el valor medido cuando no se energiza al motor, se coloca como referencia. Pero dicho valor puede variar por diversos factores como; el desgaste o la temperatura del motor.

II.- Para continuar con la obtención de los parámetros del motor se debe variar el voltaje de alimentación del banco de pruebas.

Para variar el voltaje suministrado, se necesita desconectar los cables banana-banana (**¡Error! No se encuentra el origen de la referencia.**) que provienen de la fuente interna, y conectar la fuente externa de voltaje variable (**¡Error! No se encuentra el origen de la referencia.**).



Figura 6-12 Conexión para fuente interna/externa.



Figura 6-13 Fuente de Laboratorio.

Al conectar la fuente voltaje externa, asegurar que el voltaje suministrado no exceda los 12 [v] antes de habilitar la misma. Posteriormente utilizar la interfaz de usuario y así obtener los valores del polo y la constante K.

III.- Explicar a los alumnos que de la ecuación 19 $\omega(s) = M(1 - e^{-s})$, M es el valor de la velocidad [rpm] en estado estable y ω es la velocidad angular cuando alcanza el 63.2% del valor final.

De igual manera explicar a los alumnos que para la obtención de la constante K se utiliza la relación existente en la ecuación 16.3.

$$M = \frac{K \cdot A e}{p} = -N \rightarrow (16.3)$$

En la interfaz correspondiente a la práctica, se podrá apreciar la gráfica de la velocidad ω , en la cual los alumnos tendrán que encontrar el instante en que ésta alcanza el 63.2% del valor final, es decir, encontrar la constante de tiempo (τ).

Finalmente, explicar a los alumnos que para encontrar el Polo del sistema, hay que sacar el recíproco del tiempo en que sucede $M(1 - e^{-s})$, observando también la gráfica de la velocidad ω .

Nota 2. La explicación anterior es importante para que los alumnos puedan llenar la tabla mostrada en su lista de actividades. Para así calcular los parámetros pertinentes y obtener ambas funciones de transferencia, la de posición y la de velocidad.

Resultados aproximados:

$$\frac{\Omega(s)}{E(s)} = \frac{30.2}{s+4.76} \quad \frac{\Theta(s)}{E(s)} = \frac{30.2}{s(s+4.76)}$$

Alumno

Actividad 1. Con las ecuaciones:

$$E(s) - E_g = (R_a + sL_a)I_a(s)$$

$$E_g(s) = K_e\Omega(s)$$

$$T_G(s) = K_t I_a(s)$$

$$T_G(s) - T_F(s) - T_L(s) = (B + sJ)\Omega(s)$$

Y considerando:

$$T_F(s) = 0 \quad \text{y} \quad T_L(s) = 0$$

Comprobar que la función de transferencia (FT) que relaciona la velocidad angular y el voltaje aplicado en las terminales del motor es la siguiente:

$$\frac{\Omega(s)}{E(s)} = \frac{K_t}{R_a B \left(\frac{L_a}{R_a} s + 1 \right) \left(\frac{J}{B} s + 1 \right) + K_t K_e}$$

Actividad 2. Considerando $L_a \rightarrow 0$, comprobar que la simplificación de la FT del ejercicio anterior es:

$$\frac{\Omega(s)}{E(s)} = \frac{\frac{K_t}{R_a J}}{s + \frac{B R_a J + K_t K_e}{R_a J}}$$

Actividad 3. Con base en $\frac{\Omega(s)}{E(s)} = \frac{K}{s+P}$ y $\frac{\Omega(s)}{E(s)} = \frac{\frac{K_t}{R_a J}}{s + \frac{B R_a J + K_t K_e}{R_a J}}$; encontrar la relación que tienen las

constantes K y P con K_t , K_e , R_a , J y B .

Actividad 4. Buscar el valor de la resistencia de armadura (R_a) del motor del banco de pruebas. O si se los permite el profesor medirla directamente de las terminales del motor o buscar en la descripción correspondiente al motor empleado.

Actividad 5. Con ayuda de la explicación del profesor, completar la siguiente tabla.

ΔT es la diferencia de tiempo en que la velocidad alcanza el 63.2% (T_i) menos el tiempo en que la velocidad es mayor a cero (T_0).

Como apoyo revisar las ecuaciones 16, 16.3 y 19.

Se recomienda hacer la tabla en un archivo de Excel para facilitar la aplicación de operaciones.

Los valores de voltaje Ae de la tabla son una propuesta a seguir, pero pueden variar según el ajuste que se obtenga con la perilla de la fuente. No es necesario llenar toda la tabla, es decir llegar a 6 [v], con 4 o 5 mediciones de voltaje distinto son suficientes.

Tabla de voltajes y tiempos							
Ae [V]	M [RPM]	63.2% M [RPM]	Tiempo T0 [ms]	Tiempo Ti [ms]	ΔT [ms]	P[1/s]	K [RPM/Vs]
12							
11							
10							
9							
8							
7							
6							
					PROMEDIO		

Tabla 6-2

Empezar con el experimento.

En caso de no usar la arquitectura abierta del banco de prueba, omitir el paso uno.

- 1.- Conectar el cable DB15 al banco de pruebas, el microcontrolador; y este conectarlo a la computadora. Ver los pines de conexión del microcontrolador y el banco de pruebas; en el programa correspondiente a esta práctica.⁶
- 2.- Conectar el banco de pruebas a la fuente de poder, quitando los cables de la fuente interna y hacer la conexión mediante dos cables banana-banana, el color negro corresponde al negativo y el rojo al positivo, asegurar que la conexión sea correcta. En la fuente de poder girar la perilla de corriente al máximo, la perilla correspondiente al voltaje, colocarlo a 12 [V]. Por seguridad del equipo, el botón OUTPUT de la fuente debe estar desactivado; hasta que se indique lo contrario.
- 3.- Abrir la Interfaz de LabVIEW "Interfaz_2". Cargar el programa "P4_ModeladoMatematico" al microcontrolador, activar el botón OUTPUT de la Fuente de alimentación.

⁶ Ver conexiones del cable DB15 en *Descripción de tarjeta PCB PinOut* y las características del microcontrolador en *Descripción Material Electrónico*.

4.- En la interfaz_2, llenar los cuadros de texto con la información solicitada (Ubicación de carpeta, nombre de archivo y puerto COM) y presionar el botón RUN. El led indicador del *bluetooth*, dejará de parpadear, lo que significa que la conexión se realizó correctamente.

5.- Presionar el botón Prueba.

6.- Dar reset al sistema con el **botón rojo del banco de prueba**, posteriormente presionar el botón Play que se encuentra en la interfaz_2, una vez terminada la secuencia presionar el botón Stop.

7.- Anotar en la Tabla1 el valor de voltaje aplicado y la velocidad alcanzada (M), para obtener el valor del 63.2% de la velocidad.

8.- Con ayuda de las herramientas de *Graph Palette* (Figura 6-14) ubicar en que tiempo (Ti) sucede el 63.2% de M, y el tiempo en que el sistema empieza a funcionar, es decir, el tiempo cero (T0). Anotar los tiempos obtenidos, en la tabla para finalmente obtener el valor del polo y la constante K.



Figura 6-14 Graph Palette

9.- Limpiar la pantalla del *display* de la gráfica, dando clic en el botón *Reset Graph*. Para asegurar que la pantalla este limpia.

10.- Disminuir una unidad el voltaje de la fuente de poder y repetir los pasos del 6 al 9, hasta completar la tabla.

11.- Si se desea graficar algún experimento anterior; presionar el botón Graficar y seguir las instrucciones de la interfaz para exportar la imagen de la gráfica del experimento seleccionado. El programa por seguridad despliega un mensaje de sobre escritura al momento de exportar la imagen, solo dar **OK** si es la primera vez que se exporta la imagen del mismo archivo seleccionado.

12.-Para finalizar el experimento, presionar el botón *Power* (Figura 6-15) de la interfaz, cerrarla, apagar el banco de pruebas y desconectarlo de la toma de corriente.



Figura 6-15 Botón Power

Actividad 6.

a) De la información registrada en la tabla, promediar los valores de P y de K.

b) Sustituir P y K en:

La función de transferencia que relaciona la velocidad angular y el voltaje aplicado en las terminales del motor:

$$\frac{\Omega(s)}{E(s)} = \frac{K}{s + P}$$

La función de transferencia que relaciona la posición angular y el voltaje aplicado en las terminales del motor:

$$\frac{\Theta(s)}{E(s)} = \frac{K}{s(s + P)}$$

c) Considerando que $\frac{\Omega(s)}{E(s)} = \frac{K}{s+P}$ es de primer orden, calcular también el tiempo de asentamiento (ts).

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA



Práctica 5

6.5 “Diseño de Controladores con Base en la Función de Transferencia”

Objetivos

- Diseñar un controlador con base en la Función de Transferencia que representa la salida (velocidad) entre entrada (voltaje) del banco de pruebas.
- Probar el controlador diseñado en simulación numérica y en el banco de pruebas

Introducción

Objetivos de control

- ❖ Estabilización: El sistema alcanza el valor cero desde cualquier punto inicial.

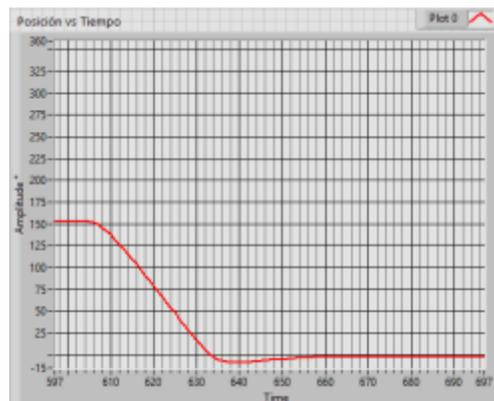


Figura 6-16 Estabilización (posición vs tiempo)

- ❖ Regulación: Que la variable a controlar sea llevada a un valor deseado constante en el tiempo.

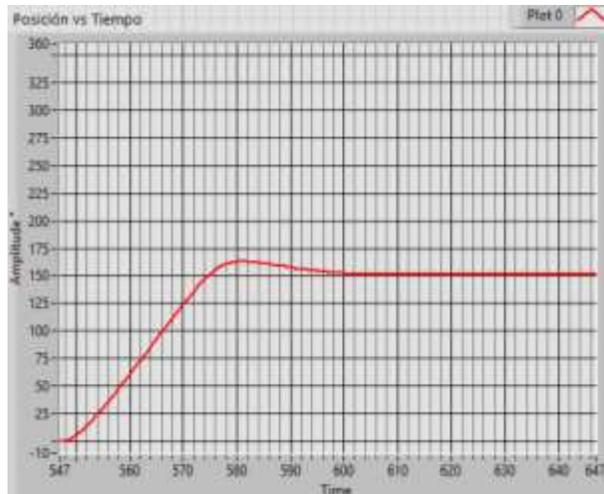


Figura 6-17 Regulación posición vs tiempo.

- ❖ Seguimiento: El sistema siga una secuencia de valores deseados (Trayectoria).

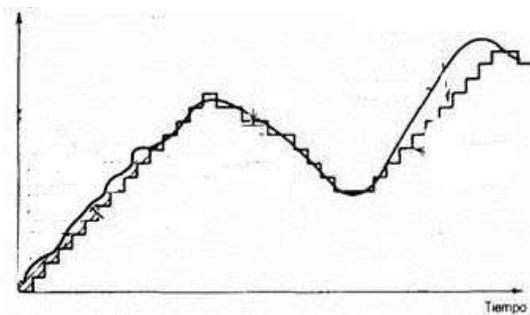


Figura 6-18 Seguimiento.

- ❖ Desempeño: Es cumplir con restricciones de sobre paso, tiempo de asentamiento, tiempo de asentamiento y tiempo pico.

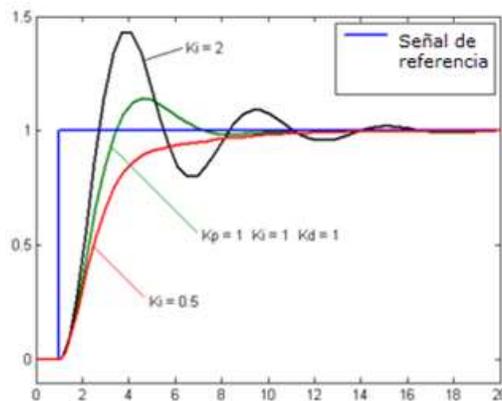


Figura 6-19 Desempeño.

- ❖ Robustez: cualidad de presentar un buen funcionamiento a pesar de la existencia de perturbaciones y/o incertidumbre paramétrica

Error en estado permanente		Tipo de Entrada		
		Escalón $r(t)=Ae$	Rampa $r(t)=mt$	Parábola $r(t)=\alpha t^2$
Tipo de GH(S)	0	cte	∞	∞
	1	0	Cte2	∞
	2	0	0	Cte3
	3	0	0	0

Tabla 6-3 Error en estado permanente

Nota: Se sugiere al profesor explicar los objetivos de control a sus alumnos antes de realizar esta práctica. Así como también una introducción al tema de los tipos de error según GH(s) contra el tipo de entrada

Material o equipo a utilizar

- Banco de Pruebas.
- Computadora con la “Interfaz_1” instalada, y la IDE de ENERGIA; para el código de esta práctica.

Si la práctica se decide hacer con la arquitectura abierta

- Cable DB15
- Microcontrolador⁷: Tiva C Series TM4C123G. EK-TM4C123GXL
- Cables tipo jumper mínimo 15 piezas (Hembra- hembra, hembra-macho)
- Protoboard

Desarrollo

PARTE UNO

Profesor

Actividad 1. Proponer el siguiente ejercicio a los alumnos: Diseñar un controlador para la planta que

⁷ El código proporcionado se realizó en la *IDE Energia* (para microcontroladores TI). Sin embargo, se puede utilizar cualquier otro microcontrolador, haciendo los cambios pertinentes de lenguaje, pero conservando la misma estructura de la cadena de datos que el microcontrolador envía a la computadora.

se muestra a continuación, de tal forma que se cumplan los objetivos de control de estabilización, regulación y desempeño.

$$U(s) \rightarrow \frac{30.2}{s + 4.76} \rightarrow Y(s)$$

Actividad 2. Realizar el análisis de la planta junto con los alumnos.

Estabilidad. Polo $P1=-4.76$, como el polo es negativo la planta es estable. Explicar a qué se debe que sea estable (polos del lado izquierdo del eje imaginario, planta estable; polos en cero o sobre el eje imaginario, planta marginalmente estable; polos del lado derecho del eje imaginario, planta inestable).
 $y^\infty = 80 \text{ rpm}$ (Valor tomado de la descripción del material banco grande)

$$T = \frac{1}{|p|} = \frac{1}{|-4.76|} = 0.210 \text{ [s]}$$

Con base en el criterio de 1% de error,

$$ts = 5 * T = 1.05 \text{ [s]}$$

Actividad 3. Proponer un control proporcional (control P) a los alumnos y explicarlo.

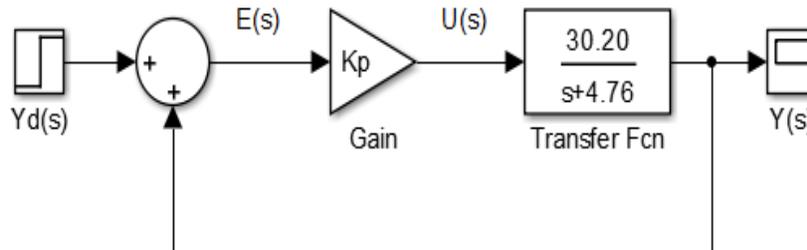


Figura 6-20 Diagrama de bloques Control P.

Ley de control. $U(s)=kp \cdot E(s) \rightarrow u(t)=kp \cdot e(t)$.

Obtener FT de lazo cerrado $T(s)$ y $GH(s)$. Que los alumnos realicen el desarrollo matemático.

$$T(s) = \frac{30.20}{s + 4.76 + 30.20 * Kp}$$

$$GH(s) = \frac{30.20 * Kp}{s + 4.76}$$

Estabilización.

- Polo de lazo cerrado $P_{lc} = -4.76 - 30.20 * Kp$.

- Para que el sistema en lazo cerrado sea estable,

$$P_{lc} < 0 \quad \rightarrow \quad -4.76 - 30.20 * Kp < 0 \quad \rightarrow \quad Kp > -0.1576$$

Regulación.

- GH(s) es de tipo Cero y orden 1. Explicar a los alumnos porqué.
- Entrada escalón y GH(s) de tipo cero, por lo tanto el error en estado permanente:

$$e^{\infty} = cte = \frac{Ae}{1 + \frac{30.20 * Kp}{4.76}}$$

- Estrictamente hablando no se cumple el objetivo de regulación, ya que el error en estado permanente es constante, sin embargo suponiendo que Ae =1

$$e^{\infty} = \frac{1}{1 + \frac{30.20 * Kp}{4.76}}$$

- Si se permite que $e^{\infty} = 0.1 \rightarrow Kp = \frac{4.76}{30.20} \left(\frac{1}{e^{\infty}} - 1 \right) = 1.4185$
- Si se permite que $e^{\infty} = 0.01 \rightarrow Kp = \frac{4.76}{30.20} \left(\frac{1}{e^{\infty}} - 1 \right) = 15.6039$
- Explicar a los alumnos que mientras más grande sea el valor de Kp el error tiende a cero, pero se deben considerar límites físicos en la realidad.

Desempeño. Tiempo de asentamiento ($t_s < t_{sd}$).

$$t_{s_{lc}} = 5T_{lc} \quad T_{lc} = \frac{1}{|P_{lc}|} = \frac{1}{|4.76 + 30.20 * Kp|}$$

$$\text{Si } t_{s_{lc}} = 5 \quad \rightarrow \quad P_{lc} = -4.76 - 30.20 * Kp = -1$$

- *Conclusión.* Con un control P se logra la estabilización y el desempeño deseado pero no la regulación, de forma estricta.

Actividad 4. Proponer un control integral (control I) a los alumnos y explicarlo.

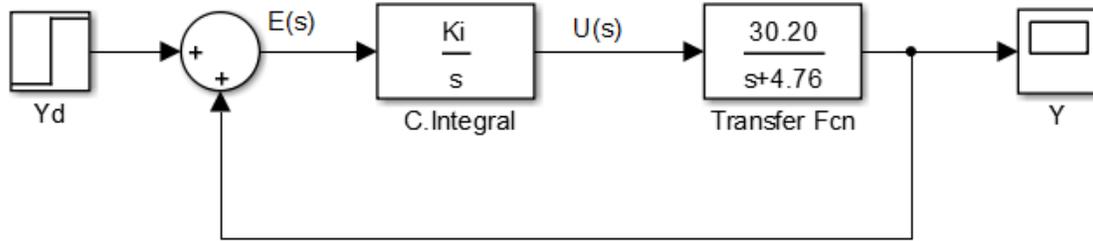


Figura 6-21 Diagrama de bloques Control I

- Obtener FT de lazo cerrado $T(s)$ y $GH(s)$. Que los alumnos realicen el desarrollo matemático.

$$T2(s) = \frac{30.20 * Ki}{s^2 + 4.76 * s + 30.20 * Ki}$$

$$GH2(s) = \frac{30.20 * Ki}{s * (s + 4.76)}$$

Estabilización.

- Con el criterio de Routh-Hurwitz o Newton, se comprueba que la FT $T2(s)$ es estable cuando $Ki > 0$.

Regulación.

- $GH2(s)$ es de tipo 1 y orden 2. Explicar a los alumnos porqué.
- Entonces el error en estado permanente con una entrada escalón es cero. $e_{\infty} = 0$.

Desempeño. Tiempo de asentamiento ($t_s < t_{sd}$).

- De $T2(s)$ la ecuación característica $s^2 + 4.76 * s + 30.20 * Ki$ se obtienen los polos de lazo cerrado:

$$P_{lc} = -2.38 \pm \frac{\sqrt{22.66 - 120.8 * ki}}{2}$$

❖ Caso sobre-amortiguado: $22.66 - 120.8 * Ki > 0$, $Ki < \frac{22.66}{120.8} \approx 0.1876$

❖ Caso críticamente-amortiguado: $22.66 - 120.8 * Ki = 0$, $Ki = \frac{22.66}{120.8} \approx 0.1876$

❖ Caso sub-amortiguado: $22.66 - 120.8 * Ki < 0$, $Ki > \frac{22.66}{120.8} \approx 0.1876$

- Si $t_{sd} = 1.5$ [s] $\rightarrow t_{s_{lc}} = \frac{5}{|Re\{P_{lc}\}|} = \frac{5}{|2.38|} \rightarrow t_{s_{lc}} = 2.1 > 1.5$
- **Conclusión.** Con control I se logra la estabilización y la regulación pero no el desempeño.

Alumno

Actividad 1. Con Matlab Simulink⁸ construir el diagrama de bloques del control P.

Actividad 2. Comprobar el comportamiento del control P, tanto en Simulink como en el banco de pruebas. $K_p=5$ con un valor deseado de 60 [rpm]

Actividad 3. Proponer nuevos valores de K_p y verificar el comportamiento del control P.

Actividad 4. Con Matlab Simulink construir el diagrama de bloques del control I.

Actividad 5. Comprobar el comportamiento del control I, tanto en Simulink como en el banco de pruebas. Utilizar al menos un valor de K_i de cada intervalo para observar los tres casos (Se sugiere probar con los siguientes valores: sobre-amortiguado $k_i=0.05$, sub-amortiguado $k_i=5$, críticamente amortiguado $k_i=0.1876$).

Inicio de experimento

En caso de no usar la arquitectura abierta del banco de prueba, omitir el paso uno.

1.- Conectar el cable DB15 al banco de pruebas, el microcontrolador; y este conectarlo a la computadora. Ver los pines de conexión del microcontrolador y el banco de pruebas; en el programa correspondiente a esta práctica.⁹

2.- Energizar el banco de pruebas.

3.- Abrir la Interfaz de LabVIEW "Interfaz_1". Cargar el programa "P5_DiseñoControladores" al microcontrolador que se encuentra en la carpeta de códigos, encender el banco de pruebas (switch encendido).

4.- En la interfaz_1 llenar los cuadros de texto con la información solicitada (Ubicación de carpeta, nombre de archivo y puerto COM) y presionar el botón RUN. El led indicador del bluetooth, dejará de parpadear, lo que significa que la conexión se realizó correctamente.

5.- Presionar el botón Prueba, para empezar con el experimento y elegir la opción de Diseño de Ctrl (Figura 6-22).



Figura 6-22 Botones para selección de prácticas

⁸ De no contar con licencia válida, se recomienda usar una versión de prueba o para estudiantes.

⁹ Ver conexiones del cable DB15 en *Descripción de tarjeta PCB PinOut* y las características del microcontrolador en *Descripción Material Electrónico*.

6.- Introducir los valores de las constantes K según sea el caso y el valor deseado. **Dar reset al sistema con el botón rojo del banco de prueba**, posteriormente presionar el botón Play que se encuentra en la interfaz_1, una vez terminada la secuencia (el motor dejó de moverse) presionar el botón Stop. Para análisis de la gráfica usar la herramienta Graph Palette.



Figura 6-23 Graph Palette

7.-Limpiar la pantalla del *display* de la gráfica, dando clic en el botón *Reset Graph*. Para asegurar que la pantalla este limpia, repetir los pasos seis al siete si es necesaria otra prueba.

8.- Si se desea graficar algún experimento anterior; presionar el botón Graficar y seguir las instrucciones de la interfaz para exportar la imagen de la gráfica del experimento seleccionado. El programa por seguridad despliega un mensaje de sobre escritura al momento de exportar la imagen, solo dar **OK** si es la primera vez que se exporta la imagen del mismo archivo seleccionado.

9.-Para finalizar el experimento, presionar el botón *Power* (Figura 6-24) de la interfaz, cerrarla, apagar el banco de pruebas y desconectarlo de la toma de corriente.



Figura 6-24 Botón Power

Actividad 6. ¿Se cumplen los casos de Sobre-amortiguado, sub-amortiguado y críticamente-amortiguado con los valores sugeridos en la actividad 5? ¿Por qué si? o ¿Por qué no? [El profesor puede ayudar en la construcción de conclusiones]

PARTE DOS (Esta segunda parte se recomienda hacerla en otra sesión)

Profesor

Actividad 1. Proponer un control PI a los alumnos y explicarlo.

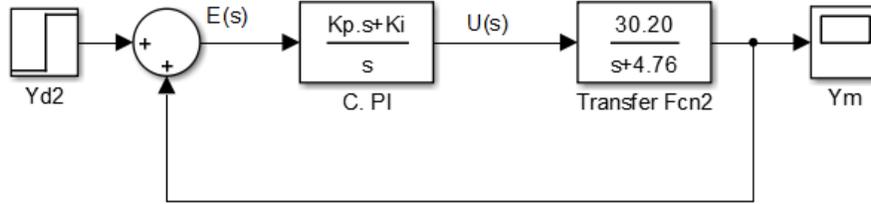


Figura 6-25 Diagrama de bloques control PI

Obtener las Funciones de transferencia de lazo cerrado $T(s)$ y lazo abierto $GH(s)$.

$$T3(s) = \frac{30.20 * (Kp * s + Ki)}{s^2 + (4.76 + 30.20 * Kp) * s + 30.20 * Ki}$$

$$GH3(s) = \frac{30.20 * (Kp * s + Ki)}{s * (s + 4.76)}$$

Estabilización.

- Por el criterio de Routh-Hurwitz o Newton, se comprueba que $T3(s)$ es estable cuando $Ki > 0$

$$4.76 + 30.20 * Kp > 0 \quad \rightarrow \quad Kp > \frac{-4.76}{30.20} \approx 0.1576$$

Regulación.

- $GH3(s)$ es de tipo 1 y orden 2. Explicar a los alumnos porqué.
- Entonces el error en estado permanente con una entrada escalón es cero $e_{\infty} = 0$.

Desempeño. Obtener polos de lazo cerrado deseados.

El desarrollo de la práctica se realizó con los siguientes valores deseados, pero quedan a consideración del profesor.

$$ts_{lcd} = 1.5, \quad \%SP_{lcd} = 15\%$$

$$ts_{lcd} = \frac{5}{|Re\{P_{lcd}\}|} \quad \%SP_{lcd} = e^{\left(\frac{-\pi * Re\{P_{lcd}\}}{Im\{P_{lcd}\}}\right)} * 100$$

$$|Re\{P_{lcd}\}| = \frac{5}{ts_{lcd}} = \frac{5}{1.5} = 3.3 \quad |Im\{P_{lcd}\}| = \frac{-\pi * Re\{P_{lcd}\}}{\ln\left(\frac{\%SP_{lcd}}{100}\right)} = \frac{-\pi[3.33]}{\ln\left(\frac{\%SP_{lcd}}{100}\right)} = 5.5199$$

- Polos de lazo cerrado deseados:

$$P_{lcd} = -3.33 \pm 5.5199j$$

- Obtener la ecuación característica deseada, con los polos calculados anteriormente.

$$[(s + 3.3) - (5.5199j)][(s + 3.3) + (5.5199j)] = (s + 3.33)^2 - (5.5199j)^2$$

$$s^2 + 6.6667 * s + 41.5808 = 0$$

- Con la ecuación característica de T3(s) ($s^2 + (4.76 + 30.20 * Kp) * s + 30.20 * Ki$) y la ecuación característica deseada ($s^2 + 6.6667 * s + 41.5808 = 0$), se obtienen los valores de las constantes Kp y Ki.

$$Kp = \frac{6.6667 - 4.76}{30.20} = 0.06313576$$

$$Ki = \frac{41.5808}{30.20} = 1.376847$$

Alumno

Actividad 1. Obtener los polos de lazo cerrado deseados para el control PI propuesto ($t_{s_{lcd}} = 1.5$, $\%SPlcd = 15\%$ Puede cambiar a consideración de profesor)

Actividad 2. Obtener la ecuación característica deseada, con los polos calculados en la actividad anterior.

Actividad 3. Con la ecuación característica de T3(s) y la ecuación característica deseada, obtener los valores de las constantes Kp y Ki para el control PI.

Actividad 4. Con Matlab Simulink construir el diagrama de bloques del control PI.

Actividad 5. Comprobar el comportamiento del control PI, tanto en Simulink como en el banco de pruebas, para un valor deseado de 60 [rpm] o 40 [rpm].

Inicio de experimento

En caso de no usar la arquitectura abierta del banco de prueba, omitir el paso 1.

1.- Conectar el cable DB15 al banco de pruebas, el microcontrolador; y este conectarlo a la computadora. Ver los pines de conexión del microcontrolador y el banco de pruebas; en el programa correspondiente a esta práctica.¹⁰

2.-Energizar el banco de pruebas.

3.- Abrir la Interfaz de LabVIEW "Interfaz_1". Cargar el programa "P5_DiseñoControladores" al microcontrolador que se encuentra en la capeta de códigos, encender el banco de pruebas (switch encendido).

4.- En la interfaz_1 llenar los cuadros de texto con la información solicitada (Ubicación de carpeta, nombre de archivo y puerto COM) y presionar el botón RUN. El led indicador del bluetooth, dejará de parpadear, lo que significa que la conexión se realizó correctamente.

5.- Presionar el botón Prueba, para empezar con el experimento y elegir la opción de Diseño de Ctrl

¹⁰ Ver conexiones del cable DB15 en *Descripción de tarjeta PCB PinOut* y las características del microcontrolador en *Descripción Material Electrónico*.

(Figura 6-22).

6.- Introducir los valores de las constantes K según sea el caso y el valor deseado. **Dar reset al sistema con el botón rojo del banco de prueba**, posteriormente presionar el botón Play que se encuentra en la interfaz_1, una vez terminada la secuencia (el motor dejó de moverse) presionar el botón Stop. Para análisis de la gráfica usar la herramienta Graph Palette.

7.-Limpiar la pantalla del *display* de la gráfica, dando clic en el botón *Reset Graph*. Para asegurar que la pantalla este limpia, repetir los pasos seis al siete si es necesaria otra prueba.

8.- Si se desea graficar algún experimento anterior; presionar el botón Graficar y seguir las instrucciones de la interfaz para exportar la imagen de la gráfica del experimento seleccionado. El programa por seguridad despliega un mensaje de sobre escritura al momento de exportar la imagen, solo dar **OK** si es la primera vez que se exporta la imagen del mismo archivo seleccionado.

9.-Para finalizar el experimento, presionar el botón *Power* (Figura 6-24) de la interfaz, cerrarla, apagar el banco de pruebas y desconectarlo de la toma de corriente.

Actividad 6. Escribe tus conclusiones, y ¿Cuál es el error entre el experimento en el Banco de Pruebas contra lo obtenido en las simulaciones, para el tiempo de asentamiento y el porcentaje de sobrepaso?

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO



FACULTAD DE INGENIERÍA



Práctica 6

6.6 “Comparación de Controladores”

Objetivos

- Verificar el funcionamiento de diferentes controladores (P, PI, PD y PID), así como identificar el efecto de cada uno de sus términos

Introducción

La ley de control es el algoritmo de cálculo de la variable de control ($u(t)$) de una planta que depende de la variable a controlar medida (Y_m) y deseada (Y_d).

Planta: Puede ser una parte de un equipo o el conjunto de algunas partes de una máquina que funcionan juntas, el propósito del cual es ejecutar una operación particular. En pocas palabras se puede considerar como el objeto o máquina que queremos controlar.

Variable a controlar: Es la cantidad física que se mide o que se quiere controlar.

Variable de control: Es la cantidad física que un controlador debe manipular con el fin de modificar la variable a controlar

Los controladores tipo PID utilizan la señal de error (e) en el tiempo $e(t)$, que es empleado en las siguientes leyes de control, donde el error es la diferencia entre el valor deseado (Y_d) y el valor medido (Y_m).

Tabla 6-4 Leyes de control (P, PI, PD y PID)

Ley de Control	Controlador
$u(t) = K_p e$	P
$u(t) = K_p e + K_i \int e dt$	PI
$u(t) = K_p e + K_d \frac{de}{dt}$	PD
$u(t) = K_p e + K_i \int e dt + K_d \frac{de}{dt}$	PID

Material o equipo a utilizar

- Banco de Pruebas.
- Computadora con la "Interfaz_1" instalada, y la IDE de ENERGIA; para el código de esta práctica.

Si la práctica se decide hacer con la arquitectura abierta

- Cable DB15
- Microcontrolador¹¹: Tiva C Series TM4C123G. EK-TM4C123GXL
- Cables tipo jumper mínimo 15 piezas (Hembra- hembra, hembra-macho)
- Protoboard

Desarrollo

Se sugiere prestar atención a las siguientes características: tiempo de asentamiento (ts), porcentaje de sobre-paso (%Sp) y error en estado permanente.

En caso de no usar la arquitectura abierta del banco de prueba, omitir el paso 1.

1.- Conectar el cable DB15 al banco de pruebas, el microcontrolador; y este conectarlo a la computadora. Ver los pines de conexión del microcontrolador y el banco de pruebas; en el programa correspondiente a esta práctica.¹²

2.-Energizar el banco de pruebas.

3.- Abrir la Interfaz de LabVIEW "Interfaz_1". Cargar el programa "P2_P3_P6" al microcontrolador que se encuentra en la capeta de códigos, encender el banco de pruebas (switch encendido).

4.- En la interfaz_1 llenar los cuadros de texto con la información solicitada (Ubicación de carpeta, nombre de archivo y puerto COM) y presionar el botón RUN. El led indicador del *bluetooth*, dejará de parpadear, lo que significa que la conexión se realizó correctamente.

5.-Presionar el botón Prueba, elegir la opción Comparación de Ctrl. Figura 6-27.

6.- Elegir uno de los controladores de la Figura 6-26. Se recomienda observarlos en el siguiente orden P, PI, PD o PID. Colocar un valor deseado (Figura 6-28) y mantenerlo fijo hasta que termine la

¹¹ El código proporcionado se realizó en la *IDE Energia* (para microcontroladores TI). Sin embargo, se puede utilizar cualquier otro microcontrolador, haciendo los cambios pertinentes de lenguaje, pero conservando la misma estructura de la cadena de datos que el microcontrolador envía a la computadora.

¹² Ver conexiones del cable DB15 en *Descripción de tarjeta PCB PinOut* y las características del microcontrolador en *Descripción Material Electrónico*.

práctica.

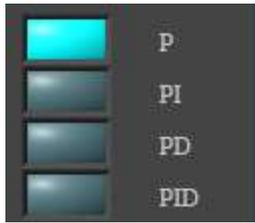


Figura 6-26 Selector de controladores



Figura 6-27 Botones para selección de prácticas



Figura 6-28 Perilla de Valor Deseado.

7.- Presionar el botón reset (**botón rojo del bando de pruebas**), acto seguido dar clic al botón Play que se encuentra en la interfaz_1, una vez terminada la secuencia (el motor dejó de moverse) presionar el botón Stop. Utilizar la herramienta *Graph Palette* si se desea ver detalles o ajustar la gráfica. La interfaz cuenta con un botón **Reset Graph** para limpiar o refrescar la pantalla donde se despliegan las gráficas, usar esta opción para no saturar la pantalla y evitar que las gráficas se empalmen dificultando su análisis.



Figura 6-29 Graph Palette.

8.- Para ver el efecto de un nuevo controlador repetir los pasos seis y siete, hasta que se hayan visto los cuatro tipos de controladores. Si se desea graficar algún experimento anterior; presionar el botón **Graficar** y seguir las instrucciones de la interfaz para exportar la imagen de la gráfica del experimento seleccionado. El programa por seguridad despliega un mensaje de sobre escritura al momento de exportar la imagen, solo dar **OK** si es la primera vez que se exporta la imagen del mismo archivo seleccionado.

9.-Elegir uno de los cuatro controladores, agregar un disco de carga (Figura 6-30). Presionar el botón reset (**botón rojo del bando de pruebas**), acto seguido dar clic al botón Play que se encuentra en la interfaz_1, una vez terminada la secuencia (el motor deajo de moverse) presionar el botón Stop.

Se recomienda repetir el paso, agregando un nuevo disco de carga.

10.-Para finalizar el experimento, presionar el botón *Power* de la interfaz de LabVIEW, cerrar el programa, apagar el banco de pruebas y quitar los discos de carga.



Figura 6-30 Discos de carga de 1 Kg

Preguntas.

En qué controlador se pudieron apreciar los siguientes efectos. La comparación se realiza con respecto al control P.

- Disminución del error en estado permanente
- Disminución del %Sp
- El t_s se alcanzó en menor tiempo

Del controlador que se le agregaron discos de carga. ¿Conserva o mejora su comportamiento al agregar más peso en cada iteración?

Opcional.

Del código de programación que viene incluido para esta práctica, indicar que sección o secciones están dedicadas para el cálculo de cada componente del PID (Ley de control).

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO



FACULTAD DE INGENIERÍA



Práctica 7

6.7 “Sintonización de Controladores PID, Método Ziegler-Nichols (Oscilaciones Continuas)”

Objetivos

- Comprender el uso del segundo método de Ziegler-Nichols para diseño de controladores.
- Diseñar un controlador PID, PI, PD o P, por medio de dicho método

Introducción

Generalmente, la sintonización de los controladores se realiza empleando métodos analíticos con el modelo matemático de la planta o proceso a controlar. Sin embargo, suele presentarse el caso en el que no es posible obtener el modelo matemático del proceso a controlar debido a la complejidad de este. Existen alternativas para solucionar este tipo de problema. Tal como el uso de métodos empíricos, siendo los métodos de Ziegler-Nichols los más utilizados.

En los métodos empíricos se requiere la obtención de los parámetros: ganancia proporcional (K_p), tiempo integral (T_i) y tiempo derivativo (T_d). Posteriormente, estos parámetros son sustituidos en la función de transferencia estándar para el controlador PID de los métodos de Ziegler-Nichols.

La función de transferencia estándar para el diseño de controladores PID es la siguiente:

$$G_{PID} = G_C = k_p \left[1 + \frac{1}{T_i s} + T_d s \right]$$

Existen dos métodos de Ziegler-Nichols para el diseño de controladores:

- 1.- Método basado en la respuesta al escalón.
- 2.- Oscilaciones Continuas.

Algunos otros métodos para el diseño de controladores son [1], [9]:

- Cohen y Coon
- Kaya y Sheib

Material o equipo a utilizar

- Banco de Pruebas.
- Computadora con la "Interfaz_1" instalada, y la IDE de ENERGIA; para el código de esta práctica.

Si la práctica se decide hacer con la arquitectura abierta

- Cable DB15
- Microcontrolador¹³: Tiva C Series TM4C123G. EK-TM4C123GXL
- Cables tipo jumper mínimo 15 piezas (Hembra- hembra, hembra-macho)
- Protoboard

Desarrollo

Profesor

Actividad 1. Explicar a los alumnos el método de *Oscilaciones Continuas* (segundo método)] de Ziegler-Nichols para diseño de controladores. Permitiendo así a los alumnos se les facilite la comprensión del desarrollo de esta práctica para que diseñen al menos un controlador.

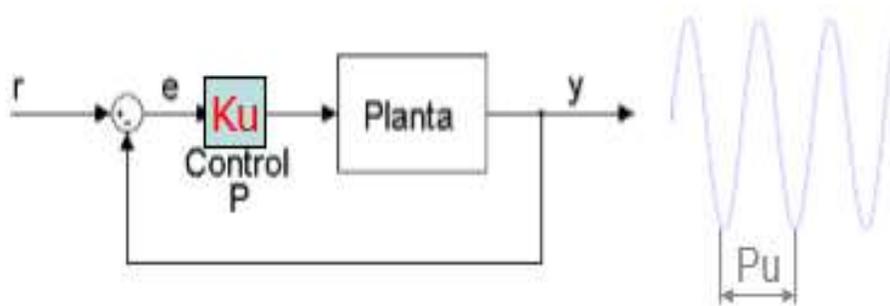


Figura 6-31 Diagrama de bloques de control P, segundo método de Ziegler Nichols

¹³ El código proporcionado se realizó en la *IDE Energia* (para microcontroladores TI). Sin embargo, se puede utilizar cualquier otro microcontrolador, haciendo los cambios pertinentes de lenguaje, pero conservando la misma estructura de la cadena de datos que el microcontrolador envía a la computadora.

Tabla 6-5 Cálculo de constantes para los tres tipos de controladores.

CONTROL	KP	TI	TD
P	0.5Ku	-	-
PI	0.45Ku	(1/1.2)Pu	-
PID	0.6Ku	(1/2)Pu	(1/8)Pu

Nota 1: La Tabla 6-5 es una sugerencia, se puede utilizar la tabla del autor de su preferencia.

Alumno

Actividad 1. Diseñe al menos un controlador, utilizando el método de *Oscilaciones Continuas* de Ziegler-Nichols. Para ello, realice lo siguiente.

Iniciar experimento.

En caso de no usar la arquitectura abierta del banco de prueba, omitir el paso 1.

- 1.- Conectar el cable DB15 al banco de pruebas, el microcontrolador; y este conectarlo a la computadora. Ver los pines de conexión del microcontrolador y el banco de pruebas; en el programa correspondiente a esta práctica.¹⁴
- 2.-Energizar el banco de pruebas.
- 3.- Abrir la Interfaz de LabVIEW “Interfaz_1”. Cargar el programa “Ziegler_Nichols” al microcontrolador.
- 4.- En la interfaz_1 llenar los cuadros de texto con la información solicitada (Ubicación de carpeta, nombre de archivo y puerto COM) y presionar el botón RUN. El led indicador del *bluetooth*, dejará de parpadear, lo que significa que la conexión se realizó correctamente.
- 5.- Presionar el botón Prueba, elegir la opción Ziegler Nichols (Figura 6-33). Posteriormente elegir la opción Posición o Velocidad Figura 6-32. Según la planta que se desee observar el controlador.
- 6.- Ingresar los valores de Kp, Ki, Kd y valor deseado (el orden de ingreso es irrelevante).



Figura 6-33 Botones para selección de prácticas Figura 6-32 Velocidad/Posición

¹⁴ Ver conexiones del cable DB15 en *Descripción de tarjeta PCB PinOut* y las características del microcontrolador en *Descripción Material Electrónico*.

Los valores de K_i y K_d son igual a cero hasta que se indique lo contrario. Presionar el botón Play que se encuentra en la interfaz_1, una vez terminada la secuencia presionar el botón Stop. **Presionar el botón rojo del banco de pruebas.**

7.- Limpiar la pantalla del *display* de la gráfica, dando clic en el botón *Reset Graph*. Para asegurar que la pantalla este limpia.

8.- Repetir el paso seis y siete aumentando el valor de la constante K_p en cada iteración hasta que el sistema muestre un comportamiento oscilante. El aumento de la K_p es gradual, se sugiere empezar con incrementos de una unidad.

9.- Una vez que el sistema presente oscilaciones continuas, con ayuda de las herramientas de *Graph Palette* obtener el periodo de PU.



Figura 6-34 Graph Palette

10.- Al obtener el periodo (P_u), y la constante K_u (la última K_p ingresada), utilizar la función de transferencia estándar del PID y la Tabla 6-5 para calcular las constantes K_p , K_i y K_d , del controlador a diseñar (P, PI, PD, PID), dependiendo el criterio del profesor. Recordar que el tiempo de la gráfica se encuentra en milisegundos, para efectos de los cálculos usar el tiempo en segundos.

11.- Una vez obtenidos los valores K_p , K_i y K_d , repetir los pasos seis y siete.

12.- Los valores de las constantes se pueden variar para mejorar la respuesta del sistema, dependiendo del criterio del profesor.

13.- Si se desea graficar algún experimento anterior; presionar el botón Graficar y seguir las instrucciones de la interfaz para exportar la imagen de la gráfica del experimento seleccionado.

El programa por seguridad despliega un mensaje de sobre escritura al momento de exportar la imagen, solo dar **OK** si es la primera vez que se exporta la imagen del mismo archivo seleccionado, para más imágenes de un mismo archivo, se recomienda cambiar el nombre de las imágenes previas para no sobre escribirlas

14.-Para finalizar el experimento, presionar el botón *Power* (Figura 6-35) de la interfaz, cerrarla y apagar el banco de pruebas y desconectarlo de la toma de corriente.



Figura 6-35 Botón Power

Actividad 2. Hacer un reporte de las gráficas obtenidas, y dar una conclusión del método con base en las gráficas obtenidas.

7 Conclusiones

Se logró elaborar un total de siete prácticas como material didáctico para algunos temas de control automático que se cree que complementarán los conocimientos y habilidades del usuario (alumno/profesor) al realizar experimentos con el motor del banco de pruebas.

Se fomenta el uso de software MATLAB como herramienta que ayuda al análisis de sistemas mediante simulación (Contar con licencia oficial o de prueba), para verificar y comprobar si el sistema planteado para el motor de forma experimental cumple o se acerca con lo descrito teóricamente. Como ejemplo se tiene la práctica de *Diseño de Controladores* donde al analizar el sistema para un control I, se tienen tres casos sub-amortiguado, sobre-amortiguado y críticamente amortiguado, cada caso se realiza en el banco de pruebas, y para corroborar las respuestas se usó MATLAB Simulink, como se observa con la Figura 7-1 y Figura 7-2.

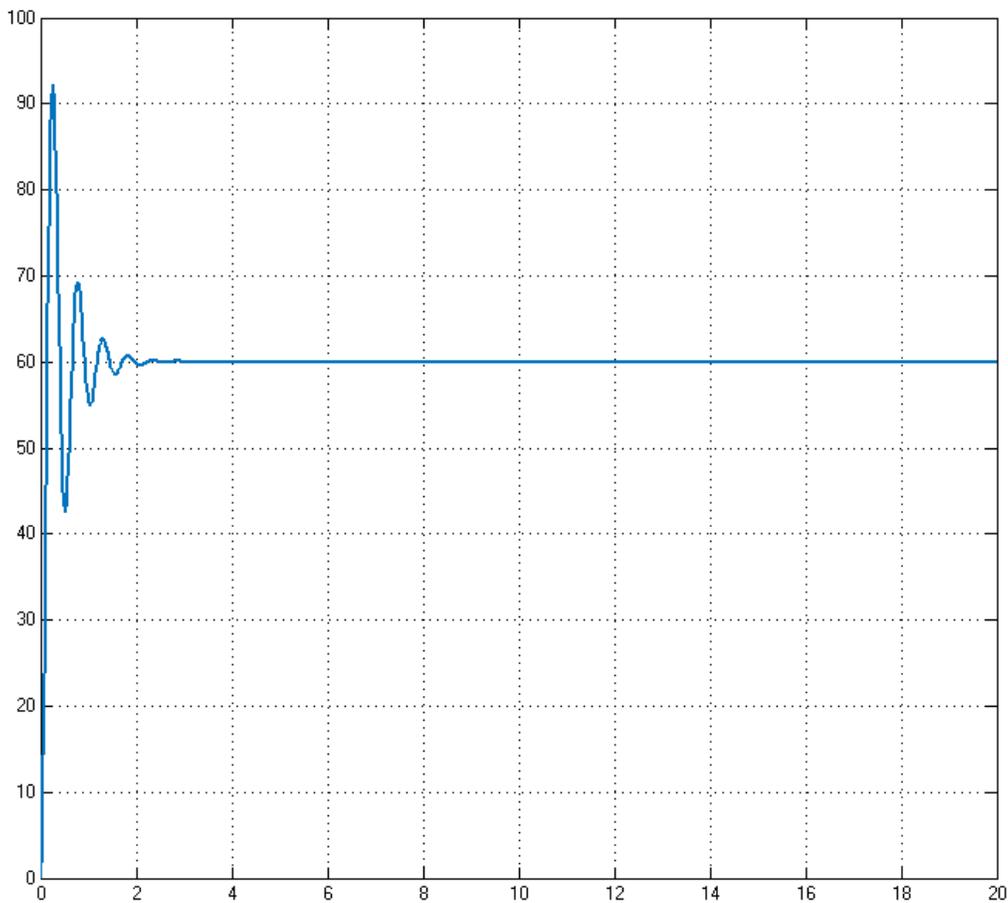


Figura 7-1 Control I ($K_i=5$) Simulación

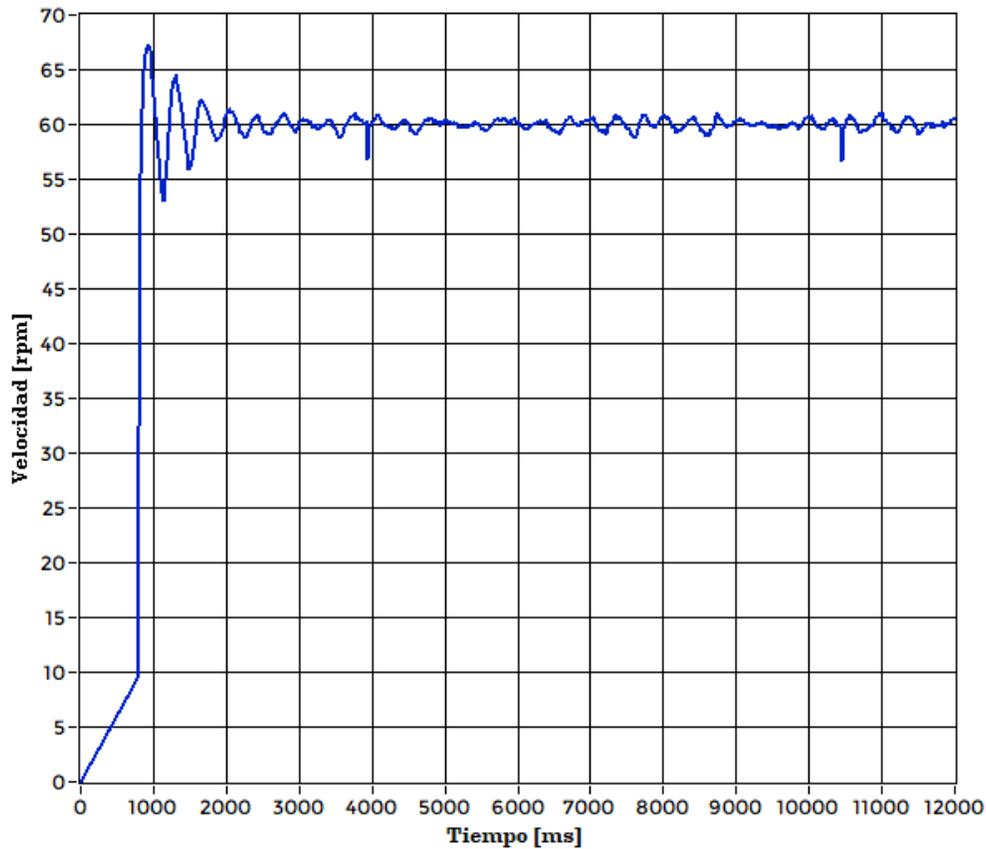


Figura 7-2 Control I ($K_i=5$) Experimental

En ambos controles (experimental y simulación) se presentan oscilaciones; es así que se verifica que el sistema planteado es sub-amortiguado, sin ser estrictos para los dos casos el tiempo en que llegan al valor deseado es de aproximadamente 2 segundos (En el caso del experimental el tiempo se mide en milisegundos).

La creación de las interfaces de usuario, facilita el análisis de los datos; porque son desplegados de forma gráfica. Cuenta con una herramienta para el análisis de la gráfica y así obtener datos como tiempo, posición o velocidad que dependen de la práctica a realizar; dichos datos se aplicarán en el desarrollo de la práctica para resolver o aclarar problemas que se planteen, tanto por el profesor como por la misma práctica. Además de poder guardar los datos en un archivo, para ser revisados o analizados en otro momento.

El uso del sistema no solo se queda en la materia de control automático, ya que existen temas que

tienen relación directa o indirecta con otras asignaturas. Entonces las prácticas pueden ser aprovechadas según lo vea conveniente el profesor que desee usar el banco de pruebas, por ejemplo en la materia de Modelado de Sistemas también es posible usar la práctica Modelado Matemático, o en la materia de Diseño Mecatrónico con la práctica diseño de controladores o en general cualquier práctica en la que pueda implementar conocimientos del tema de diseño de software o para ejemplificar los componentes de un sistema mecatrónico.

Además si las prácticas planteadas no son suficientes para alguna materia, existe la posibilidad de usar la arquitectura abierta, con la que se pueden crear mayor número de prácticas, y para diversas asignaturas, porque se puede disponer de las señales provenientes del sistema y poder enviar señales de control, según los objetivos del usuario, sin tener la limitación de las prácticas ya planteadas.

Pienso que la implementación y creación de bancos de pruebas para complementar y dar apoyo didáctico en las carreras de ingeniería que incluyan una o varias asignaturas como Control, es algo en lo cual invertir de forma educativa. En el trabajo de Salgado, O. Q [11] en el tema 1.2 menciona diferentes antecedentes de bancos de pruebas con motores dc, que muestra la importancia del desarrollo de este tipo de sistemas.

En una oportunidad que tuve de visitar la Tecnológico de Estudios Superiores de Chimalhuacán (TESCHI), en la exposición de proyectos de la carrera de Ingeniería Mecatrónica, los profesores comentaron la necesidad de “Bancos de pruebas con motores de corriente directa” para complementar sus clases de control. Es así que creo que el presente manual de prácticas será de gran utilidad para los estudiantes y profesores que le den uso.

8 Fuentes de consulta

[1] Alfaro R, Víctor M. (2002). MÉTODO SDE SINTONIZACIÓN DE CONTROLADORES PID QUE OPERAN COMO REGULADORES. *Ingeniería*, 12 (1,2), 21-36.

[2] Anaheim Automation. *Encoder Guide*. Recuperado de <<<http://www.anaheimautomation.com/manuals/forms/encoder-guide.php>>> el 19 octubre de 2016

[3] Arduino PID - Guía de uso de la librería. Recuperado de <<<http://brettbeauregard.com/blog/wp-content/uploads/2012/07/Gu%C3%ADa-de-uso-PID-para-Arduino.pdf>>>

[4] Barrientos Antonio., Sanz R., Matía F., Gambao E. (1996) *Control de Sistemas Continuos, Problemas resueltos*. McGraw-Hill.

[5] Dulhoste, Jean F. *Teoría de Control, Tema 10 Ajuste de Controladores PID*. Escuela de Ingeniería Mecánica-ULA. Recuperado de <<http://webdelprofesor.ula.ve/ingenieria/djean/index_archivos/Documentos/TC10_Ajuste_Controladores.pdf>>, el 25 de octubre de 2016

[6] Nómadas Electrónicos. *Encoder de cuadratura*. Recuperado de <<<https://nomadaselectronicos.wordpress.com/2013/02/01/encoder-de-cuadratura/>>> el 19 de octubre de 2016

[7] Norman S. Nice. (2006). *Sistemas de Control para Ingeniería*. México. Continental

[8] Ogata Katsuhiko. (2010). *Modern control engineering*, fifth edition, Pearson Prentice Hall

[9] Reglas para la sintonización de controladores PID. Recuperado de <<<http://prof.usb.ve/montbrun/Sintonizacion%20de%20Controladores.pdf>>>, el 23 noviembre de 2016

[10] Ruge, R. Ilber A. *Método básico para implementar un controlador digital pid en un controlador pic para desarrollo d aplicaciones a bajo costo*. Universidad de Cundinamarca. Recuperado de <<http://www.edutecne.utn.edu.ar/microcontrol_congr/industria/MTODOB~1.PDF>>

[11] Salgado, O. Q. (2015). *DESARROLLO DE DOS PROTOTIPOS DIDÁCTICOS PARA PRÁCTICAS DE CONTROL AUTOMÁTICO EN MOTORES DE CORRIENTE DIRECTA*. Tesis de licenciatura, UNAM Ciudad Universitaria, México, D.F.

[12] Venegas Javier. *Encoders*. Mayo 2009. Recuperado de <<<http://ramos.elo.utfsm.cl/~elo212/docs/Encoders-jvr-v01.pdf>>> el 19 de octubre de 2016

[13] West INSTRUMENTS DE MEXICO, *Manual de Aplicación de Encoders*,
<<[http://www.westmexico.com.mx/pfd/dynapar/catalogos/4.-
Manual%20de%20Aplicacion%20de%20Encoders.pdf](http://www.westmexico.com.mx/pfd/dynapar/catalogos/4.-Manual%20de%20Aplicacion%20de%20Encoders.pdf)>> el 20 de octubre de 2016

9 Anexos Generales.

9.1 Códigos de programación

Se realizaron cuatro diferentes códigos para ser empleados en las siete prácticas de este manual, elaborados en ENERGIA IDE. Todos los programas tienen la siguiente estructura.

1. Variables.
2. Configuraciones (*void setup*.)
3. Programa principal que se repite cíclicamente (*void loop*).
4. Sub programas o funciones.

Tanto las variables como algunas funciones se repiten en los cuatro programas, solo se mencionarán una única vez, aunque no todos los programas ocupen en su totalidad todas las variables o subprogramas.

Se coloca un programa principal como base, y posteriormente mostrando las variaciones de los otros tres programas.

9.1.1 Variables.

```
//-----Constantes K's-----
double kp=0.00;
double ki=0.00;
double kd=0.00;
//-----Indicadores visuales LED's-----
int blue=40;      //El led azul está en el pin 40
int red=30;       //El led azul está en el pin 30
int green=39;     //El led azul está en el pin 39

//-----Variables auxiliares-----
char bandera='S'; // l =inicio S=stop Letra de inicio para los comandos
char cierre;      //Variable que ayuda a limpiar el buffer, para evitar errores de comunicacion
int retardo=90;   //Retardo para la lectura de los datos en la funcion Limpieza //90 es el mejor tiempo
hasta el 30/4/2018
int Max=255;      //limites máximo del PWM para evitar el wind up
int Min=-255;     //limites mínimo del PWM para evitar el wind up
int botonA=36;    //Pin para el botón azul, al presionar el boton el motor gira para
                  //ser ubicado en el origen, se recomienda hacer reset en el microcontrolador
                  //para que tanto visualmente como en programa se establezca el cero
int giro=1;       //Estado deseado del pin botonA

//-----Variables para el filtro EMA (velocidad)-----
double filter=0.0; //variable para el filtro (EMA) exponential moving average,
float alpha=9.0179; // YM=(vel + 9.0176*YM)/10.002; --> YM=(vel + alpha*YM)/beta;
float beta=10.0;   // -->YM=(vel*alpha1 + beta2*YM); version extendida
```

```

//-----Variables para el uso del encoder (Codificador)-----
int encoderPin1 = 2; //Pin de la tarjeta dedicado para la lectura de los pulsos del encoder
int encoderPin2 = 3; //Pin de la tarjeta dedicado para la lectura de los pulsos del encoder
int lastMSB = 0; //Variable para el valor más significativo en la suma de los pulsos del encoder
int lastLSB = 0; //Variable para el valor menos significativo en la suma de los pulsos del encoder
volatile int lastEncoded = 0; //Variable que almacena el valor previo de la suma de los pulsos
volatile long encoderValue = 0; //Variable que entrega la cuenta de los pulsos, esta variable se puede usar el
cálculo de la posición

//-----Variables para el cálculo de posición-----
//Las constantes se pueden cambiar según el banco de pruebas "pequeño" o "grande" que se utilice //revisar
la descripción del material
float cpr = 64.0; //cpr = counts per revolution, resolución del encoder utilizado
float kred = 131.25; //kred = factor de reducción del motor utilizado

//-----Variables para el cálculo de velocidad-----
double vel; //Variable que guarda el valor de la velocidad
double rev; //Variable en la que se guarda la cuenta de las revoluciones del motor
double vfil; //variable en la que se guardara al velocidad filtrada
double revold = 0; //Valor previo de la cuenta de las revoluciones
unsigned long Tv=0.0; //tiempo de muestreo para el cálculo de velocidad, el tiempo está en segundos
unsigned long ahora2,antes2; //Variables para obtener la diferencia Tv

//-----Variables para el cálculo del error-----
double YM; //Valor medido posición o velocidad
double YD=0; //Setpoint o valor de deseado (posición o velocidad)
double error = 0; //Error calculado, valor deseado menos valor medido
double error_pre = 0.0; //Valor previo de la variable error
double errorSuma = 0; //Suma del error (integral)
double derror = 0; //Derivada del error
int contador1=0; //Variable destinada para que en la primer iteración el error previo sea el valor
deseado

//-----Variables para el motor-----
int motorA =4; //Pin A para giro el motor
int motorB =5; //Pin B para giro del motor
int pwm =23; //Pin para la señal U de la ley de control

//-----Variables relacionadas con el tiempo-----
unsigned long ahora, antes; //Variables para determinar el intervalo de tiempo de cada iteración
long dt = 0; //Diferencial de tiempo, ahora menos antes
long DT=20.0; //20 //Intervalo de tiempo con el que se ejecutara la función PID
double tiempo;
double TMseg = 0.0; //Tiempo de Muestreo en segundos

//Señales ds Salida del Sistema de la funcion de transferencia del controlador (U=P+I+D)
int U;
int u1,u2;

```

9.1.2 Configuraciones.

```
void setup()
{
  Serial.begin(115200);// Inicializar la comunicación Serie

  //Configuración de los pines que se utilizan para medir las señales del encoder
  pinMode(encoderPin1, INPUT);
  pinMode(encoderPin2, INPUT);
  digitalWrite(encoderPin1, HIGH); //turn pullup resistor on
  digitalWrite(encoderPin2, HIGH); //turn pullup resistor on
  attachInterrupt(encoderPin1, updateEncoder, CHANGE); //Configuración para interrupción
  attachInterrupt(encoderPin2, updateEncoder, CHANGE); //Configuración para interrupción

  //Configuraciones del LED rgb
  pinMode(blue,OUTPUT); //Configurar el led azul
  pinMode(red,OUTPUT); //Configura led rojo
  pinMode(green,OUTPUT); //Configura led verde

  //Configuraciones de los pines destinados a dar dirección al motor
  pinMode(motorA,OUTPUT); //Configuración de pines para la dirección del motor
  pinMode(motorB,OUTPUT); //Configuración de pines para la dirección del motor
  digitalWrite(motorA, LOW); //turn pullup resistor on
  digitalWrite(motorB, LOW); //turn pullup resistor on
  pinMode(botonA,INPUT);

  //Parpadeo del LED rojo, para indicar que el void setup ha terminado de ejecutarse
  digitalWrite(red,HIGH);
  delay(50);
  digitalWrite(red,LOW);
  delay(100);
  digitalWrite(red,HIGH);
  delay(50);
  digitalWrite(red,LOW);
  delay(100);
  digitalWrite(red,HIGH);
  delay(50);
  digitalWrite(red,LOW);
  delay(100);
  digitalWrite(red,HIGH);
  delay(50);
  digitalWrite(red,LOW);
}
```

9.1.3 Programa principal.

Código P2_P3_P6

```
//Inicio del ciclo de trabajo del microcontrolador
void loop()
{
  /* Verifica si hay 38 datos en el buffer, después leer el primer carácter para entrar al IF
  * y asegurar un estado para la variable bandera, En el caso verdadero se ejecuta el controlador
  * en el caso falso, se detiene el motor.
  */
  if(Serial.available()>38) //S010.000000,020.000000,030.000000,0047 //38 caracteres o datos
  {
    //I010.000000,000.000000,000.000000,0050 ejemplos de comandos recibidos

    rgb(1,0,0); //El led rojo se prende para indicar que se están recibiendo datos
    char inicio=Serial.read();
    if(inicio=='I')
    {
      InDatos(); //Se guardan los valores en sus respectivas variables
      bandera='I'; //Se asegura el estado
    }
    if(inicio=='S' || inicio=='s')
    {
      InDatos(); //Se guardan los valores en sus respectivas variables
      bandera='S'; //Se asegura el estado
    }
    Limpieza(); //Función destinada a borrar todo lo que puede quedar en el buffer y así evitar
                //conflictos con LabVIEW respecto a la interpretación de los comandos
  }
  switch(bandera)
  {
  case 'I':
    rgb(0,0,1);
    ahora2=micros();
    Tv=ahora2-antes2;
    YM=(double)encoderValue * 360 / (cpr * kred); //se calcula la posición en ángulos
    if((tiempo/1000.0)>=20.0) bandera='S'; //La ejecución del ciclo solo durará 20 segundos
    ahora=millis(); //Empezar a contar tiempo transcurrido en desde que se encendió el
                    //microcontrolador
    dt=ahora-antes; // Se obtiene la delta de tiempo, que ayudara para que la ejecución
                    //del PID en intervalos DT iguales
    if(dt>=DT) //verificar si el tiempo para la ejecución del PID se ha cumplido y
                //ejecutarlo, en caso contrario no ejecutar y continuar con el programa
    {
      //int PID_1(kp,ki,kd,valor deseado,valor medido) descripción de la función
      U=PID_1(kp,ki,kd,YD,YM); //Ejecución de la función PID_1 que entrega el valor voltaje
                              //para alimentar el motor,
      analogWrite(pwm,abs(U)); //Activación del motor analogWrite(pwm,U)
      revold=rev; //Guardar el valor de revolución pasado en revold
      antes=ahora; //Actualización del tiempo anterior para el intervalo de ejecución del PID
      tiempo+=dt;
      antes2=ahora2; //Actualización del tiempo anterior para la medición de la velocidad
    }
  }
}
```

```

} // 00255.00,00000.00,00000.00/n Ejemplo de cadena enviada a la computadora

break;//-----FIN case I
case 'S':
//Se reinician variables
kp=0.00;
ki=0.00;
kd=0.00;
YD=0.00;
error=0.0;
contador1=0;
errorSuma=0;
derror=0;
u2=0;
tiempo=0;
rgb(0,1,0); //Prender led verde
giro=digitalRead(botonA);
if(giro==0) //gira el motor al presionar el botón azul
{
  digitalWrite(motorA,HIGH);
  digitalWrite(motorB,LOW);
  analogWrite(pwm,1);
  delayMicroseconds(40);
}
else
{
  analogWrite(pwm,0);
}
break;//-----FIN case S
}
} //Fin del loop

```

Código P4_ModeladoMatematico

```

void loop()
{
  ahora=micros();
  dt=ahora-antes;
  if(Serial.available(>=3) //S010.000000,020.000000,030.000000,0047 //38 caracteres o datos
  {
    rgb(1,0,0); //El led rojo se prende para indicar que se están recibiendo datos
    char inicio=Serial.read();
    if(inicio=='I')
    {
      InDatos(); // Se guardan los valores en sus respectivas variables
      bandera='I'; //Se asegura el estado
      tiempo=0;
    }
    if(inicio=='S' || inicio=='s')
    {
      InDatos(); // Se guardan los valores en sus respectivas variables
      bandera='S'; //Se asegura el estado
    }
  }
}

```

```

    }
    Limpieza();
}

switch(bandera)
{
case 'I':
    rgb(0,0,1);
    ahora2=micros();
    Tv=ahora2-antes2;
    rev=(double)encoderValue/(cpr*kred); //se calculan las revoluciones que ha dado el motor
    vel=((rev-revold)*60000000.0)/Tv;
    YM=FiltroEMA(vel);
    if((tiempo/1000000.0)>=10.0) bandera='S';
    digitalWrite(motorA, HIGH);
    digitalWrite(motorB, LOW);
    analogWrite(pwm,255);
    agregar_ceros(YM);
    Serial.print(YM);
    Serial.print(',');
    agregar_ceros((tiempo/1000.0));
    Serial.print((tiempo/1000.0));
    Serial.println();
    revold=rev; //Guardar el valor de revolución pasado en revold
    antes=ahora; //Actualización del tiempo anterior para el intervalo de ejecución del PID
    antes2=ahora2; //Actualización del tiempo anterior para la medición de la velocidad
    tiempo+=dt;
    // }
break;//-----FIN case I
case 'S':
    rgb(0,1,0); //Prender led verde
    analogWrite(pwm,0);
    break;//-----FIN case S
} //fin switch
} //Fin del loop

```

Código P5_DiseñoControladores

```

void loop()
{
    if(Serial.available(>38)
    {
        rgb(1,0,0);
        char inicio=Serial.read();
        if(inicio=='C')
        {
            InDatos(); // Se guardan los valores en sus respectivas variables
            bandera='C'; //Se asegura el estado
            tiempo=0;
        }
        if(inicio=='S' || inicio=='s')
        {

```

```

    InDatos();      // Se guardan los valores en sus respectivas variables
    bandera='S';    //Se asegura el estado
}
Limpieza();
}
switch(bandera)
{
case 'C':
    rgb(0,0,1);
    ahora2=micros();
    Tv=ahora2-antes2;
    rev=(double)encoderValue/(cpr*kred);
    vel=((rev-revold)*60000000.0)/Tv;
    YM=FiltroEMA(vel);
    if((tiempo/1000.0)>=15.0) bandera='S'; //La ejecución del ciclo solo durara 15 segundos
    ahora=millis();
    dt=ahora-antes;
    if(dt>=DT)
    {
        U=PID_1(kp,ki,kd,YD,YM);      //int PID_1(kp,ki,kd,valor deseado,valor medido)
        digitalWrite(motorA, HIGH);
        digitalWrite(motorB, LOW);
        analogWrite(pwm,abs(U));      //Activación del motor analogWrite(pwm,U)
        revold=rev;                  //Guardar el valor de revolución pasado en revold
        antes=ahora;
        tiempo+=dt;
        antes2=ahora2;              //Actualización del tiempo anterior para la medición de la velocidad
    }
    break;//-----FIN case I
case 'S':
    kp=0.00;
    ki=0.00;
    kd=0.00;
    YD=0.00;
    error=0.0;
    contador1=0;
    errorSuma=0;
    derror=0;
    u2=0;
    tiempo=0;
    YM=0.0;
    rgb(0,1,0); //Prender led verde
    encoderValue=0;
    analogWrite(pwm,0);
    break;//-----FIN case S
}
} //Fin del loop

```

Código P7_Ziegler_Nichols

```

void loop()
{

```

```

if(Serial.available(>38) //S010.000000,020.000000,030.000000,0047 //38 caracteres o datos
{
  rgb(1,0,0); //El led rojo se prende para indicar que se están recibiendo datos
  char inicio=Serial.read();
  if(inicio=='I')
  {
    InDatos(); // Se guardan los valores en sus respectivas variables
    bandera='I'; //Se asegura el estado
    tiempo=0;
  }
  if(inicio=='S' || inicio=='s')
  {
    InDatos(); // Se guardan los valores en sus respectivas variables
    bandera='S'; //Se asegura el estado
  }
  if(inicio=='c' || inicio=='C')
  {
    InDatos(); // Se guardan los valores en sus respectivas variables
    bandera='C'; //Se asegura el estado
    tiempo=0;
  }
  Limpieza();
}

switch(bandera)
{
case 'I': //Este caso solo sirve para controlar la posición
  rgb(0,0,1);
  YM=(double)encoderValue*360.0/(cpr * kred); //se calcula la posición en ángulos
  if((tiempo/1000.0)>=10.0) bandera='S'; //La ejecución del ciclo solo durará 60 segundos
  ahora=millis();
  dt=ahora-antes;
  if(dt>=DT)
  {
    //int PID_1(kp,ki,kd,valor deseado,valor medido)
    U=PID_1(kp,ki,kd,YD,YM,Max,Min);
    analogWrite(pwm,abs(U)); //Activacion del motor analogWrite(pwm,U)
    //revold=rev; //Guardar el valor de revolución pasado en revold
    antes=ahora;
    tiempo+=dt;
  }
  break;//-----FIN case I
case 'C': //Caso en el que se controla la velocidad.
  rgb(1,1,1);
  ahora2=millis();
  Tv=ahora2-antes2;
  rev=(double)encoderValue/(cpr*kred); //se calculan las revoluciones que ha dado el motor
  vel=(((rev-revold)*60000000.0)/Tv);
  YM=FiltroEMA(vel);
  if((tiempo/1000.0)>=10.0) bandera='S'; //La ejecución del ciclo solo durará 60 segundos
  ahora=millis();
}

```

```

dt=ahora-antes;
if(dt>=DT)
{
//int PID_1(kp,ki,kd,valor deseado,valor medido, umbral superior, umbral inferior)
U=PID_1(kp,ki,kd,YD,YM,Max,MinV);
digitalWrite(motorA, HIGH);
digitalWrite(motorB, LOW);
analogWrite(pwm,abs(U)); //Activacion del motor analogWrite(pwm,U)
revold=rev; //Guardar el valor de revolución pasado en revold
antes=ahora;
tiempo+=dt;
antes2=ahora2; //Actualización del tiempo anterior para la medición de la velocidad
}

break;//-----FIN case C

case 'S': //Caso default se trata de que todas las variables vuelvan a su estado inicial, para asegurar que todo
vuelva a cero reiniciar el microcontrolador
kp=0.00;
ki=0.00;
kd=0.00;
YD=0.00;
error=0.0;
contador1=0;
errorSuma=0;
derror=0;
u1=0;
tiempo=0;
encoderValue=0;
vel=0.0;
rev=0.0;
revold=0.0;
rgb(0,1,0); //Prender led verde
analogWrite(pwm,0);
break;//-----FIN case S
}
} //Fin del loop

```

9.1.4 Funciones.

```

//-----
//-----Actualización del encoder-----
//-----
void updateEncoder()
{
int MSB = digitalRead(encoderPin1); //MSB = most significant bit
int LSB = digitalRead(encoderPin2); //LSB = least significant bit
int encoded = (MSB << 1) | LSB; //converting the 2 pin value to single number
int sum = (lastEncoded << 2) | encoded; //adding it to the previous encoded value

if(sum == 0b1101 || sum == 0b0100 || sum == 0b0010 || sum == 0b1011) encoderValue ++;

```

```

if(sum == 0b1110 || sum == 0b0111 || sum == 0b0001 || sum == 0b1000) encoderValue --;
lastEncoded = encoded; //store this value for next time
}

//-----
//-----Función que entrega la velocidad filtrada-----
//-----Filtro EMA Low Pass-----
//-----
double FiltroEMA(double ValM)
{
    filter = (ValM+alpha*filter)/beta; // YM=(vel + 9.0176*YM)/10.002;
    return filter;
}

//-----
//-----Función que agrega ceros a los valores a enviar-----
//----- y así enviar el mismo tamaño de palabra -----
//-----
void agregar_ceros(double lectura)
{
    if(lectura==0){
        Serial.print("0000");
    }
    if (lectura>0)
    { lectura=lectura*1;
      if(lectura<10000)
      {
          Serial.print('0');
          if(lectura<1000)
          {
              // el valor leído es menor a mil
              Serial.print('0'); //se agrega un cero
              if(lectura<100)
              {
                  //el valor leído ahora es menor que cien
                  Serial.print('0'); //se agrega un nuevo cero
                  if(lectura<10)
                  {
                      //finalmente el valor es menor que 10
                      Serial.print('0'); // se agrega un cero, en este punto al valor se le agregan tres ceros
                      //antes de enviarse por el puerto serie
                  }
              }
          }
      }
    }
}

if(lectura<0)
{ lectura=lectura*-1;
  if(lectura<10000)
  {
      Serial.print('-');
      if(lectura<1000)
      {
          // el valor leído es menor a mil
          Serial.print('0'); //se agrega un cero
      }
    }
}

```

```

    if(lectura<100)
    {
        //el valor leído ahora es menor que cien
        Serial.print('0'); //se agrega un nuevo cero
        if(lectura<10)
        {
            //finalmente el valor es menor que 10
            Serial.print('0'); // se agrega un cero, en este punto al valor se le agregan tres ceros
            //antes de enviarse por el puerto serie
        }
    }
}
Serial.print(lectura);
}

//-----Función en la que se guardan los valores del buffer de entrada-----
//-----
void InDatos()
{
    kp = Serial.parseFloat(); // guardar el primer valor en kp
    Serial.read(); //se elimina el separador: ','
    ki = Serial.parseFloat(); //guardar el segundo valor en ki
    Serial.read(); //se elimina el separador: ','
    kd = Serial.parseFloat(); //guardar el tercer valor en kd
    Serial.read(); //se elimina el separador: ','
    YD = Serial.parseFloat(); //guardar el ultimo valor en valD (valor deseado)
    Serial.read(); //se borran los datos sobrantes
}

//-----Función que limpia el buffer, después de guardar datos-----
//-----
void Limpieza()
{
    while (Serial.available() > 0) //asigna al último carácter a la variable de cierre
    {
        delayMicroseconds(100); //Tiempo de espera para terminar de leer bien los datos, vaciar el
        //buffer y una buena comunicación con labview
        cierre=Serial.read(); //borra caracteres en el buffer
    }
}

//-----Función para ajustar las constantes-----
//-----
void ConstantesK(double kp, double ki, double kd)
{
    TMseg = (double)(DT/1000.0); //El tiempo se divide entre mil para que el resultado este en segundos
    kp = kp;
    ki = ki * TMseg;
}

```

```

kd = kd / TMseg;
}

//-----
//-----Función del controlador PID-----
//-----
int PID_1(double kp, double ki,double kd,double yd,double ym)
{
  ConstantesK(kp,ki,kd);
  error=yd-ym;           //Cálculo del error , valor deseado menos valor medido
  Direccion(error);     //Dependiendo del signo del error, el motor gira en un sentido o en otro
  if(contador1==0)      // para la primera iteración, el error previo es igual al valor deseado
  error_pre=yd;         //error previo igual al valor deseado.
  errorSuma += (ki*error); //Cálculo del error integral
  if(errorSuma>Max) errorSuma=Max;
  else if(errorSuma<Min)errorSuma=Min;
  derror=(error-error_pre); //Cálculo del error derivativo
  u1=kp*error+errorSuma+kd*derror; //Ley de control
  //Anti wind-up versión del PDF PID Digital en PIC y del PDF del PID de arduino
  if(u1>Max) u1=Max;
  else if(u1<Min) u1=Min;
  error_pre=error;      //actualización del error previo
  //Empieza sección para imprimir el valor medido, la señal de control, valor deseado
  agregar_ceros(u1);
  Serial.print(',');
  agregar_ceros(ym);
  Serial.print(',');
  agregar_ceros(tiempo);
  Serial.println();
  //Fin de sección de impresión // 00255.00,00000.00,00000.00/n 28 datos enviados
  contador1=contador1+1; // el contador aumenta para que el error previo ya no sea el valor deseado
  return u1;
}

//-----
//-----Función para dirección del motor-----
//-----
void Direccion(double ERROR)
{
  if(ERROR<0)
  {
    digitalWrite(motorA, LOW);
    digitalWrite(motorB, HIGH);
  }
  else
  {
    digitalWrite(motorA, HIGH);
    digitalWrite(motorB, LOW);
  }
}
}

```

```
//-----  
//-----Función para encender el led rgb-----  
//-----  
void rgb(byte R, byte G,byte B) //función para prender el led rgb, 1 = ON 0= OFF  
{  
  digitalWrite(blue,B);  
  digitalWrite(red,R);  
  digitalWrite(green,G);  
}
```

9.2 Código de programación de las interfaces.

Las interfaces de usuario se realizaron con la siguiente estructura, ambas interfaces 1 Y 2 tiene pequeñas diferencias, como la velocidad de baudios y el tener más o menos botones de interacción. Entonces solo se describe la interfaz más compleja que es la interfaz_1. La descripción del código solo será de forma general, conocer el código completo se tiene que pedir los archivos correspondientes al profesor encargado del banco de pruebas.

La interfaz tiene como estructura principal un ciclo *While loop*, que está dividido en varias partes, tal como se muestra en la siguiente Figura.

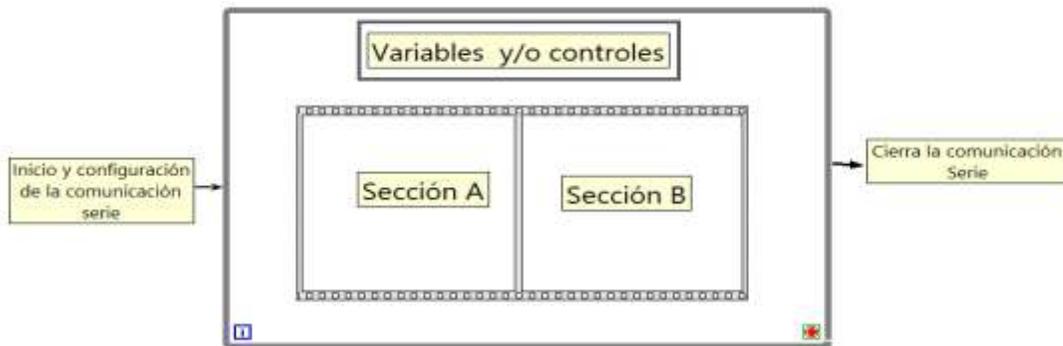


Figura 9-1 Vista General de la Programación de la Interfaz

El programa comienza por Inicializar tres arreglos en cero, al igual que la configuración de la comunicación serie. Dentro de la sección A, de izquierda a derecha, se cuenta con nodos de propiedad que sirven para hacer que los controles o indicadores (K_p , K_i , K_d , ZN , $Obj...$) sean visibles u ocultos dentro del case y acorde al caso serán ocultos o visibles en la interfaz, así mismo dentro de los casos se cambian las etiquetas de la gráfica en la que se muestran el valor medido (velocidad/posición) contra el tiempo. En la parte superior esta otro case para cambiar las etiquetas de la gráfica que se encuentra en la zona de GRAFICAR de la interfaz. En el lado izquierdo esta la estructura *event*, que lanza eventos según el control presionado (STOP, START, Exportar Imagen, Reset Graph y Archivo a Graficar). Los eventos STOP y START crean una cadena de caracteres con la información que llegará al microcontrolador y este poder ejecutar lo que se indica. En START se crea un archivo y se inicia un arreglo para guardar los datos del experimento. Al ejecutarse STOP los datos del arreglo se guardan en el archivo creado al inicio de la prueba. En Reset Graph vuelve a iniciar los arreglos de los datos que se muestran en las gráficas, es decir, se borran los datos desplegados. Para el evento Exportar Imagen se configura la invocación de nodo para que sea posible guardar una imagen con los datos

del archivo creado en el evento START y STOP, mientras que Archivo a Graficar sirve para seleccionar la ruta que tiene el archivo y graficarlo nuevamente.

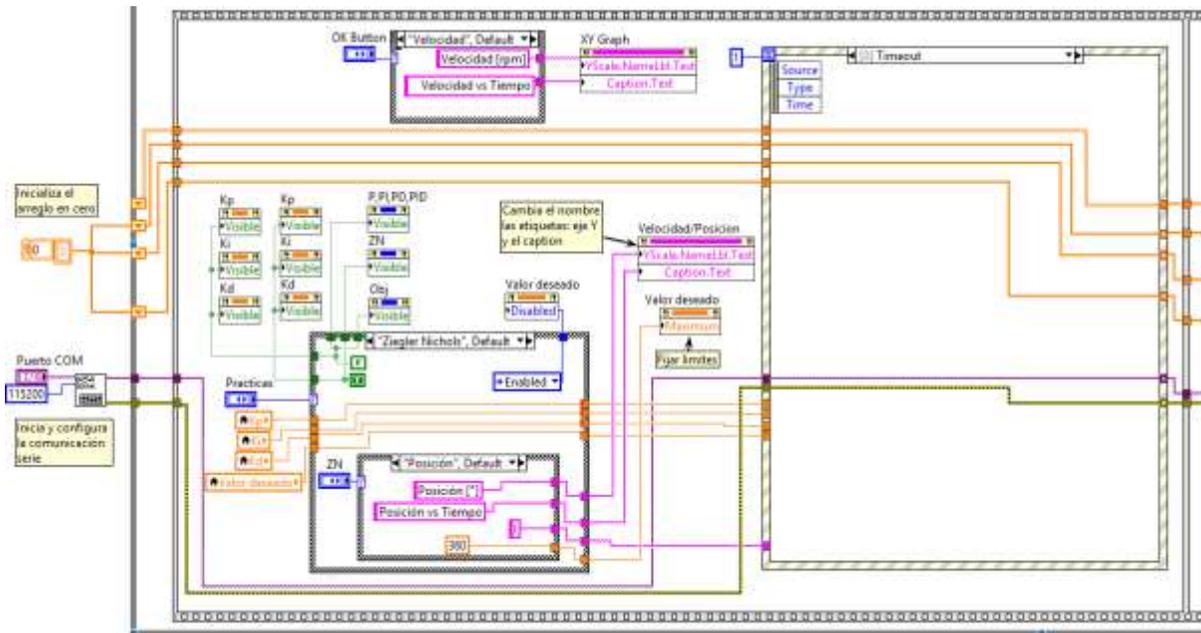


Figura 9-2 Sección A

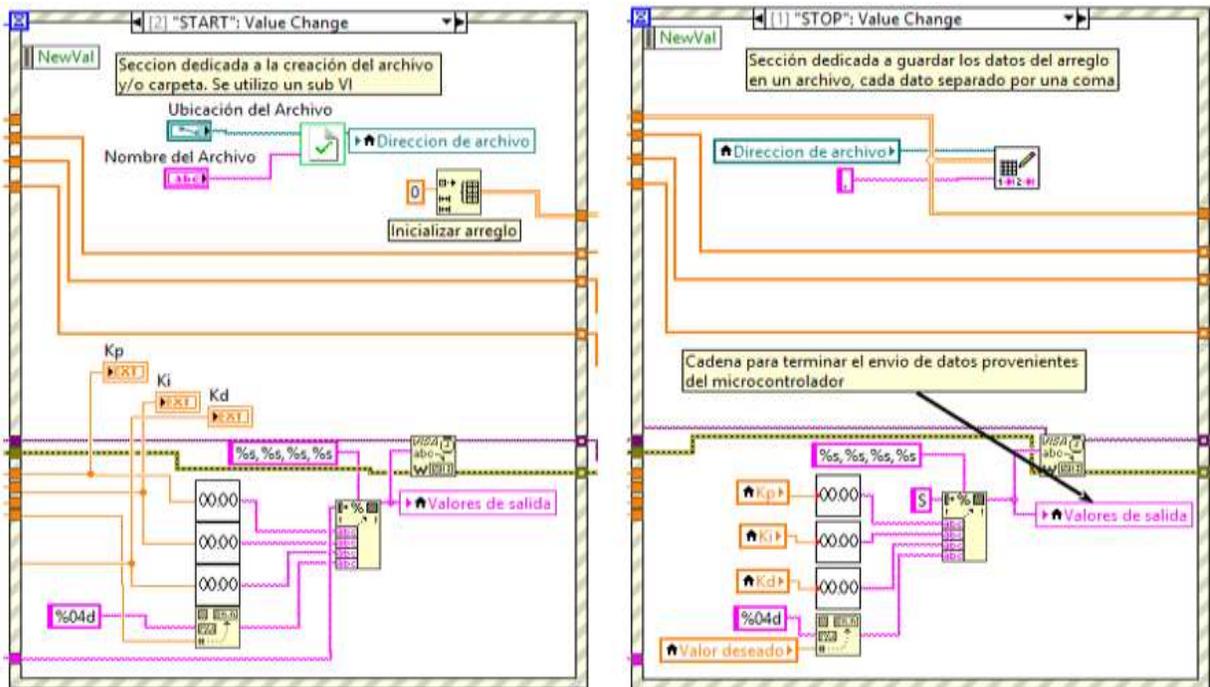


Figura 9-3 Eventos START y STOP

La sección B consta de un case que le permite leer los datos proveniente del microcontrolador, y así mostrarlos en las gráficas correspondientes. Los datos son tomados y guardados en el arreglo que se inició en START.

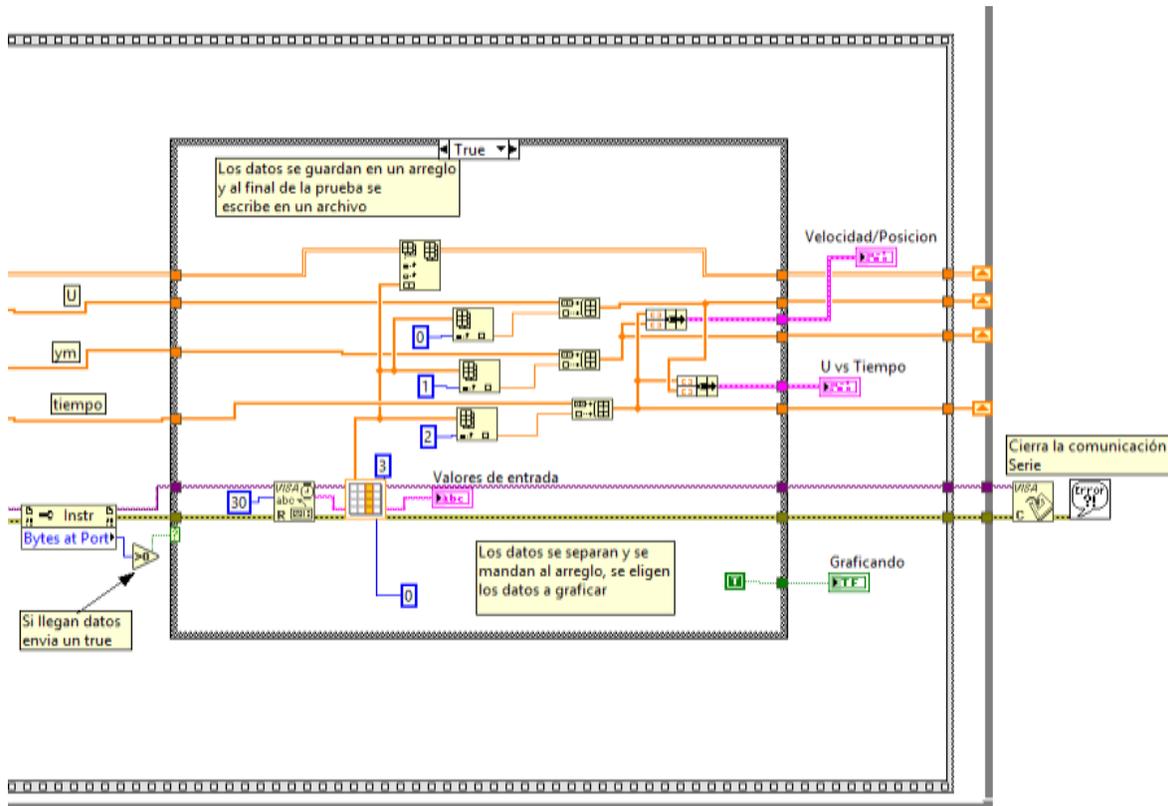


Figura 9-4 Sección B

Por último cabe mencionar que se utilizan Sub VI (subprogramas), los cuales se pueden consultar con detalle en la carpeta de las interfaces, que puedes pedir al profesor encargado. El primero de ellos es **sub Creacion_Carpeta_Archivo.vi** que facilita la creación del archivo donde se guardan los datos del experimento, además de crear una carpeta para los experimentos. Y **sub Tratamiento_String.vi** es empleado para separar la cadena de caracteres provenientes de uC. Finalmente **sub AgregarCeros_2.vi** agrega ceros a la cadena de caracteres enviada al uC para que siempre tenga el mismo número de caracteres.



Figura 9-7 sub
Creacion_Carpeta_Archivo.vi



Figura 9-6 sub
Tratamiento_String.vi



Figura 9-5 sub
AgregarCeros_2.vi

9.3 Especificaciones de material eléctrico o electrónico.

9.3.1 Metal Gearmotor 37Dx57L mm con Encoder 64 CPR

- Dimensiones: 2.71"x1.45"x1.45"
- Eje en forma de "D" 6mm de diámetro.
- Velocidad libre de 40 [RPM] a 6 [v].
- Corriente rotor libre 250 [mA] a 6 [v].
- Corriente rotor bloqueado 2500 [mA] a 6 [v].
- Par (torque) de rotor bloqueado 125 [oz*in] = 9 [Kgf-cm].
- Velocidad libre de 80 [RPM] a 12 [v].
- Corriente rotor libre 300 [mA] a 12 [v].
- Corriente rotor bloqueado 5000 [mA] a 12 [v].
- Par (torque) de rotor bloqueado 250 [oz*in] = 18 [Kgf-cm].
- Gearbox o reducción de 131.25:1.
- Encoder de cuadratura de efecto hall integrado de 64 CPR.
- La resistencia del motor se puede tomar directamente de las terminales del mismo (sin carga). También se puede aproximar dividiendo el voltaje aplicado entre la corriente cuando el eje está en paro.
- K_e puede ser aproximada dividiendo el voltaje aplicado entre la velocidad a rotor libre (del voltaje aplicado).
- $K_t = 3.59753028 \left[\frac{Kg\,f\,cm}{A} \right]$ (Valor experimentalmente).
- No existe hoja de especificaciones (Datasheet). Para más información consultar la página del proveedor: <https://www.pololu.com/product/1447/sp-ecs>

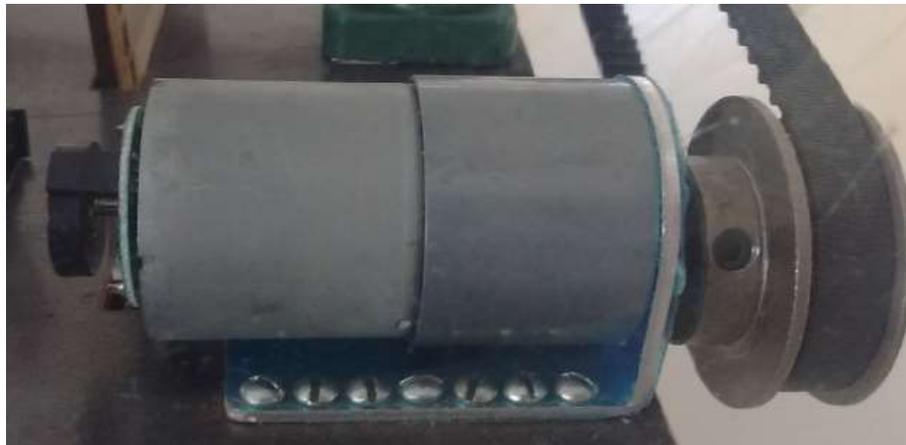


Figura 9-8 Motor Grande

9.3.2 Micro Metal Gearmotor 100:1

- Dimensiones: 0.94"x0.39"x 0.47".
- Motor con escobillas 6 [V]. 320 [RPM].
- 120 [mA] rotor libre, 1.6 [A] rotor bloqueado.
- 2 [Kgf-cm] de par nominal.
- Gearbox: 100.37:1.
- Eje en forma de "D" 3 [mm] de diámetro. 9 [mm] de largo.
- Encoder de cuadratura con fototransistores. 20 CPR. 5 [V] de alimentación.

- Para más información <<
<https://www.pololu.com/product/3065>
>>



Figura 9-9 Motor pequeño

9.3.3 ACS714 Current Sensor Carrier -30A to +30A

- Opera a 5 [V]
- Sensibilidad de 66 [mV/A]
- Corriente de entrada bidireccional de -30 [A] a 30 [A]
- Resistencia del sensor: 1.8 [OHMS]
- Para más información <<
<https://www.pololu.com/product/1187> >>

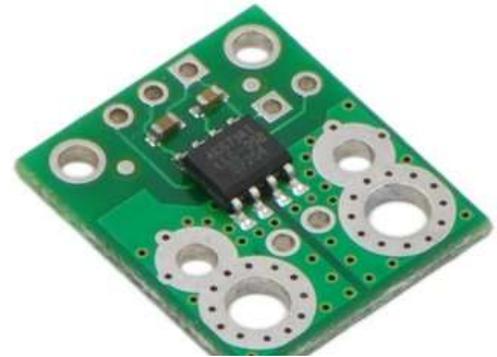


Figura 9-10 Sensor de corriente ACS714

9.3.4 VNH5019 Motor Driver Carrier

- 5.5 a 24 [V] de alimentación para el motor.
- Puede liberar corriente continua de 12 [A]. (30A pico).
- Trabaja con niveles lógicos de 2.5 a 5 [V].
- Protección contra sobre-voltaje o voltaje en reversa, temperatura y corrientes elevadas.
- Para más información <<
<https://www.pololu.com/product/1451> >>

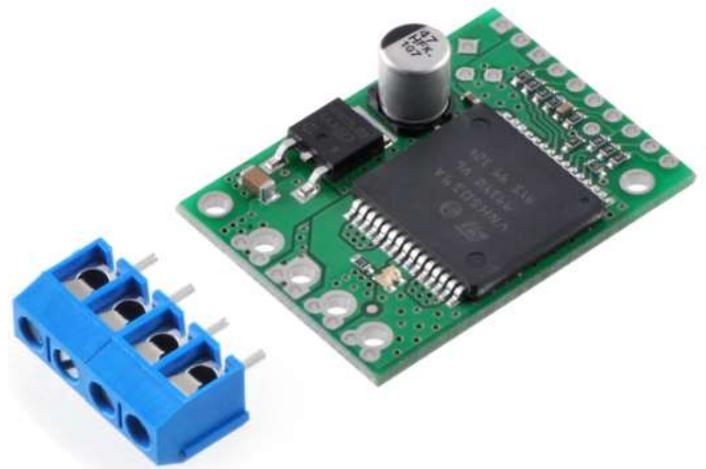


Figura 9-11 VNH5019

9.3.5 Bluetooth HC-06

- SPP (Serial Port Protocol).
- Interface UART con baud rate programable.
- Antena integrada.
- Voltajes de entrada y salida de 3.3 [V] a 5.0 [V].
- Corriente promedio 8 [mA] a 25 [mA].



Figura 9-12 Bluetooth HC-06

9.3.6 Tiva C Series TM4C123G. EK-TM4C123GXL

- Voltaje de alimentación de la tarjeta: 4.75 a 5.25 [V].
- Dimensiones (LargoxAnchoxAlto): 5.0x5.715x1.0795 [cm].
- Voltaje brindado: 3.3 [V] (300 [mA] Max).
- Palabra de 32 bits.
- 256 KB de memoria flash.
- Frecuencia de reloj de 80 MHz.
- ADC de 12 bits.
- 23 salidas de PWM.
- 12 entradas analógicas.
- << http://energia.nu/pin-maps/guide_tm4c123launchpad/ >>



Figura 9-13 TM4C123G. EK-TM4C123GXL

9.4 Especificación de tarjeta PCB del banco de pruebas grande

En esta sección se muestra la tarjeta PCB Figura 9-14 con la que cuenta el banco de prueba, tanto la imagen como la tarjeta cuenta con etiquetas (Nombre y número de pin) para facilitar la identificación de los componentes al momento de su respectiva conexión. También se añadieron salidas extras para voltaje de 3.3 y GND para nuevas aplicaciones que se quieran desarrollar con el bando, además de contar con un apartado llamado “FUENTE” para agregar una fuente extra de 5V en caso de que se necesite, y no emplear la salida de 5 volts del microcontrolador.

Se agregan tablas para identificar el número de pin de cada componente. Por ejemplo las salidas del puente H (H-BRIDGE) que se conectan a la tarjeta, están enumeradas del 1 al 5 y en la tabla se muestra a que corresponde el pin 1 al pin 5.

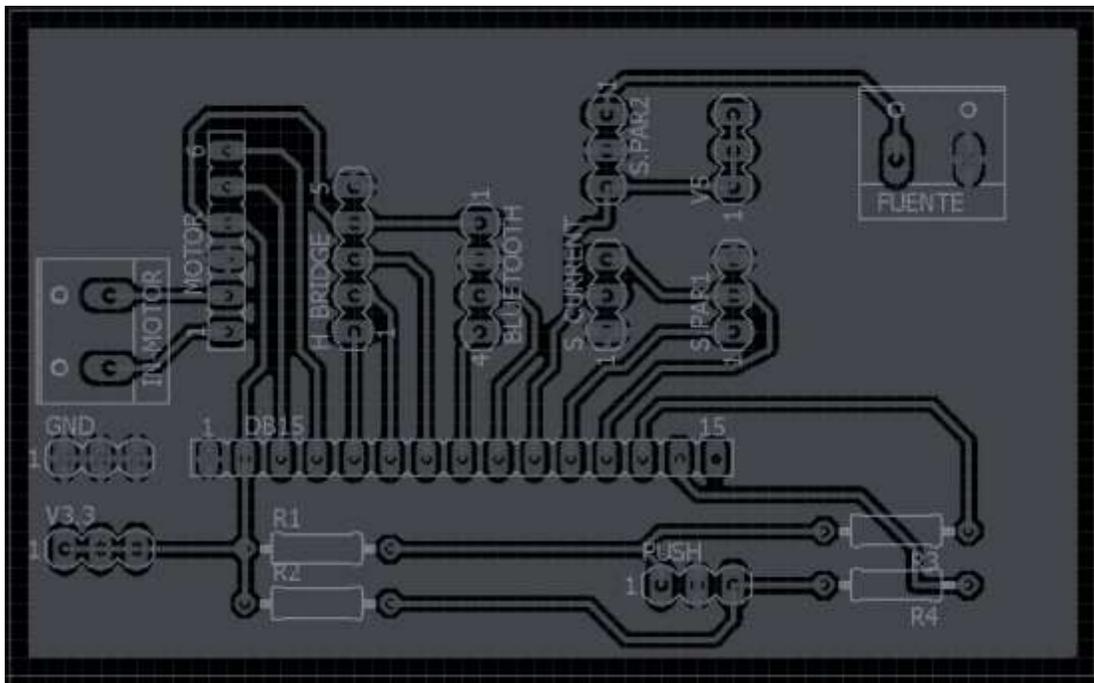


Figura 9-14 Placa PCB

BLUETOOTH	
PIN	Nombre
1	3.3V
2	GND
3	TX
4	RX
S.CURRENT	
PIN	Nombre
1	GND
2	5V
3	VOUT(CORRIENTE)

Resistencias	
1,2	1K Ω
3,4	330 Ω

S.PAR1	
PIN	Nombre
1	PAR
2	VOUT(CORRIENTE)
3	GND

PUSH	
PIN	Nombre
1	BOTON(ROJO)
2	GND
3	BOTON(AZUL)

S.PAR2	
PIN	Nombre
1	FUENTE
2	GND
3	5V

DB15	
PIN	Nombre
1	GND
2	3.3V
3	EncoderA
4	EncoderB
5	DriverA
6	DriverB
7	PWM
8	RX
9	TX
10	VCC(5V)
11	VOUT(PAR)
12	VOUT(CORRIENTE)
13	Botón(ROJO)
14	Botón(AZUL)

H.BRIDGE	
PIN	Nombre
1	INA
2	INB
3	PWM
4	3.3V
5	GND
MOTOR	
PIN	Nombre
1	M.powerRojo
2	M.powerNegro
3	GND
4	3.3V
5	Encoder A
6	Encoder B