Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

IE-0502

Proyecto Eléctrico I

Aplicación de los algoritmos PID a un Controlador Lógico Programable

Elaborado por:

José Francisco Miranda Vázquez C.U. 992515

Diciembre 2004

Aplicación de los algoritmos PID a un Controlador Lógico Programable

Por

José Francisco Miranda Vázquez

Sometido a la Escuela de Ingeniería Eléctrica
de la Facultad de Ingeniería
de la Universidad de Costa Rica
como requisito para optar por el grado de

BACHILLER EN INGENIERIA ELECTRICA

Aprobado por el Tribunal

_____	_____
Ing. Ismael Mazón, M.Sc	Ing. Jorge Blanco, M.Sc

_____

Ing. Peter Zeledón

Diciembre, 2004

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

Simbología

PLC	Controlador lógico programable
CPU	Unidad central de proceso
PC	Computador personal
V	Voltios. Unidad de medida de la diferencia de potencial eléctrico
A	Amperios. Unidad de medida de la corriente eléctrica
Hz	Hertz. Unidad de medida de la frecuencia
ms	Milésima parte de un segundo
PID	Algoritmo de control de acción proporcional, integral y derivativa
s	Variable compleja
z	Variable compleja
A/D	Convertidor analógico a digital
D/A	Convertidor digital a analógico
RAM	Memoria de acceso aleatorio
TCP/IP	Protocolo de capa de transporte/protocolo de internet
ac	Corriente alterna
dc	Corriente directa
VD	Valor deseado
VC	Variable controlada
SC	Salida del controlador

RESUMEN

El proyecto consistió en la utilización del controlador lógico programable Allen Bradley SLC 500 disponible en el Laboratorio de Automática de la Escuela de Ingeniería Eléctrica de la Universidad de Costa Rica, como controlador en un sistema automático.

En el trabajo se estudia la aplicación de los algoritmos PID al PLC con el fin de controlar una planta analógica en un lazo realimentado. Primero se realiza una descripción detallada de los componentes del equipo, después se presentan los programas necesarios para la utilización del PLC. Además se explica la configuración del controlador, su lenguaje de programación y el direccionamiento de las variables internas.

Se realizó la sintonización del algoritmo PID para que el sistema automático cumpliera con ciertos parámetros de diseño. Después se realizó un modelado matemático del sistema para poder comparar los datos obtenidos experimentalmente. El modelado se realizó mediante la utilización de las funciones de transferencia continuas y discretas.

Se implementó el sistema de control en el Laboratorio de Automática de la Escuela, donde se realizaron numerosas pruebas experimentales bajo diferentes configuraciones del equipo. Se diseñó una interface gráfica con el usuario que le permite monitorear y configurar el sistema automático.

Finalmente se analizaron los resultados experimentales obtenidos, y se concluyó sobre aspectos como la estabilidad y el método de sintonización utilizado.

Introducción

El proyecto consiste en utilizar un controlador lógico programable (PLC por sus siglas en inglés) para el control de un proceso continuo en un lazo realimentado, haciendo uso de los algoritmos PID (proporcional, integral, derivativo) como estrategia de control. Para implementar el sistema se utilizará el PLC SLC 500 de Allen Bradley® y un proceso que será una planta electrónica analógica. Además se desarrollará una interface con el usuario, a través de un computador personal, con el objetivo de poder seguir el comportamiento del proceso, y de modificar las variables de control.

Los objetivos y la metodología del proyecto se desarrollan en el capítulo 1. Además, se presenta una justificación que resume los motivos que movieron a la realización de este trabajo.

En el capítulo 2 se presentan los conceptos básicos de los controladores lógicos programables. Se explica su principio de funcionamiento, las partes que los constituyen y algunos conceptos sobre su modo de empleo.

El capítulo 3 es dedicado a una descripción básica del PLC SLC 500 de Allen Bradley. Primero se presentan las especificaciones del equipo para después introducir los programas RSLogix y RSview de Allen Bradley®, necesarios para la programación del PLC y la interface al ususario. Se exploran conceptos tales como el direccionamiento de variables y las instrucciones básicas del lenguaje de programación.

El proyecto de control es desarrollado en el capítulo 4. Primero se especifica el equipo y las conexiones necesarias para la implementación. Posteriormente se presenta la programación del PLC. Finalmente se explicará cómo se crea la interface al usuario que se ejecutará en el PC. En este capítulo se definirán las especificaciones que el sistema de control deberá cumplir y se configurará el algoritmo PID tal que se cumpla con ellas. Finalmente, se realiza un modelo del

sistema en tiempo continuo y en tiempo discreto, que será utilizado como parámetro de comparación de resultados.

En el capítulo 5 se presentan los resultados de las pruebas realizadas con el sistema en el laboratorio. Con ayuda del modelo, se realiza un análisis de resultados.

Finalmente, en el capítulo 6 se desarrollan conclusiones y recomendaciones para el adecuado uso de un PLC en sistemas de control realimentado.

Además, se incluye un apéndice con los conceptos básicos del diseño de sistemas realimentados de control.

Capítulo 1: Objetivos y Metodología

1.1 Objetivo General.

Utilizar un PLC para el control realimentado de un proceso continuo, haciendo uso de una estrategia de control PID.

1.2 Objetivos específicos.

1. Utilizar un PLC en una configuración de lazo cerrado para controlar una planta mediante un algoritmo PID.

2. Aplicar los métodos de sintonización de controladores PID a los parámetros de control implementados en el PLC.

3. Comunicar el PLC con una computadora que servirá como interface al usuario.

4. Monitorear el comportamiento de la planta a través de datos enviados por el PLC a un computador.

1.3 Justificación.

La creciente necesidad de obtener opciones de control de procesos industriales que sean flexibles, económicas y de mayor precisión incentiva la aplicación de teorías de control conocidas a equipos más avanzados. Es por ello que se realiza este estudio; que busca implementar la teoría PID a los controladores lógicos programables, equipos de avanzada tecnología, alta precisión y gran flexibilidad.

Tradicionalmente, los PLC han sido utilizados en el control de procesos secuenciales. Entre estos procesos están: el arranque y paro de motores, coordinación de sensores para muestreo de datos, manejo de transferencias de energía de respaldo, control de comunicación de datos entre equipos periféricos, entre otros. Por otro lado, las teorías de control para procesos más avanzados de regulación y servomecanismos como los algoritmos PID, son llevadas a la práctica por equipos electrónicos analógicos discretos, los cuales son poco flexibles. Por ello surge la inquietud de explorar el uso de los PLC en el control de procesos más complejos de manera que todas las ventajas que ofrecen puedan ser aprovechadas en procesos como: regulación de temperatura de calderas, posicionamiento de robots, preparación de sustancias químicas, etc.

1.4 Metodología.

El proyecto busca implementar la teoría PID a un control de una planta electrónica analógica en un lazo cerrado con un PLC como controlador. Para lograr llevar a cabo la construcción del sistema, se realizará primero una investigación bibliográfica sobre algoritmos PID, la programación de controladores Allen Bradley®, sobre las comunicaciones seriales bajo el protocolo de capa física RS-232 y sobre la programación de una interface en RSView32®.

Después de la recolección de la información se harán los cálculos necesarios para sintonizar el controlador. Se realizará la programación necesaria del PLC mediante la herramienta RSLogix y se montará la interface al usuario en la plataforma RSView32.

Finalmente se realizarán las conexiones del equipo, para luego obtener las respuestas del sistema en el laboratorio para diferentes situaciones. Estas respuestas serán comparadas con los resultados teóricos obtenidos del análisis del sistema en tiempo continuo y en tiempo discreto. De este análisis se obtendrán las conclusiones y recomendaciones.

Capítulo 2: Fundamento teórico

2.1 Sistemas de Control Digital.

Un sistema de control donde el controlador es un computador digital es conocido como *sistema de control digital*. El computador digital es un elemento electrónico, que utiliza señales digitales en su funcionamiento.

Un *sistema de control digital* se divide en las siguientes partes:

Proceso: Es la parte del sistema que realiza la tarea requerida con el fin de obtener un servicio o producto. También conocido como *planta,* posee parámetros físicos manipulables sobre los que se aplica control con el fin de que la tarea o producto cumpla con las *especificaciones de diseño*. El proceso puede estar constituido por motores, circuitos electrónicos, o partes mecánicas.

Controlador digital: Es un equipo electrónico digital que obtiene información del proceso, por medio de sus entradas, para luego aplicarle el *algoritmo de control*. Los resultados son aplicados a la planta, a través de sus salidas. Las señales que maneja son eléctricas y codificadas en el sistema de numeración binario.

Elementos de medición y transmición: son dispositivos que sirven para medir variables físicas de cualquier índole y transformalas en señales eléctricas, que son realimentadas al controlador digital.

Muestreador: es el elemento que transforma las señales eléctricas analógicas en señales muestreadas moduladas en amplitud. En este trabajo siempre se supondrá que los muestreadores son ideales, representados por interruptores que se abren y cierran con un periodo **T**. Todos los interruptores en los diagramas de bloques están sincronizados.

Retenedor: Dispositivo que transforma las señales muestreadas en señales cuantizadas. En este trabajo, los retenedores del sistema serán representados por el modelo *ZOH* (retenedor de orden cero por sus siglas en inglés).

Actuador: El actuador recibe las señales eléctricas que provienen del controlador y realiza una acción directa sobre el parámetro manipulable de la planta. Recuérdese que este parámetro manipulable debe tener un efecto final sobre la variable física que se está controlando.

La figura 2.1 muestra un ejemplo de un sistema de control digital. En este caso, se trata del control del nivel de fluido en un tanque.

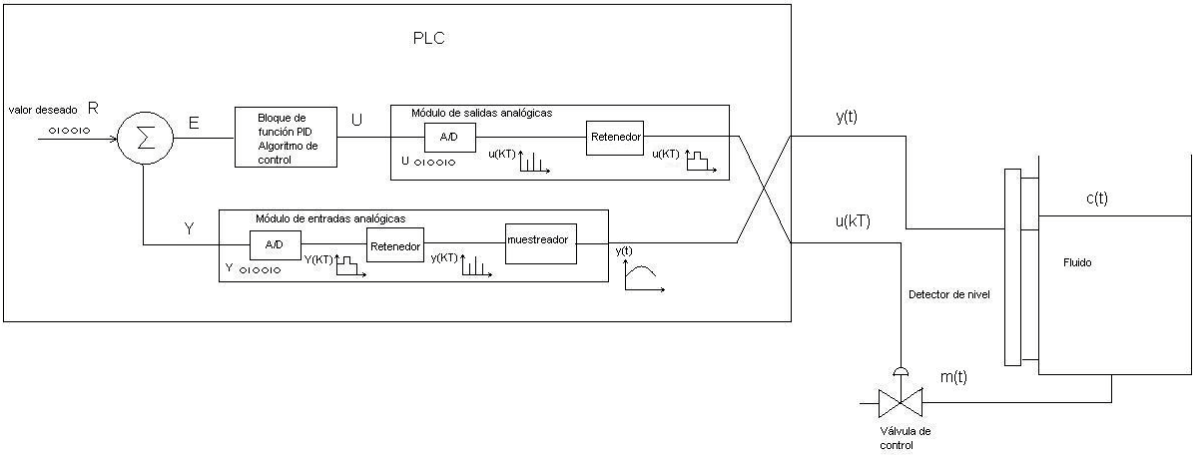


Figura 2.1 Sistema de control digital.

El actuador es una electroválvula que deja escapar el fluido de la base del tanque. El controlador digital es un PLC, que obtiene el valor del nivel presente de un sensor transmisor. Las señales que se pueden identificar en el sistema son:

.

Valor deseado ó consigna R. Es el valor de meta para la *variable controlada*. Esta señal puede ser transmitida al PLC por medio de una entrada analógica, ó por medio de una PC.

Error. E: Señal de error, que es igual a la *señal de consigna* menos la *señal realimentada*. Es una señal digital.

Salida del controlador U: es el resultado de la aplicación de la ley de control sobre el error. Es una señal digital.

Salida del PLC u (k*T): es la señal presente en los bornes de salida del PLC. Es una señal analógica que comanda el actuador.

Variable manipulada m(t): es el parámetro físico de la planta sobre el que se puede actuar para obtener un cambio en la *variable controlada*.

Variable controlada c(t): es la variable de la planta que se controla directamente. En la figura 2.1, esta variable sería el nivel del fluido en el tanque. La magnitud de la *variable controlada* es representada por una señal eléctrica llamada *señal realimentada*. El controlador recibe la información de la *variable controlada* por medio de ella.

Señal realimentada y(t): Es la señal que se obtiene del sensor transmisor. Es una señal eléctrica analógica

Ahora se explicarán cada uno de los elementos de la figura 2.1.

Entradas del PLC: Es un circuito electrónico, parte del PLC, que realiza las acciones de muestreador y retenedor. Este circuito recibe una señal analógica de entrada y la transforma en una señal cuantizada.

Convertidor analógico/digital A/D: Un convertidor analógico/digital es un codificador que convierte una señal analógica de entrada en un código numérico, generalmente binario. Este tipo de señales son las que el controlador digital puede manipular. Dicha conversión es realizada con un grado de incertidumbre que depende de las características del convertidor.

Convertidor digital/analógico D/A: Es un decodificador que transforma la señal digital de salida del *algoritmo de control* en una señal tipo cuantizada.

El lazo de control funciona así: la *señal realimentada* es primero transformada a una señal digital que es realimentada y restada a la señal de *valor deseado*. La diferencia es procesada por el PLC al aplicarle el *algoritmo de control*. El resultado es una señal digital que se transforma en analógica mediante el convertidor D/A, para luego ser aplicada al actuador. Finalmente el actuador varía la *variable manipulada* para obtener el resultado en *la variable controlada*.

2.2 Controladores lógicos programables PLC.

Los controladores lógicos programables son computadores digitales industriales dedicados a las tareas de control de procesos. Dichos dispositivos fueron creados para mejorar el sistema de control convencional mediante contactores. Entre los problemas que presentaban los sistemas de control convencional están: poca flexibilidad, difícil supervisión y corrección de errores, poca confiabilidad y alto consumo de energía. Las razones de estos problemas radican en que los sistemas convencionales implementan la lógica de control mediante cableados complicados y múltiples elementos discretos como temporizadores, contactores, interruptores, enclaves mecánicos y botoneras, todos los cuales están sujetos a fallo. Además, el hecho de que la lógica esté implementada por cableados hace difícil su modificación.

Como respuesta, surgió la idea de tener un único elemento programable que realizara la lógica de control. Entonces, bastará con programar dicha lógica en la memoria del dispositivo para obtener el mismo resultado que con la intricada red de contactores que se tenía previamente. El dispositivo fue llamado controlador lógico programable PLC, nombre que resalta su característica más importante: el hecho de que es programable. Esta cualidad permite que el equipo pueda ser utilizado en una gran diversidad de procesos, ofreciendo a la industria flexibilidad y adaptabilidad a los cambios.

2.2.1 Componentes básicos de los PLC.

1- Entradas: Constituyen la etapa de entrada del PLC. Desde la parte externa del PLC lucen como una bornera donde se deben colocar los cables con las señales que provienen de los transductores, pero internamente están conformadas por circuitos electrónicos que acoplan esas señales a las especificaciones de señales que el PLC puede manipular.

Según la naturaleza de la señal que se recibe de los transductores, las entradas se clasifican en:

a-) *Entradas digitales:* Estas entradas se diseñan para recibir señales cuantizadas de los sensores de campo. Dichas señales varían sólo entre dos estados. El PLC codifica estas señales según su amplitud en: 1 lógico para el valor de amplitud mayor, y 0 lógico para el nivel de amplitud menor. Los niveles de amplitud que el PLC entenderá son definidos por el fabricante. Este tipo de señales generalmente provienen de transductores como: interruptores, botoneras, sensores de fin de carrera, etc.

b-) *Entradas analógicas*: son las que reciben señales analógicas de los transductores de campo. Estas señales generalmente provienen de sensores que miden el valor instantáneo de una variable física. Ejemplos de este tipo de señales son: la salida de una tacométrica, de un fotosensor o de un sensor de nivel. El valor de la señal analógica se transforma en una señal digital de tal forma que el procesador la pueda manipular. Un aspecto importante de esta transformación es la resolución con que se realiza en el interior del PLC. Por resolución se entenderá la cantidad valores cuantizados disponibles para representar una señal analógica. Por ejemplo, si se tiene sólo dos valores cuantizados para representar una señal que varía de 0 a 5 V, se dice que se tiene una resolución de dos. La resolución depende de las características de la entrada. La cantidad de valores cuantizados es igual a 2^n, con **n** el número de bits del registro donde se almacena la variable digital que resulta de la transformación. Generalmente, en los controladores más sofisticados, se asocia un registro de 16 bits a cada una de las entradas analógicas, con lo que se tiene una resolución de 2^{16}.

Según el tipo de señal eléctrica que reciban, las entradas también se clasifican en: de corriente y de voltaje. A las entradas está asignado un espacio de memoria del PLC llamado *imagen de entradas*, el cual contiene la información de todas las entradas en todo momento.

2- Salidas: Internamente son circuitos electrónicos que realizan el acople entre las señales digitales utilizadas por el PLC y las señales analógicas o cuantizadas que utilizan los actuadores. Externamente lucen como una bornera donde se realizan las conexiones entre el PLC y los actuadores .

Las salidas se clasifican, al igual que en el caso de las entradas, en digitales y analógicas. Las salidas digitales se aplican a actuadores como bobinas de contactores, electroválvulas, etc.

Existen salidas digitales: de voltaje y de relé. Las salidas de voltaje asignan una magnitud de voltaje, que depende del fabricante, al estado 1 lógico y de 0 V al estado 0 lógico. Las salidas de relé consisten en un contacto seco que se cierra en el estado1 y se abre en el estado 0.

En el caso de salidas analógicas, los valores de salida están generalmente entre 0 V*dc* a 10 V*dc* para las salidas de voltaje y de 4 mA a 10 mA para las de corriente, aunque estos valores varían según el fabricante. Estas señales comandan actuadores como válvulas solenoides, servomotores, etc.

A las salidas se les asigna un espacio de memoria del PLC llamado *imagen de salida*, el cual contiene la información de todas las salidas en todo momento.

2- Unidad central de proceso: ó CPU por sus siglas en inglés. Es el elemento principal de procesamiento del PLC. Una vez digitalizadas, las señales de entrada son pasadas al CPU, el cual les aplica el *algoritmo de control* para generar las salidas. El *algoritmo de control* está almacenado en la memoria interna del PLC en forma de un *programa,* el cual es creado y

almacenado por el usuario. Además de ejecutar el *programa*, el CPU realiza acciones como verificación del sistema, actualización de las imágenes de entrada y salida y la medición del tiempo de ejecución del *programa*.

3- Memoria del PLC: es el lugar físico donde residen el sistema operativo, el *programa*, los datos de ejecución y las imagenes de entrada y salida. El sistema operativo es un programa que utiliza el PLC para iniciar su operación y realizar las configuraciones propias de su funcionamiento.

La memoria del PLC se clasifica en diferentes clases dependiendo de su modo de acceso y volatibilidad.

a-) EEPROM: es una memoria de sólo lectura que puede ser escrita por medios electrónicos. No necesita de una fuente de poder para mantener sus datos. Por su característica no volátil, se utiliza para guardar datos esenciales, tal como el sistema operativo y el *programa*.

b-) RAM: es una memoria reescribible de acceso aleatorio que se utiliza para guardar los datos generados mientras se ejecuta el programa. Es volátil, por lo que los datos almacenados se pierden si se le suspende la alimentación.

4-) Fuente de poder: Es el elemento que brinda la alimentación a todos los componentes del PLC. Generalmente los componentes funcionan a bajos voltajes de *dc*. La fuente realiza la transformación de los voltajes *ac* de las lineas de potencia a esos niveles *dc*.

2.2.2 Tipos de PLC.

Los PLC se clasifican, según la forma como se presentan sus componentes, en: compactos y modulares:

Compactos: Todos los componentes se encuentran integrados en un solo gabinete. El ususario no tiene acceso a ellos, por lo que no los puede modificar. Generalmente se pueden encontrar con diferentes capacidades en aspectos como: número de entradas, capacidad de memoria, número de salidas, opciones de comunicación, etc.

Modulares: Consisten en un bastidor donde se introducen los diferentes componentes o módulos. Los módulos son intercambiables de un bastidor a otro por lo que las capacidades de un PLC pueden ser ampliadas fácilmente. Generalmente son más costosos que los tipo compacto, pero son mucho más versátiles y útiles en aplicaciones que exigen adaptabilidad a cambios.

2.2.3 Programación de un PLC.

Para que el PLC puede relacionar lógicamente las entradas con las salidas, necesita seguir un *programa* en su memoria. El *programa* tiene descrito, en forma de instrucciones, el *algoritmo de control* deseado.

El *programa* consiste en un archivo o archivos que son generados por la *aplicación de programación*. Estas *aplicaciones de programación* son ejecutadas en dispositivos especiales como herramientas protátiles o computadores personales. Una vez generado el archivo de *programa*, éste se debe descragar a la memoria del PLC.

Las *aplicaciones de programación* brindan una serie de herramientas al usuario para que pueda completar la creación de un *programa*. Entre las herramientas que se suelen ofrecer están: configuración de los componentes del PLC que se utilizará, opciones de comunicación,

diferentes editores de programa, un compilador y plataformas de simulación y de monitoreo. La plataforma de simulación sirve para corroborar el funcionamiento del *programa,* facilitando la puesta en marcha en el campo. La plataforma de monitoreo ayuda al operario a verificar el correcto funcionamiento del sistema desde un equipo remoto como un PC.

Las instrucciones que contiene el *programa* son ejecutadas secuencialmente de forma repetitiva por el CPU.

La mayoría de los fabricantes ofrecen tres editores de *programa*: lista de instrucciones, escalera y diagrama de flujo.

Lista de instrucciones: consiste es una programación por texto, en la cual se le indica al CPU la operación a realizar mediante un comando. Los comandos que se pueden utilizar están predeterminados por el fabricante.

Escalera: Es un lenguaje gráfico que se parece mucho a los diagramas en escalera que se acostumbran en el control convencional. Su lógica incluye los conceptos de contactos normalmente abiertos, cerrados, salidas hacia bobinas, etc.

Diagrama de función: es un lenguaje gráfico por bloques, en el que se dispone de una serie de bloques que realizan funciones específicas. La lógica del *programa* se logra al interconectar los bloques.

2.2.4 Tiempo de ciclo de programa.

El controlador requiere un tiempo para procesar el *programa*. Dicho tiempo se conoce como **tiempo de ciclo de programa**, y depende de la cantidad, tipo de instrucciones del *programa* y de la velocidad del CPU del PLC.

Capítulo 3: Descripción del PLC SLC 500 de Allen Bradley

La serie SLC 500 de Allen Bradley es una familia de PLC modulares que ofrecen una gran cantidad de opciones en módulos de entradas y salidas, comunicación y memoria. Además ofrecen diferentes tipos de CPU según las características de la aplicación.

3.1 Componentes del PLC SLC 500 de Allen Bradley disponible en el Laboratorio de Automática de la Escuela.

A continuación se describirán los módulos específicos de la serie SLC500 que fueron utilizados en este proyecto.

Bastidor 1746-A7: Este bastidor tiene una capacidad de siete módulos. Cada módulo debe ocupar una ranura. Las direcciones de las ranuras son 0 a 6, asignadas de izquierda a derecha. La fuente se debe colocar en un espacio especial, fuera del área de ranuras, en el extremo izquierdo del bastidor. El CPU debe colocarse en la ranura 0.

Fuente de alimentación 1746-P2: Esta es la fuente que alimenta los módulos del PLC. El voltaje de entrada debe estar en el rango de 85-132/170-255 V*ac*, 47-63 Hz. Tiene un voltaje de salida de 5/24 V*dc*. Su consumo típico de potencia es de 135 VA. La selección de la fuente depende del consumo de los módulos que se utilizarán.

CPU SLC 5/05 1747-L551: Este CPU tiene una capacidad de memoria de 16Kbits. Puede direccionar hasta un máximo de 4096 entradas y salidas digitales pues soporta hasta 3 bastidores y 30 ranuras de conexión. Poseé dos canales de comunicación:

Canal 1: protocolo de capa física RS-232 full-duplex master/slave.

Canal 2: protocolo de capa física Ethernet TCP/IP.

Ambos canales soportan los protocolos de comunicación DH-485, ACSII RS-232, Data highway plus, Ethernet, Devicenet, Controlnet.

Además, el CPU tienen una capacidad de memoria para programa de hasta16 mil palabras y una memoria Flash de respaldo.

Módulo de entradas digitales 1746-IA16: consiste en 16 entradas digitales que asignan el estado verdadero al voltaje de 100/120 V*ac*. Se tienen 16 bornes para los voltajes de entrada y un borne para el común o neutro. Los ámbitos de operación para la entrada son 85 a 132 V*ac*, 47 a 63 Hz. El consumo es de 85mA a 5 V*dc* de alimentación.

Módulo de salidas digitales 1746-OW16: El módulo consta de 16 salidas tipo relé, con dos puntos comunes. Por ser contactos de relé, los niveles de las señales de salida se establecen por el usuario. Consume 170 mA a 5 V*dc*. Cada contacto está diseñado para soportar hasta 2.5A a 120V*ac*.

Módulo de entradas/salidas analógicas 1746-NIO4I: en este módulo se tienen dos entradas analógicas de corriente ó voltaje (seleccionable por medio de un interruptor en el módulo) y dos salidas analógicas de corriente. Las entradas del módulo convierten las señales de entrada analógicas en un valor de 16 bits que se guarda en la memoria del PLC. La siguiente tabla muestra los rangos decimales de la imagen de entrada, el número de bits significantes y la resolución, según el rango de la entrada.

Rango de entrada	Rango decimal, imagen de entrada	Número de bits significantes	Resolución
-10 Vdc a 10 Vdc	-32768 a +32767	16	0.305 mV
0 a 10 Vdc	0 a 32676	15	0.305 mV
0 a 5 Vdc	0 a 16384	14	0.305 mV
-20 mA a 20 mA	-16384 a 16384	15	0.00122 mA
0 mA a 20 mA	0 a 16384	14	0.00122 mA

Tabla 3.1 Características del convertidor A/D de los canales de entrada del módulo NIO4I.

La siguiente ecuación sirve para calcular el valor en voltios de la magnitud de la señal de entrada.

$$10\ V/32768 * \textit{valor de entrada} = \text{voltaje de la señal.} \qquad (3.1)$$

Con *valor de entrada*: valor decimal del dato de 16 bits guardado en la imagen de entrada.

Las salidas son señales analógicas cuantizadas. La siguiente tabla muestra el número decimal equivalente en la imagen de salida, el número de bits significantes y la resolución según el rango de la salida.

Rango de salida	Rango decimal, imagen de salida	Número de bits significantes	Resolución
0 a 21 mA	0 a 32,764	13 bits	2.56348 µA
0 a 20 mA	0 a 31,208	12.92 bits	2.56348 µA
4 a 20 mA	6,242 a 31,208	12.6 bits	2.56348 µA

Tabla 3.2 Características del convertidor D/A de los canales de salida del módulo NIO4I.

La figura 3.1 muestra cómo se realizan las conexiones en este módulo.

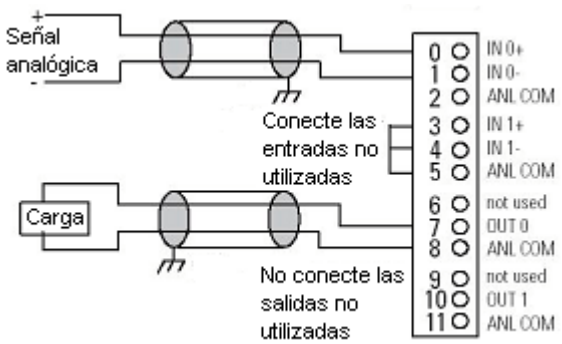


Figura 3.1 Alambrado del módulo de entradas/salidas analógicas.

La siguiente ecuación sirve para calcular el valor decimal que debe ser movido a la imagen de salida, para obtener una señal de salida con una magnitud en miliamperes.

$$32768/21\text{mA} * \text{corriente deseada (mA)} = \text{valor decimal a grabar.} \qquad (3.2)$$

Las especificaciones técnicas del módulo son:

-Aislamiento en las entradas de hasta 500 V.

-Tiempo de actualización de valores en entradas y salidas de 512 µs.

-Ambito de entrada -20 a 20 mA/-10 a 10 V.
-Codificación del módulo para la señal de entrada: -32768 a 32767 (decimal).

3.2. Programación del PLC SLC 500.

Los programas necesarios para programar el PLC SLC 500 son: RSLinx y RSLogix, propiedad de Allen Bradley. Estos programas son diseñados para un PC con una plataforma Windows de 32 bits versión 95 ó superior.

El programa RSLinx es una aplicación de comunicación entre los sistemas operativos Windows de 32 bits versión 95 ó superior y una serie de aplicaciones creadas por Rockwell Software para Allen Bradley, entre ellas, RSLogix.

En la aplicación RSLogix el usuario puede configurar el PLC y programar el *algoritmo de control.* Configurar el PLC es importante, pues se debe corroborar que las direcciones de las variables de entrada/salida que se utilizan en el *programa* corresponden realmente a los módulos conectados al bastidor. RSLogix permite escribir el *programa* y descargarlo en el PLC.

RSView32 es un programa que permite al usuario crear animaciones, simulaciones e interfaces gráficas entre el PLC y un computador personal. La interface permite monitorear, modificar e introducir comandos al *programa* que está almacenado en el PLC. RSView32 está diseñado para ser ejecutado en una computadora personal con un sistema operativo Windows de 32 bits versión 95 ó superior.

3.2.1 Tiempo de ciclo del PLC SLC 500.

La operación básica del CPU es ejecutar, de forma repetitiva, el *programa* que fue cargado por el usuario en la memoria del PLC. El tiempo de ciclo es el tiempo que requiere el CPU para ejecutar ese *programa* una sola vez. Este tiempo se divide en:

Tiempo de actualización de entradas: Durante esta etapa, el CPU lee los datos de entrada y genera una *imagen de entrada* que almacena en un espacio de la memoria.

Tiempo de ejecución de programa: Aquí el CPU ejecuta, línea por línea, las instrucciones del *programa*.

Tiempo de actualización de las salidas: Durante este intervalo el CPU escribe los nuevos datos obtenidos del algoritmo de control en la *imagen de salida.*

Tiempo de configuración: es el tiempo utilizado por el CPU para operaciones internas propias de su operación.

3.2.2 Programación del PLC mediante la aplicación RSLogix.

La aplicación RSLogix permite crear y cargar un *programa* en el PLC. Además permite actualizarlo y modificarlo según lo requiera el usuario.

Para realizar un *programa* se deben seguir los siguientes pasos:

1. Abra el programa RSLogix, previamente instalado en un computador personal.
2. Crear un proyecto nuevo. En el cuadro de diálogo que se despliega, se debe dar un nombre al procesador ó CPU, escoger el tipo de CPU. En manipulador "Driver", se debe escoger el manipulador creado en la aplicación RSLinx. Además, se debe escoger un nodo "Processor Node" de comunicación que se asignará al PLC.
3. En la ventana del proyecto, seleccionar la opción configurar IO "IO Configuration" en la carpeta Controlador "Controller". Aquí, se deben configurar los módulos que están instalados en el PLC, especificando el modelo y el número de ranura que ocupan. Se debe escoger, también, el modelo de la fuente instalada en el bastidor. La configuración de los componentes del PLC es muy importante, pues identifica las direcciones de entrada o salida válidas, y, dependiendo del CPU instalado, las funciones que puede realizar el PLC.
4. En el menú "Comms" y en la opción Comms de sistema "System Comms" se debe escoger el Nodo de Procesador "Processor Node" definido en el paso 2. Además, en última configuración "Last Configured" se debe escoger el nombre de la estación con que se está

trabajando según RSLinx. Después se debe presionar Aplicar "Apply". Si la configuración está bien realizada, en el espacio Manipulador "Driver" deberá aparecer: AB_DF1-1.

5. Ahora se puede proceder a crear el *programa* que especificará el *algoritmo de control.* El área de programación esta a la derecha, en la ventana del proyecto. Para introducir instrucciones de *programa* en una línea, ésta se debe seleccionar, marcándola en rojo. Las instrucciones están disponibles en una barra deslizable sobre el área del programa. Estas instrucciones serán colocadas en la fila seleccionada en el orden que el usuario las seleccione.
6. Una vez que se ha terminado de introducir la lógica del *algoritmo de control*, se debe corroborar que no se han cometido errores de sintáxis. Para ello, se puede seleccionar la opción verificar programa "verify program" en el menú Editar "Edit".
7. Si el *programa* no tiene errores, puede ser guardado con un nombre adecuado en un directorio del PC, bajo la extención ".RSS". Una vez guardado se descarga al PLC. Para ello conecte el PLC a la computadora, mediante el cable serial. En el botón que dice Fuera de linea "OFFLINE" en la esquina superior izquierda de la ventana del proyecto, escoja la opción Descargar "DOWNLOAD". El *programa* será transferido al PLC. Una vez terminada la descarga, RSLogix preguntará si se quiere Estar en linea "go on line", esto significa que el *programa* puede ser monitoreado y modificado desde el PC. Si escoge NO, la computadora queda desconectada del PLC. Si se escoge SI "YES", la función en línea queda activa.

8. Una vez que todo el equipo ha sido conectado y revisado, se puede poner en marcha la ejecución del *programa*. Para ello se debe girar la llave en el panel frontal del PLC a la posición CORRER "RUN".

3.2.2.1 Elementos Básicos de los *programas* escritos en la aplicación RSLogix.

El *programa* contiene el algoritmo que utiliza el PLC para obtener las salidas a partir de una configuración de entradas dada.

La serie de procesadores SLC 500 tiene un set de instrucciones, codificadas en lenguaje escalera, que especifican todas las operaciones que el PLC puede realizar. Los objetos sobre los que las instrucciones actúan se llaman elementos de lenguaje. Estos elementos residen en la memoria interna del PLC. La memoria está estructurada en archivos, cada uno con una capacidad de 256 elementos. Cada elemento está formado de 16 bits, 32 bits o 48 bits dependiendo del tipo de archivo. Existen archivos de datos y de programa.

3.2.2.1.1 Elementos de Lenguaje.

Dependiendo del tipo, los archivos de dato se pueden clasificar en: salida, entrada, estado, bit, temporizador, contador, control, entero y punto flotante. Existen 256 archivos de dato disponibles. El tipo de archivo específico es definido por el usuario, excepto para los primeros 9, los cuales están definidos con un tipo predeterminado. La forma en que el CPU direcciona un elemento dentro de un archivo depende de la clase de archivo que se trate. Un elemento, entonces, es caracterizado por el archivo en donde se encuentra. Para el caso de los archivos de programa existen solamente dos tipos: de sistema y de programa.

Archivos de salida y entrada, elementos de salida y entrada.

Los archivos 0 y 1 son los de tipo salida y entrada respectivamente. Estos dos tipos no pueden ser asignados a otros archivos. Estos son las imágenes de las entradas y salidas del PLC. Cada ranura del bastidor del PLC tiene asignado un elemento de estos archivos. Cada elemento consta de 16 bits, y cada bit representa un borne de conexión en el módulo. En el caso de las entradas/salidas analógicas, se asigna un elemento a cada entrada o salida.

Direccionamiento de los elementos de archivos salida y entrada.

El formato de direccionamiento de los elementos dentro de estos archivos es como sigue:
O:e.s/b para las salidas.
I:e.s/b para las entradas.

El significado de cada parte de la dirección es:

O: salida.

I: entrada.

e: se refiere al número de ranura donde se encuentra el módulo que se quiere direccionar.

s: número de elemento. Esto es necesario sólo si el número de entradas/salidas excede 16.

b: número de terminal. Es el borne específico que se quiere direccionar. Van de 0-15.

Archivos de estado, elementos de estado.

El archivo de datos número 2 y los de programa 0 y 1 son de tipo estado. Ningún otro archivo puede ser definido como de estado. Estos archivos son manipulados exclusivamente por el CPU, por lo tanto no pueden ser modificados ni borrados por el *programa* de control. Sin embargo, poseen información valiosa para la toma de decisiones, por lo que se pueden leer. Cada elemento dentro de esos archivos representa alguna información.

Direccionamiento de los elementos de archivos de estado.

El formato de direccionamiento de los elementos dentro de estos archivos es como sigue:

S:e/b

El significado de cada parte de la dirección es:

S: archivo de estado.

e: se refiere al número elemento dentro del archivo. Para cada archivo se tiene un ámbito de 0 a 82.

b: número de bit. Localiza la información específica en caso de que ésta sea de la forma de bit. Van de 0-15.

Archivos de bit, elementos de bit.

El archivo de datos número 3 es de tipo Bit. Cualquier otro archivo de dato del 9 al 255 puede ser asignado como de tipo bit. Estos archivos son variables de apoyo para la lógica del

programa. Generalmente se utilizan como marcas internas de memoria que almacenan resultados parciales, condiciones de verdadero o falso, elementos de decisión para la ejecución de subrutinas, etc.

Direccionamiento de los elementos de archivos de bit.

Existen dos formatos para direccionar las marcas ó bits dentro de los archivos de bit:

Bf:e/b se direcciona el bit a través de la palabra.

Bf/b: se direcciona el bit directamente.

El significado de cada parte de la dirección es:

B: archivo de bit.

f: número de archivo. Por defecto 3. Se pueden definir los archivos de datos del 9-255 como de bit.

e: se refiere al número elemento dentro del archivo. Para cada archivo se tiene un ámbito de 0 a 255.

b: número de bit. Esta es la dirección de un bit en una palabra. Varía de 0-15 para el primer caso. Para el segundo caso, no es necesario especificar la palabra, y el ámbito es de 0 a 4095.

Archivos de temporizador, elementos de temporizador.

El archivo de datos número 4 es de tipo temporizador. Cualquier otro archivo de dato del 9 al 255 puede ser asignado de tipo temporizador. Este tipo de archivo es utilizado por el bloque de instrucción de temporizador. Este bloque consta de 3 palabras. La palabra 0 es la palabra de estado. La palabra 1 indica el valor preestablecido del temporizador y la palabra 2 guarda el valor actual de la cuenta.

Los bits de estado son EN, TT, DN, los cuales indican si el temporizador está habilitado, si esta contando o si ya terminó su cuenta, respectivamente.

Direccionamiento de los elementos de archivos de temporizador.

El formato de direccionamiento es el siguiente:

Tf:e.s/b

El significado de cada parte de la dirección es:

T: archivo tipo temporizador.

f: número de archivo de datos. Por defecto 4. Sin embargo cualquier archivo del 9 al 255 puede ser definido de tipo temporizador.

e: número de elemento. Rango 0-255. Cada elemento es de 3 palabras.

s: número de palabra dentro del elemento.

b: número de bit. Rango de 0-15.

Archivos de contador, elementos de contador.

El archivo de datos número 5 es de tipo contador. Cualquier otro archivo de dato del 9 al 255 puede ser asignado como de tipo contador. Este tipo de archivo es utilizado por las instrucciones de contador. Cada dirección de contador está formada de un elemento tipo contador de 3 palabras. La palabra 0 es la palabra de estado y contiene los bits de estado de la instrucción. La palabra 1 indica el valor preestablecido del contador y la palabra 2 guarda el valor acumulado de sucesos (transiciones de 0 a 1).

Los bits de estado son:

CU: contador incremental habilitado.

CD: contador decremental habilitado.

DN: cuenta alcanza valor preestablecido.

OV: bit de rebase.

UN: bit de cuenta baja.

UA: bit de actualización de acumulador.

Direccionamiento de los elementos de archivos de contador.

El formato de direccionamiento es el siguiente:

Cf:e.s/b

El significado de cada parte de la dirección es:

C: archivo tipo contador.

f: número de archivo de datos. Por defecto 5. Sin embargo cualquier archivo del 9 al 255 puede ser definido como tipo contador.

e: número de elemento. Rango 0-255. Cada elemento es de 3 palabras.

s: número de palabra dentro del elemento. Rango de 0-2.

b: número de bit. Rango de 0-15.

Archivos de control, elementos de control.

El archivo de datos número 6 es de tipo control. Cualquier otro archivo de dato del 9 al 255 puede ser asignado como de tipo control. Los elementos de este tipo de archivo constan de 3 palabras que son utilizadas por las instrucciones FIFO, LIFO, ABL, ACB, AHL, ARD, AWA, y AWT. La palabra 0 es la palabra de estado y contiene los bits de estado de la instrucción. La palabra 1 indica la longitud del dato guardado y la palabra 2 indica la posición.

Direccionamiento de los elementos de archivos de control.

El formato de direccionamiento es el siguiente:

Rf:e.s/b

El significado de cada parte de la dirección es:

R: archivo tipo control.

f: número de archivo de datos. Por defecto 6. Sin embargo cualquier archivo del 9 al 255 puede ser definido como tipo control.

e: número de elemento. Ambito 0-255. Cada elemento es de 3 palabras.

s: número de palabra dentro del elemento. Ambito de 0-2.

b: número de bit. Ambito de 0-15.

Archivos de número entero, elementos de número entero.

El archivo de datos número 7 es de tipo número entero. Cualquier otro archivo de dato del 9 al 255 puede ser asignado como de tipo número entero. Los elementos de este tipo de archivo constan de 1 palabra y son utilizados por el programa del usuario, en cada aplicación en la que se necesite almacenar temporalmente un dato de 16 bits. El direccionamiento puede ser a nivel de elemento (palabra) o a nivel de bit.

Direccionamiento de los elementos de archivos de número entero.

El formato de direccionamiento es el siguiente:

Nf:e/b

El significado de cada parte de la dirección es:

N: archivo tipo número entero.

f: número de archivo de datos. Por defecto 7. Sin embargo cualquier archivo del 9 al 255 puede ser definido como tipo número entero.

e: número de elemento. Ambito 0-255. Cada elemento es de 1 palabra.

b: número de bit. Ambito de 0-15.

Archivos de número en coma flotante, elementos de número en coma flotante.

El archivo de datos número 8 es de tipo número en coma flotante. Cualquier otro archivo de dato del 9 al 255 puede ser asignado como de tipo número en coma flotante. Los elementos de este tipo de archivo constan de 2 palabras. Son utilizados por el programa del usuario en cada aplicación en la que se necesite almacenar un dato de 32 bits, para lograr mayor precisión en los resultados. El direccionamiento puede ser a nivel de elemento (palabra).

Direccionamiento de los elementos de archivos de número en coma flotante.

El formato de direccionamiento es el siguiente:

Ff:e

El significado de cada parte de la dirección es:

F: archivo tipo número en coma flotante.

f: número de archivo de datos. Por defecto 8. Sin embargo cualquier archivo del 9 al 255 puede ser definido como tipo número en coma flotante.

e: número de elemento. Ambito 0-255. Cada elemento es de 2 palabras.

3.2.2.1.2 Conjunto de instrucciones de la familia SLC 500.

El lenguaje de escalera es el más comúnmente utilizado en la programación de los PLC, por ello es de inclusión obligatoria por los fabricantes en el conjunto de lenguajes de programación de sus PLCs. Esto es debido a su similitud con los diagramas de contactores en el control convencional. De esta forma el usuario es capaz entender rápidamente la lógica del programa.

El lenguaje se estructura de la siguiente forma. Se compone de una serie de filas, cada una se ejecuta una sola vez. El estado de las variables que se consultan, a la hora de procesar una fila, es el que resulta de la lógica de las filas anteriores, a excepción de las variables de archivo de salida y entrada, las cuales se actualizan hasta el final de la pasada actual del programa.

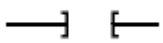
Una fila puede tener hasta 5 instrucciones de bit básicas, y una instrucción de bloque avanzado. Si se utilizan las instrucciones de bit, en una fila deben estar presentes al menos 2 instrucciones, una de tipo consulta y otra de tipo salida. Si se utilizan instrucciones de bloques avanzados, no es necesario que existan instrucciones de bit tipo consulta o tipo salida.

1- Instrucciones básicas.

Las instrucciones básicas se caracterizan por actuar sobre bits de elementos tipo bit.

A continuación se presentan las instrucciones básicas más comunes.

XIC: Esta instrucción consulta el estado de un bit. Si el bit es 1, la instrucción es evaluada como 1, si el bit es 0, la instrucción es 0. El símbolo escalera se muestra a continuación:

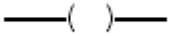


El direccionamiento del bit a consultar se realiza al escribir la dirección del mismo en el espacio habilitado para tal fin en el editor de programa.

XIO: Esta instrucción consulta el estado de un bit. Si el bit es 0, la instrucción es evaluada como 1, si el bit es 1, la instrucción es 0. El símbolo escalera se muestra a continuación:

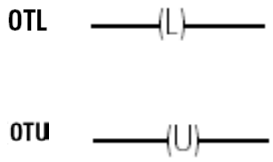
El direccionamiento del bit a consultar se realiza al escribir la dirección del mismo en el espacio habilitado para tal fin en el editor de programa.

OTE: Esta instrucción establece el bit direccionado si las instrucciones de consulta de la misma fila son evaluadas como 1. En caso contrario, el bit permanece con su valor en 0. El símbolo escalera se muestra a continuación:



El bit que fue puesto es 1 por esta instrucción permanecerá así hasta que las condiciones que están siendo evaluadas cambien, de tal forma que las instrucciones de consulta sean 0.

OTL y OTU: La instrucción OTL establece el bit direccionado si las instrucciones de consulta de la fila en que se encuentra son 1, sin embargo esta instrucción sólo es capaz de establecer el bit, por lo que el mismo permanecerá en 1, aun si las instrucciones de consulta cambian su estado a 0. La instrucción OTU reestablece el bit direccionado si las instrucciones de consulta son 1. Al igual que la anterior esta sólo puede reestablecer el bit. Por su naturaleza, estas instrucciones se utilizan en pareja direccionando el mismo bit. El símbolo escalera para cada instrucción se muestra a continuación:



OSR: esta instrucción es de tipo consulta y es 1 si las instrucciones de consulta de su fila cambian de 0 a 1. La instrucción permanece en 1 sólo durante un ciclo de programa, pues la instrucción evalúa sólo el cambio de las condiciones de consulta de 0 a 1, no el estado 1 en sí. El símbolo de la instrucción es:

Esta instrucción puede ser seguida de instrucciones de salida, así como de más instrucciones de consulta y bloques de función.

2- Instrucciones de bloque de función avanzado.

Instrucciones de temporizador

Estas instrucciones tienen los siguientes parámetros.

ACC: Es un registro que lleva el tiempo desde que la instrucción fue reestablecida por última vez.

PRE: es el valor que el temporizador debe alcanzar para establecer el bit DN.

Base de tiempo "**Timebase**": La cuenta se aumenta cada vez que un periodo igual a la base de tiempo pasa.

TON: Esta instrucción establece o reestablece una salida de bit cuando el valor en el registro **PRE** ha sido alcanzado. La instrucción empieza a contar a partir de que las instrucciones de consulta de su fila son 1. Sin embargo la función es reestablecida y el valor de cuenta actual también si las instrucciones de consulta cambian a 0.

Los bits de estado de la instrucción son:

DN: se establece si el valor actual de cuenta es igual o mayor al valor preestablecido.

TT: se establece si las condiciones de la fila son 1 y el valor actual de la cuenta es menor al del valor preestablecido.

EN: se establece si las condiciones de la fila del bloque son 1.

TOF: Esta instrucción establece o reestablece una salida de bit cuando el valor en el registro **PRE** ha sido alcanzado. La instrucción empieza a contar a partir de que las instrucciones de consulta de su fila son 0. Sin embargo la función es reestablecida y el valor de cuenta actual también, si las instrucciones de consulta cambian a 1.

Los bits de estado de la instrucción son:

DN: se reestablece si el valor actual de cuenta es igual ó mayor al valor preestablecido.

TT: se establece si las condiciones de la fila son 0 y el valor actual de la cuenta es menor al del valor preestablecido.

EN: se establece si las condiciones de la fila del bloque son 1.

Instrucciones manipulación de datos.

MOV: ésta es una instrucción de salida que mueve el dato de la dirección de fuente a la dirección de destino, si las instrucciones de consulta de su fila son 1.

Los parámetros de este bloque son:

Fuente "source": Aquí debe especificarse la dirección de la palabra que contiene el dato que se quiere mover. También es posible que este parámetro sea una constante.

Destino "Dest": Esta es la dirección donde se moverá el dato.

MVM: Esta es una instrucción que mueve un dato de una dirección fuente a una dirección destino y que además permite que el dato sea enmascarado por una palabra separada si las instrucciones de consulta de su fila son 1. Los parámetros de este bloque son:

Fuente "source": Aquí debe especificarse la dirección de la palabra que contiene el dato que se quiere mover. También es posible que este parámetro sea una constante.

Destino "Dest": Esta es la dirección donde se moverá el dato.

Máscara "Mask": Es la dirección de la palabra que contiene la máscara con la cual se enmascarará el dato de la fuente. Este parámetro puede ser una constante.

Los bits en la fuente que coinciden con ceros en la máscara son ignorados de tal forma que esos bits en el contenido del destino no son alterados. Los bits en la fuente que coinciden con unos en la máscara son copiados sin cambio en la palabra de destino.

Instrucción proporcional, integral, derivativa.

Este bloque de instrucción aplica el algoritmo PID sobre variables internas tipo entero de 16 bits. Se utiliza generalmente en lazos cerrados de control de procesos de variables físicas como temperaturas, presiones, niveles o flujos.

Las variables del sistema de control en lazo cerrado que son monitoreadas por el controlador son: la variable controlada (VC) y el valor deseado (VD). Así mismo, la salida de controlador (SC) manipulará el actuador del sistema de control.

La figura 3.2 muestra el control del nivel de un tanque mediante un lazo cerrado en modo servomecanismo, donde el controlador es el PLC.

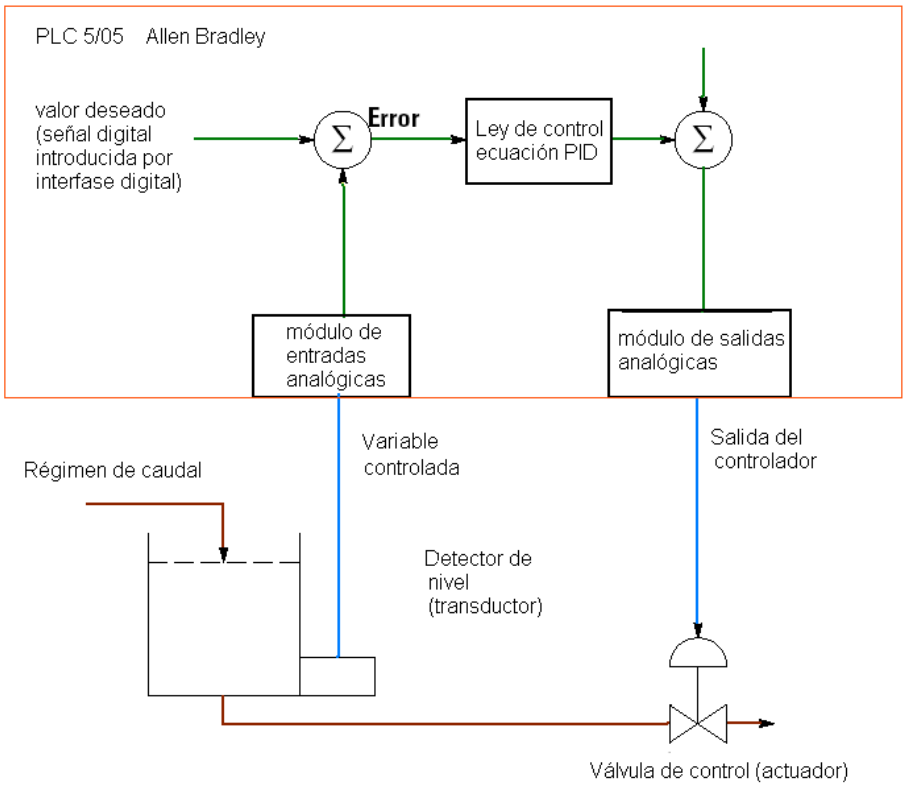


Figura 3.2 Sistema de control de lazo cerrado con controlador digital.

Es importante notar que las variables físicas son medidas y transformadas a señales eléctricas por los transductores, que sirven como parte del enlace entre el sistema hidráulico y el controlador digital. Los dos transductores que se muestran son: la válvula de control y el sensor de nivel. Todas las señales digitales están definidas dentro del controlador, los módulos de entradas y salidas analógicas realizan las conversiones de analógico a digital. El valor de consigna ó valor deseado es una señal digital, la cual es manipulada por el usuario mediante una interface hombre-máquina.

Esta instrucción necesita de un bloque de 23 palabras de un archivo tipo entero para operar debidamente. Ese bloque contiene los valores de resultados parciales que la instrucción obtiene mientras realiza la operación PID. Además contiene palabras con parámetros del bloque y bits de control.

La salida de la instrucción es escrita como un elemento tipo número entero. Este elemento puede estar en el mismo archivo donde está el bloque de parámetros, pero no puede ser parte de él. Este elemento es la señal de salida del controlador SC, por ello al final del programa debe ser movido a la imagen de salidas del PLC.

El valor consigna VD es una palabra de 16 bits creada en una dirección en el mismo bloque de control.

La variable controlada VC reside en un elemento tipo entrada correspondiente al módulo de entradas analógicas y debe ser movida a un elemento de 16 bits tipo entero en el mismo archivo donde está el bloque de control de la instrucción PID.

Un valor adicional de nivel dc puede ser sumado a la señal de salida de la algoritmo de control.
La ecuación PID que es implementada por el PLC es la siguiente.

$$\text{"Salida del controlador"} := KC \cdot \left[E + \left(\frac{1}{TI} \right) \cdot \int E\,dt + TD \cdot \frac{d}{dt} VC \right] + \text{"valor dc"} \qquad (3.3)$$

con E=VD-VC.

Aquí se nota que el modo derivativo es aplicado sólo a la variable controlada. Los términos KC, KC/TI y KC*TD son las ganancias de los modos proporcional, integral y derivativo, respectivamente.

Los parámetros del bloque de función son: ganancia del controlador KC, término de reestablecimiento TI, término de razón de cambio TD. La relación de éstos con los términos ganancia proporcional K_c, tiempo integral T_i y tiempo derivativo T_d en un algoritmo PID ideal es:

KC (sin unidades) = K_c (sin unidades)

TI (minutos) = T_i / 60 (T_i en segundos)

TD(minutos) = T_d / 60 (T_d en segundos)

Los ámbitos de los parámetros del bloque de función se muestran en la tabla 3.3.

Constante	Límites
KC	0.01-327.67
TI	0.01-327.67 (minutos)
TD	0.01-327.67 (minutos)

Tabla 3.3 Los ámbitos de los parámetros KC, TI y TD del bloque de función PID.

La distribución de las palabras del bloque de parámetros de esta función se muestra en la Figura 3.3

Al programar esta función en el lenguaje escalera, se coloca el bloque de función en una fila sin instrucciones de consulta. Además se introduce en la instrucción la dirección del bloque de parámetros, la variable controlada y la salida del controlador.

Word 0	EN		DN	PV	SP	LL	UL	DB	DA[1]	TF	SC	RG[1]	OL[2]	CM[2]	AM[2]	TM[2]
Word 1	PID Sub Error Code (MSbyte)															
Word 2	Setpoint SP															
Word 3	Gain K_c															
Word 4	Reset T_i															
Word 5	Rate T_d															
Word 6	Feed Forward/Bias															
Word 7	Setpoint Max (SMax)															
Word 8	Setpoint Min (SMin)															
Word 9	Deadband															
Word 10	Internal Use Do Not Change															
Word 11	Output Max															
Word 12	Output Min															
Word 13	Loop Update															
Word 14	Scaled Process Variable															
Word 15	Scaled Error SE															
Word 16	Output CV% (0 to 100%)															
Word 17	MSW Integral Sum															
Word 18	LSW Integral Sum															
Word 19	Internal Use Do Not Change															
Word 20	Internal Use Do Not Change															
Word 21	Internal Use Do Not Change															
Word 22	Internal Use Do Not Change															

Figura 3.3 Bloque de parámetros del bloque de función PID.

La modificación de los parámetros del bloque de función puede ser realizada mediante lógica de escalera, o mediante la ventana de parámetros del bloque de función, la cual puede ser desplegada dando doble click izquierdo en bloque de función. La figura 3.4 muestra la ventana de parámetros.

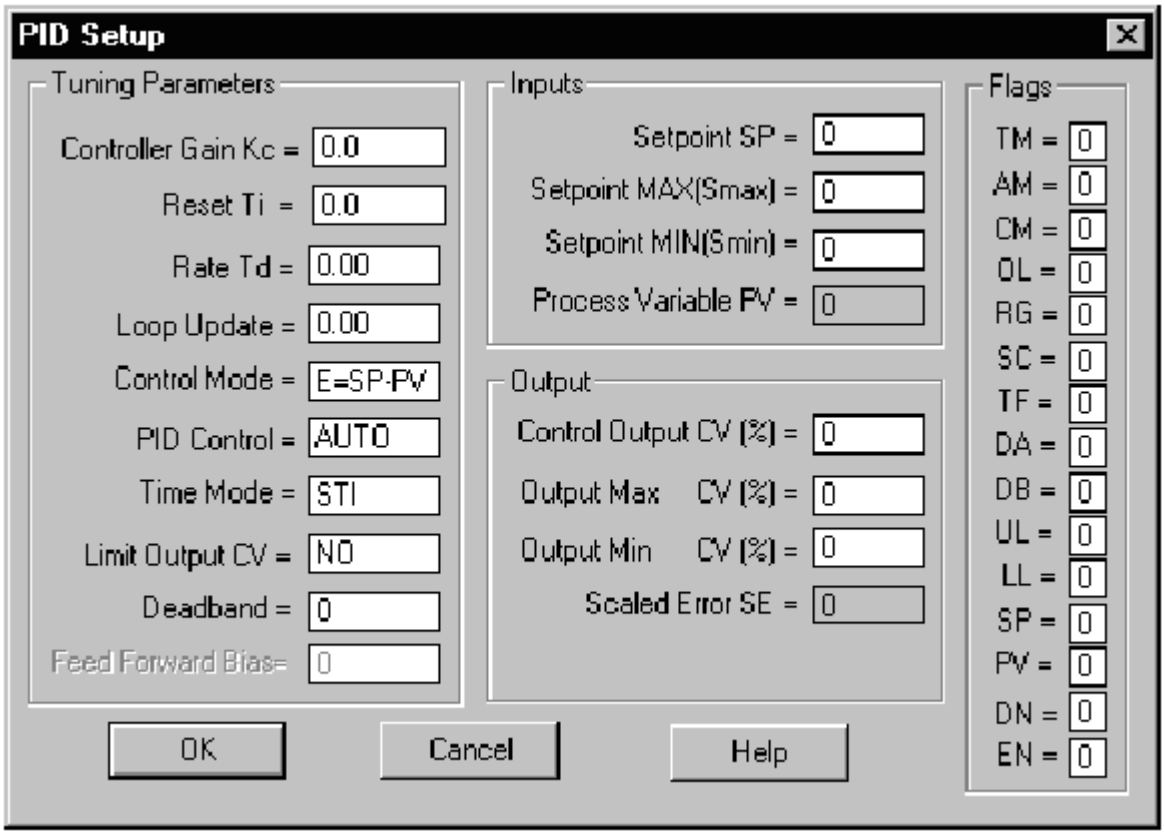


Figura 3.4 Ventana de parámetros del bloque de función PID.

A continuación se presenta una explicación de los parámetros del cuadro anterior.

Ganancia del controlador "Controller Gain" KC: es la ganancia del controlador. Ambito 0-327.67 ó 0-32767 en decimal.

Término de reestablecimiento "Reset Term" TI: es la ganancia del modo integral. Ambito 0-327.67 repeticiones por minuto ó 0-32767 en decimal.

Término de régimen "Rate term" TD: es la ganancia del modo derivativo. Ambito 0-327.67 minutos ó 0-32767 en decimal.

Valor dc "Feed Forward/Bias": es un nivel dc que se suma a la salida del controlador. Ambito: -16383-16383.

Modo "Mode" TM: determina el modo de actualización de la salida del controlador. Estados: 0 (modo STI) - 1 (modo temporizado). El modo STI implica que la salida se actualizará cada vez que el programa del controlador ingrese a la subrutina donde está definida la instrucción PID. En el modo temporizado, el bloque PID actualiza la salida cada vez que el intervalo de tiempo que especifica el parámetro **actualización del bloque de función PID "Loop Update"** ha pasado. Asegúrese que este intervalo de tiempo sea diez veces mayor que el tiempo de ciclo del programa del PLC.

Actualización del bloque de función PID "Loop Update": es el intervalo de tiempo que debe pasar entre cada actualización de la salida del Bloque de función PID. Ambito 0.01-10.24 segundos ó 1-1024 en decimal.

Error "Scaled error": es el valor de la señal E, la diferencia entre la variable controlada y el valor consigna. Ambito -32768-32767 decimal.

Auto/Manual AM: Especifica si la salida del bloque de control es manipulada por el algoritmo PID (Auto) o por el programa del usuario (Manual). Estados: 1(Manual)-0(Auto).

Control "Control" CM: Especifica cómo se calcula la variable E. Estados: 0 (E=VD-VC)-1 (VC-VD).

Capítulo 4: Implementación del proyecto de control

El proyecto consiste en controlar un proceso mediante un PLC, en un lazo realimentado. Se utilizará el algoritmo PID para lograr que la variable controlada del proceso cumpla con ciertas especificaciones. El proceso será una planta electrónica analógica constituida por amplificadores operacionales. La salida del controlador se acopla directamente a la planta, y la señal realimentada es igual a la variable controlada. El sistema se comportará como un servomecanismo, en el que la variable controlada (la salida de la planta electrónica) seguirá el valor de la señal de consigna.

Se fabricará una interfece al ususario que correrá en un PC utilizando la aplicación RSView32, programa propiedad de Allen Bradley. Mediante esta interface, el usuario podrá variar los parámetros del controlador y monitorear las señales: variable controlada y salida del controlador. Además, la señal de consigna será transmitida al PLC a través de esta interface.

En este capítulo, la implementación se separará en dos etapas. En la primera etapa se realizarán las conexiones necesarias entre los equipos. Estás conexiones se dividen en control y comunicación. En la segunda etapa, se especificará la configuración del equipo, programación del PLC y creación de la interface al usuario. En esta etapa se debe definir cómo se distribuirá la memoria del PLC, a cuales de los registros del bloque de función PID tendrá acceso el usuario y que funciones de control se tendrán sobre el proceso.

En el apartado 4.3 se realiza la sintonización de los parámetros del algoritmo PID. Se modela el sistema matemáticamente, con el fin de comparar los resultados experimentales.

4.1 Etapa 1: Conexión del equipo.

Se presenta a continuación las conexiones necesarias para implementar el proyecto.
Las características de la planta electrónica se presentan a continuación:
Señal de entrada: 0-10V.
Señal de salida: 0-10V.

Alimentación externa: 110V*ac*. 60Hz.

La constante de tiempo de la planta, así como su retardo de transporte, pueden ser escogidos mediante interruptores en su panel forntal.

El equipo específico que se utilizará es el siguiente:

Nombre del Equipo	Marca	Modelo	Cantidad
Procesador SLC 5/05	**Allen Bradley**	**1747-L551 1**	**1**
Bastidor	**Allen Bradley**	**1746-A7**	**1**
Fuente de alimentación	**Allen Bradley**	**1746-P2**	**1**
Módulo de entradas digitales	**Allen Bradley**	**1746-IA16**	**1**
Módulo de salidas digitales	**Allen Bradley**	**1746-OW16**	**1**
Módulo de entradas/salidas analógicas	**Allen Bradley**	**1746-NIO4I**	**1**
Cable PLC-PC	**Allen Bradley**		**1**
Planta electrónica			**1**
Amplificador operacional	**National**	**LM358**	**1**
Resistencias		**470Ω**	**2**

Tabla 4.1 Equipo a utilizar en el proyecto.

La figura 4.1 muestra la forma en que se deben conectar entre sí los equipos utilizados.

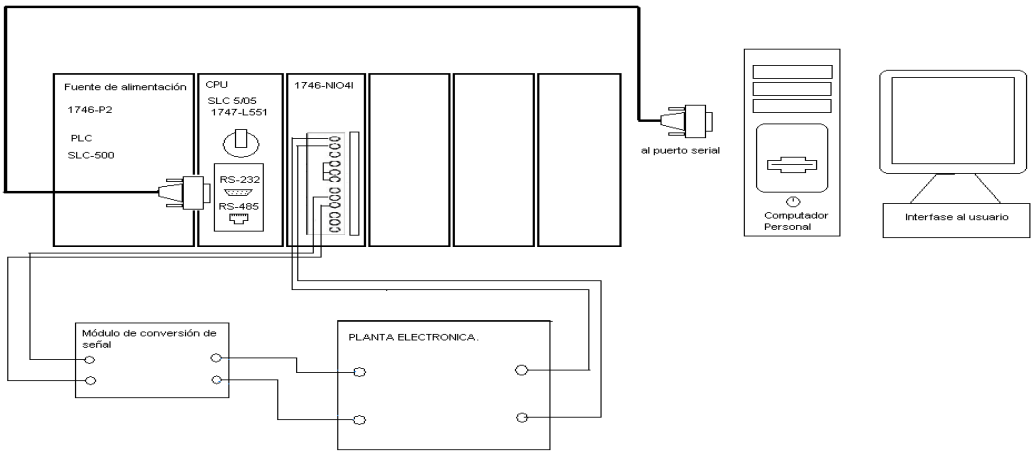


Figura 4.1 Conexión del equipo necesario para realizar el control sobre la planta proceso.

El modelo matemático de esta planta es:

$$Gp(s) := \frac{e^{-tm \cdot s}}{(\tau_1 \cdot s + 1)\cdot(\tau_2 \cdot s + 1)} \tag{4.1}$$

donde $\mathbf{t_m}$, $\boldsymbol{\tau_1}$ y $\boldsymbol{\tau_2}$ representan el tiempo muerto y las constantes de tiempo respectivamente.

El bloque **módulo de conversión de señal,** transforma la señal de corriente, que proviene del módulo de salida del PLC, en una señal de voltaje que pueda ser interpretada por la planta. El diagrama esquemático de este bloque se muestra a continuación:

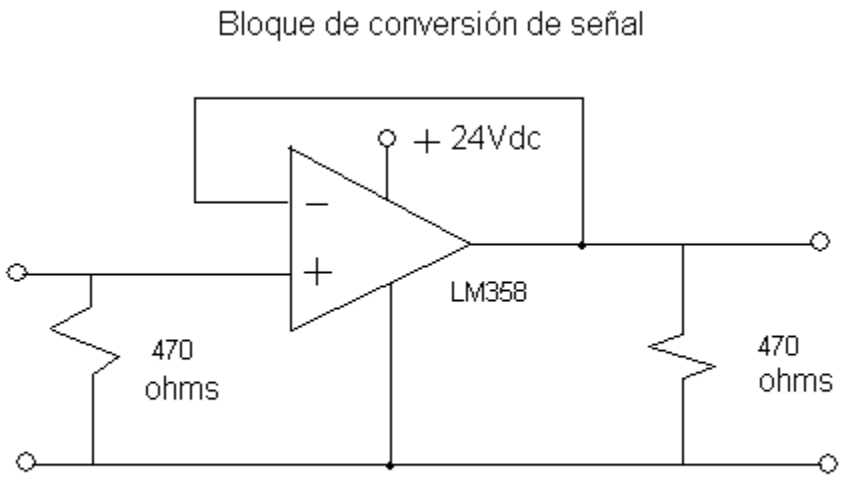


Figura 4.2 Bloque de conversión de señal.

Como se puede notar en la figura anterior es necesario alimentar con +24V*dc* al amplificador operacional LM358.

Como la salida del módulo de entradas/salidas analógicas puede variar entre 0 y 10 mA, la salida del bloque de conversión de señal variará de 0 a 4.7 V*dc*.

4.2 Etapa 2: Configuración del equipo.

En este apartado se realizará la configuración necesaria para poner en marcha un programa en el PLC. Para realizar esto, es necesario utilizar los programas RSLinx y RSLogix de Rockwell Software. RSLinx es el primero que se debe ejecutar, pues éste permite la cominicación entre el PLC y la computadora personal. RSLogix es la herramienta de programación del PLC. Mediante este programa se puede crear el *algoritmo de control* deseado y descargarlo al PLC. RSView32 es necesario para realizar la interface al ususario.

Antes de programar el PLC, es necesario establecer ciertos aspectos de la estructura de comunicaciones, opciones de control y estructura de memoria de las palabras y bits de la memoria interna del PLC, ya que esta información es la base del *programa* del PLC.

4.2.1 Aspectos preliminares.

4.2.1.1 Comunicaciones.

Las comunicaciones se establecen en el protocolo de capa física RS-232. La conexión sigue una topología punto a punto entre el puerto 1 del PLC y el puerto serie del computador.
Parámetros del protocolo RS-232.
Velocidad: 19200 baudios.
Bits de datos: 8.
Paridad: No.
Bits de parada: 1.
Fin de trama: 10.
Time out respuesta: 1s.
Time out de trama: 4ms.

4.2.1.2 Opciones de control.

Se tendrá un control mediante la interface creada en RSView32. El usuario podrá sintonizar los parámetros del PID; cambiar el valor consigna, ya sea continuamente ó en un cambio tipo

escalón de 0 V a 2.5 V, y monitorear el valor de la variable controlada y de la salida del controlador.

4.2.1.3 Estructura de memoria.

La forma en que se estructura la memoria interna del PLC se muestra en la tabla 4.2. donde se especifica la dirección de la variable, el tipo de dato y una breve descripción del uso que se le da en el *programa*.

Dirección	Tipo de archivo	Tipo de dato	Descripción
B11:0/0	BIT	BIT	Incremento de valor consigna
B11:0/0	BIT	BIT	Decremento de valor consigna
N10:1	Entero	Entero	Vector de valor -1
N10:0	Entero	Entero	Valor temporal de consigna
N7:0-N7:31	Entero	Entero	Bloque de parámetros de función PID
N7:30	Entero	Entero	Variable controlada
N7:31	Entero	Entero	Salida del controlador
I:2.0	Entrada	Entrada	Imagen de entrada. Variable controlada
O:2.0	Salida	Salida	Imagen de salida. Salida del controlador

Tabla 4.2 Estructura de memoria.

4.2.2 Configuración de las comunicaciones mediante el programa RSLinx.

Para poder tener una comunicación entre el PLC y el computador donde se utilizarán RSLogix y RSView32, se debe ejecutar y configurar RSLinx. Para ello, abra RSLinx mediante el vínculo en el apartado ROCKWELL SOFTWARE, en el submenú programas del menú inicio en Windows.

NOTA: si está utilizando su PC en una red de area local o está utilizando la plataforma Windows XP, ejecute primero el Panel de control de servicio RSLinx "RSLinx Service Control Panel" presionando Iniciar "Start" en la ventana que se despliega, para ejecutar después RSLinx.

Seguidamente se debe configurar el controlador de la comunicación entre el PC y el PLC. En el menú Comunicaciones "Communications" se debe escoger la opción Configurar controlador "Configure Driver". En el recuadro controladores disponibles "Available Drivers" escoger el controlador RS-232 DF1 Devices, presionando agregar nuevo "Add New".

En la pantalla que se despliega, configure lo siguiente:

Nombre del artefacto "Device Name": AB_DF1-1.
Puerto Comm "Comm Port": COM1.
Artefacto "Device": SLC-CH0/Micro/PanelView.
Taza de baudios "Baud Rate":19200.
Número de Estación "Station Number": 00.
Paridad "Parity": None.
Revisión de Error "Error Checking": BCC.
Bits de Parada "Stop Bits": 1.
Protocolo "Protocol": Full Duplex.

Una vez hecho esto, presionar OK.

De esta manera queda configurada la aplicación RSLinx. El programa no debe cerrarse hasta que se termine de utilizar la comunicación entre el PLC y el PC.

4.2.3 Programación del PLC mediante la aplicación RSLogix.

Una vez creada la aplicación RSLinx, se puede tener una comunicación entre el PLC y la computadora. A continuación se presenta cómo programar el PLC mediante la aplicación RSLogix. Primero se debe crear un proyecto; para ello abra el programa RSLogix y cree un nuevo proyecto. En el cuadro de diálogo seleccione lo siguiente:

Manipulador "Driver": AB_DF1-1.
Nodo del procesador "Processor Node": 1.
Procesador "Processor": 1747-L551.

En configuración IO seleccione lo siguiente:
Bastidor: 1746-A7.
Fuente de alimentación*:* 1746-P2.
Procesador: SLC 5/05 1747-L551 en la ranura 0.
Módulo de entradas/salidas analógicas: 1746-NIO4I en la ranura 2.
Módulo de entradas digitales: 1746-IA16 en la ranura 5.
Módulo de salidas digitales:1746-OW16 en la ranura6.

Configuración de comunicación:
Manipulador "Driver": AB_DF1-1.
Ruta "Route": local.
Nodo del procesador: 1.
Ahora se puede escribir el *programa* del PLC mediante el editor de escalera. La figura 4.3 muestra el programa que debe escribirse para la implementación de este proyecto.

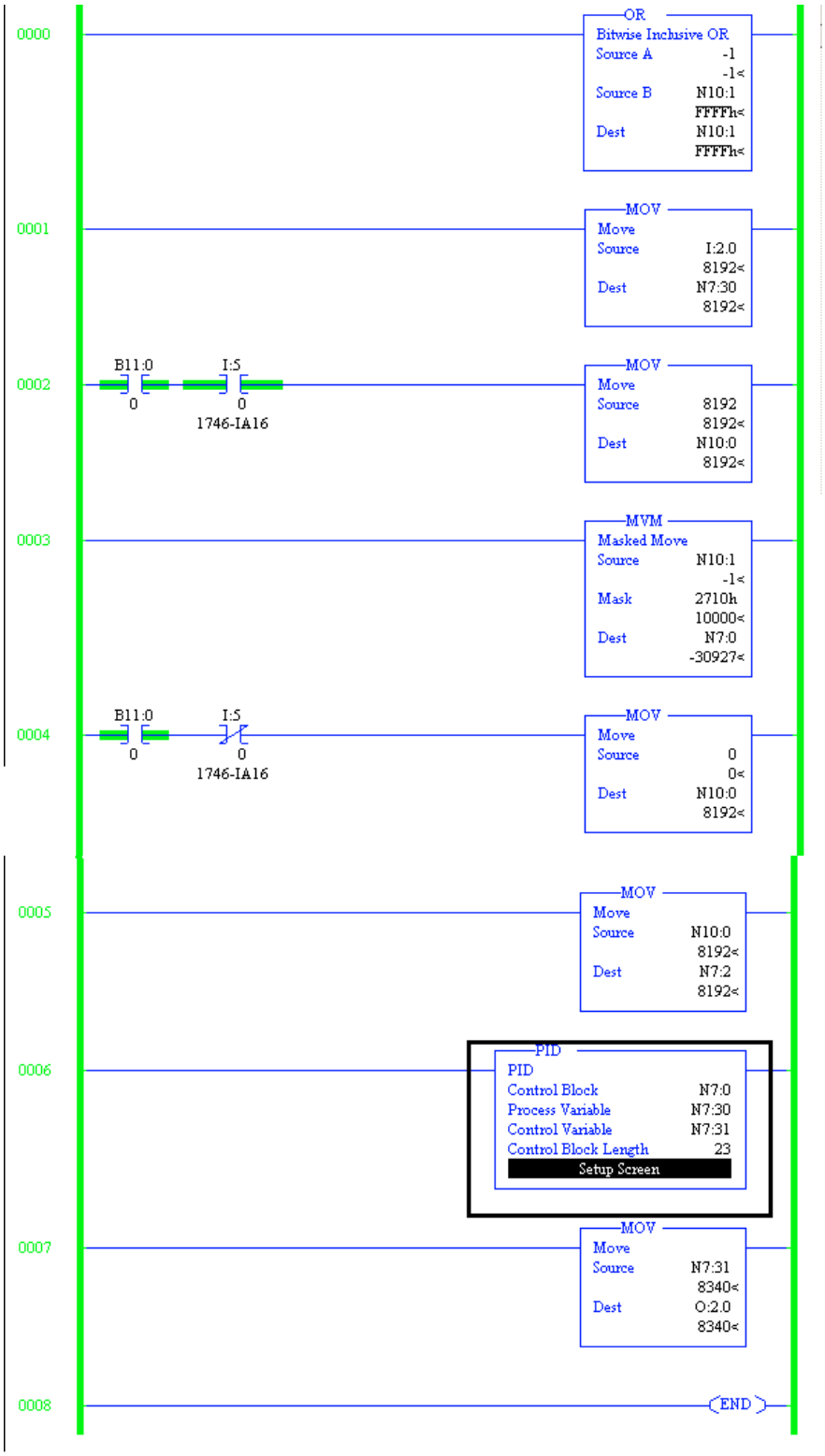


Figura 4.3 Programa del proyecto de control.

Una vez verificado el *programa*, el mismo se descarga al PLC.

4.2.4 Programación de la interface al usuario mediante RSView32.

Seguidamente se debe programar la interface al usuario. Con esta interface el usuario podrá modificar algunos parámetros del bloque de función de PID y monitorear los cambios en la variable controlada y la salida del controlador.

En el programa RSView, cree un nuevo proyecto:

1- En el menú principal, seleccionar la opción archivo “file” y crear un nuevo proyecto presionando sobre nuevo “New”. Aparecerá un bloque de diálogo donde se debe escribir el nombre del proyecto “project name” y el nombre del directorio “save in” donde se almacenará en la memoria del PC. Precione abrir “open”, una vez que se han completado los aspectos anteriores. Se presentará una ventana llamada de proyecto.

2- En esta ventana, abrir la carpeta sistema “system” y seleccionar canal “channel”. Aquí se debe configurara el canal de comunicaciones que se utilizará. En esta ventana, configurar lo siguiente:

Canal de comunicación “Channel”: 1-DH-485.

Tipo de red “Network Type”: DH-485.

Manipulador primario de comunicación “Primary Communication Driver”: AB_DF1-1.

Manipulador activo “Active Driver”: primario.

Una vez hecho esto, presione está bien “OK”.

3- En la misma carpeta sistema, seleccione la opción nodo “node” y configure:

Nombre del proyecto “Name”.

Canal “Channel”: 1-DH-485.

Estación “Station”: 01.

Tipo “Type”: SLC 5.

Ahora se deben crear las banderas que se utilizarán en el proyecto de RSView32. Cada bandera está relacionada a un elemento en el *programa* del PLC presentado anteriormente. La tabla 4.3 muestra las banderas que se deben crear.

Para crear estas banderas, seleccione la opción base de datos de banderas "tag database", en la carpeta sistema. Una vez que se han creado todas las banderas, se deben programar los elementos gráficos que conformarán la interface

Nombre	Tipo	Descripción	Dirección de memoria PLC	ámbito
SETPOINT	Entera	Valor deseado	N7:2	1-32767
VC	Entera	Variable controlada	N7:30	1-32767
SC	Entera	Salida del controlador	N7:31	1-32767
TI	Entera	Término de reestablecimiento	N7:4	1-32767
TD	Entera	Término de taza de cambio	N7:5	1-32767
KC	Entera	Ganancia del controlador	N7:3	1-32767
START	Digital	Modo escalón	B11:0/0	0-1
STEP	Digital	Escalón de 2.5Vdc	B11:0/1	0-1

Tabla 4.3 Banderas utilizadas en la aplicación RSView32.

.

Elemento gráfico 1.

Para crear este elemento seleccione la opción deplegar "display" en la carpeta gráficos "grafics" en la parte izquierda de la ventana del proyecto. Se desplegará una ventana que contiene el elemento gráfico. En la opción ventana de tendencia del menú de herramientas, cree un gráfico y relacione a éste la bandera *variable controlada*. La tendencia es en el tiempo.

La Figura 4.4 muestra como se visualiza este elemento

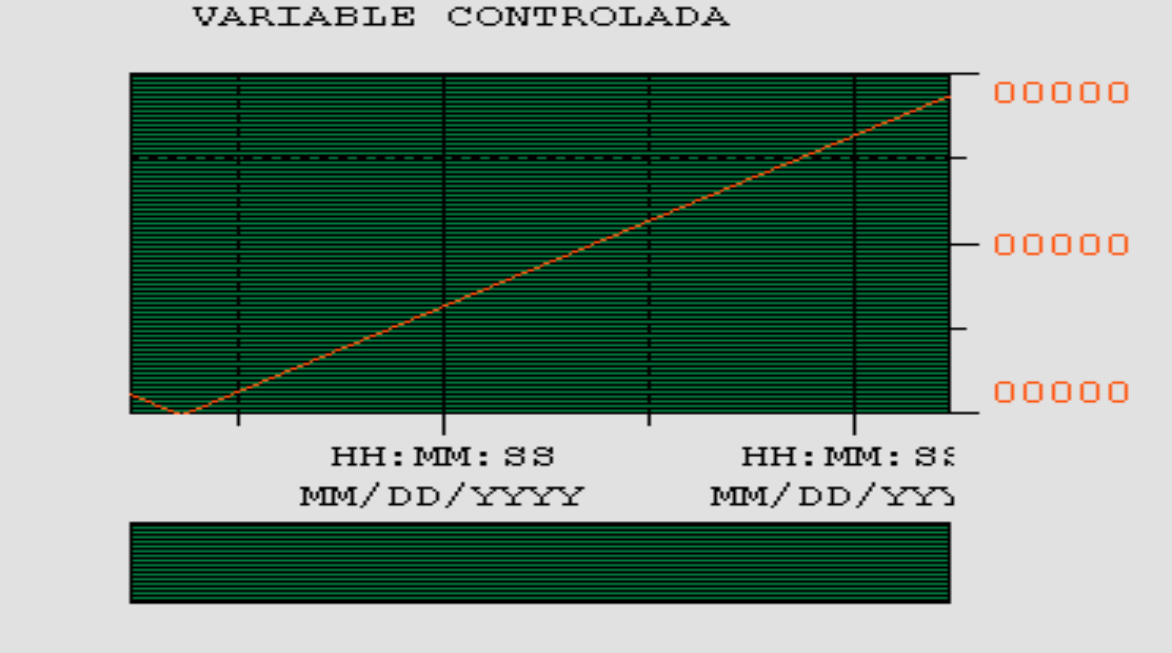


Figura 4.4 Elemento gráfico 1.

Elemento gráfico 2.

Siga los pasos daods para crear el elemento 1 y relacione al gráfico de tendencia la bandera *salida del controlador*. La tendencia es en el tiempo.

La Figura 4.5 muestra como se visualiza este elemento.

Elemento gráfico 3.

Siga los pasos para crear un elemento gráfico. Dentro de la ventana abierta configure el color de la ventana como gris. Cree cuatro deslizadores. Cada uno con una propiedad de animación por posición vertical. Asigne las banderas *TI, TD, KC*, y *SP* a cada deslizador. Asegúrese de

marcar el ámbito de la bandera según el ámbito del elemento relacionado en el programa del del PLC.

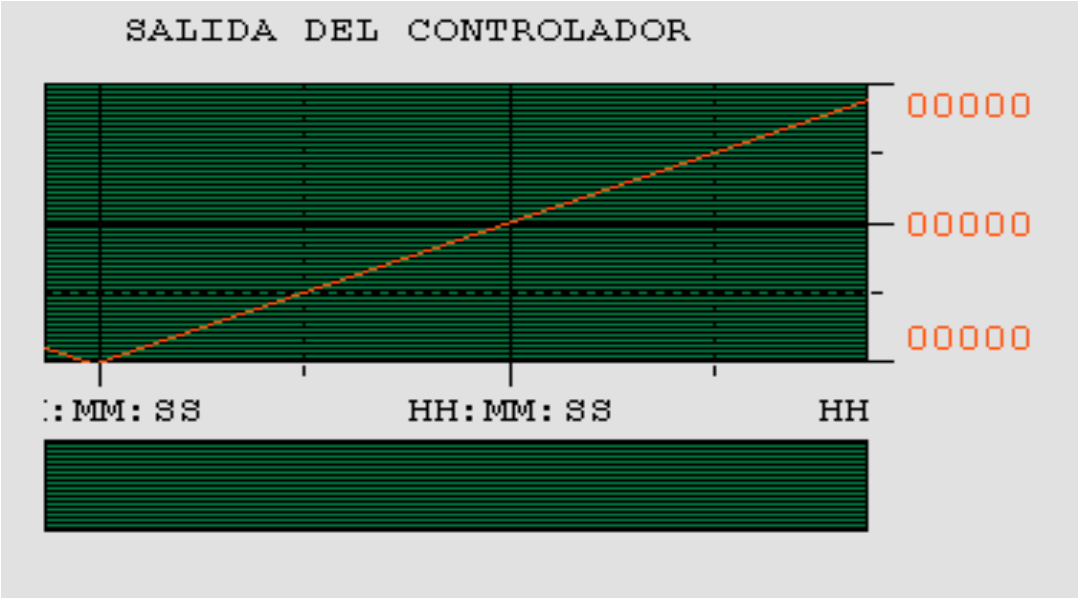


Figura 4.5 Elemento gráfico 2.

Cree dos botones con propiedad de abrir una ventana relacionada a un gráfico tipo "display" y relacione los elementos gráficos 1 y 2 a cada botón, de manera que al presionarlo se despliegue el elemento específico.

Además, se deben crear tres botones. El primero llevará escrito*: Comportamiento tipo escalón en el valor deseado 0v-2.5v*. Este botón se relacionará con la bandera START.. Para ello, seleccione el botón y con "click" derecho, en la ventana que se presenta escoger la opción tocar "Touch". A continuación se debe relacionar la bandera, presionando banderas "Tags", se debe escoger START. Lo mismo debe realizarse para los otros dos botones. Uno de ellos se identifica con el texto: *2.5v*, el otro con: *0v.* A ambos se les asignará la bandera STEP.

La interface al usuario es el elemento 3. La figura 4.6 muestra este elemento.

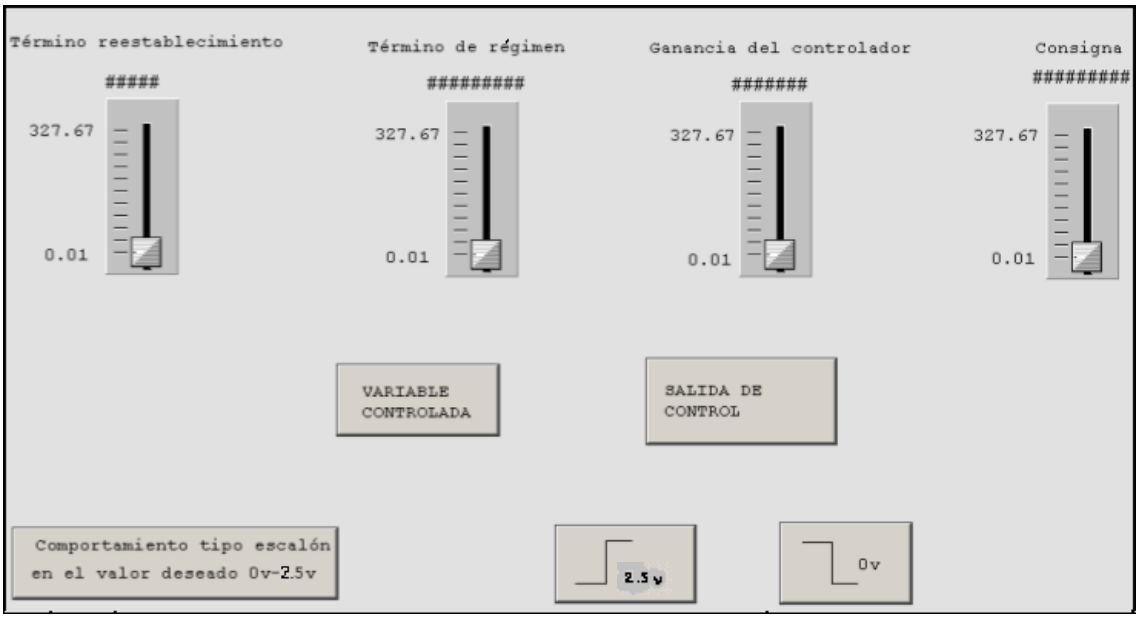


Figura 4.6 Elemento gráfico 3. Interface al usuario.

4.3 Especificaciones de diseño y modelado matemático.

A este punto toda la configuración y el montaje del equipo está concluido. A continuación se realizarán cálculos realcionados al algoritmo PID y al análisis del sitema.

Existen dos aspectos que se quieren analizar en el proyecto. El primero es la capacidad del método de sintonización para lograr una buena sintonización para el control sobre plantas con diferentes tiempos muertos. El segundo aspecto es la relación que existe entre el periodo de actualización del bloque de función PID y la estabilidad del sistema. Para poder analizar el primer aspecto, se realizarán pruebas experimentales del sistema de control, para diferentes tiempos muertos de la planta. Las pruebas se dividirán en tres casos:

Caso I: Se modificará la planta de tal forma que su función de transferencia sea de segundo orden sin tiempo muerto.

Caso II: Se modificará la planta de tal forma que su función de transferencia sea de segundo orden con un tiempo muerto de 10ms.

Caso III: Se modificará la planta de tal forma que su función de transferencia sea de segundo orden con un tiempo muerto de 1 segundo.

Para analizar el segúndo aspecto se realizarán varios experimentos en cada caso, cada uno para un diferente tiempo de actualización del bloque de función PID.

En el apartado 4.3.1 se sintonizan los parámetros del PID para cada caso. El método de sintonización que se utilizará es el de **síntesis de controladores**.

En el apartado 4.3.2 se realiza el modelo matemático del sistema. La información más importante que se obtendrá del modelo es su respuesta ante un cambio escalón unitario en el valor deseado.

4.3.1 Sintonización del algoritmo PID.

Para sintonizar el PID, se utilizará un modelo del sistema en tiempo continuo. Esto debido a que el método **síntesis de controladores** es aplicable a sistemas continuos. Ya que el PLC es un controlador discreto, la respuesta del sistema de control real puede diferir de la respuesta que se obtendrá mediante este modelo continuo. Los parametros derivados de la respuesta al escalón del modelo continuo se identificarán como **parámetros continuos**.

El diagrama en bloques del modelo en tiempo continuo del sistema de control se muestra en la figura.4.7.

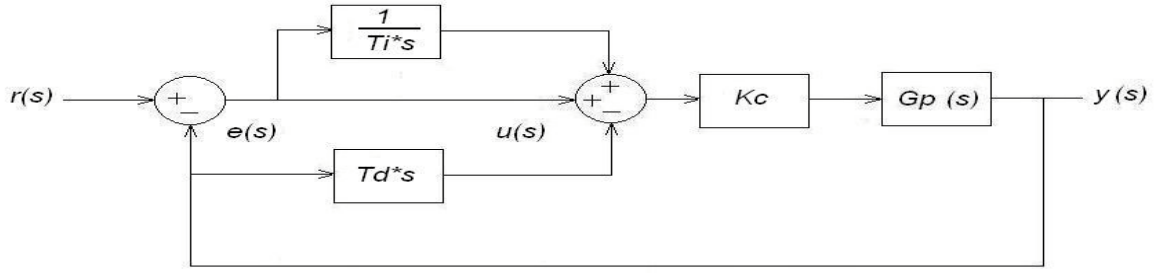


Figura 4.7 Diagrama en bloques del modelo en tiempo continuo del sistema de control.

Notese que el modo derivativo es aplicado a la variable realimentada, tal y como indica el algoritmo del bloque de función PID del PLC.

La función de transferencia del sistema en lazo cerrado es:

$$\frac{Y(s)}{R(s)} := \frac{K_c \cdot Gp(s) \cdot \left(1 + \frac{1}{T_i \cdot s} \right)}{1 + K_c \cdot Gp(s) \cdot \left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s \right)} \tag{4.2}$$

Con

$$Gp(s) := \frac{e^{-t_m \cdot s}}{(\tau_1 \cdot s + 1) \cdot (\tau_2 \cdot s + 1)} \tag{4.3}$$

el modelo de la planta.

Se escoge $\tau_1 := \tau_2 := \tau$ de tal forma que el polo de la planta sea doble. Los parámetros de la planta serían entonces '$\mathbf{t_m}$' y '$\boldsymbol{\tau}$'. Para todos los casos, '$\boldsymbol{\tau}$' = 1 segundo.

4.3.1.1 Especificaciones de diseño del sistema de control.

El objetivo del método de síntesis de controladores es lograr que el sistema de lazo cerrado se comporte como un modelo de primer orden más tiempo muerto. El tiempo muerto del modelo de lazo cerrado es siempre igual al de la planta, y se denotará como '$\mathbf{t_m}$'.
La tabla 4.4. muestra las especificaciones de diseño del sistema de control para cada caso de planta proceso. La función de transferencia del sistema es:

$$Mr(s) := \frac{e^{-t_m \cdot s}}{\tau_c \cdot s + 1} \qquad (4.4)$$

Con $\mathbf{t_m}$ = tiempo muerto $\boldsymbol{\tau_c}$ = constante de tiempo del modelo Mr(s).

Caso	tiempo muerto (segundos)	tiempo de asentamiento al 5% (segundos)	error permanente (%)	Constante de tiempo (segundos)
Caso I	0	1.5	0	0.5
Caso II	0.01	1.51	0	0.5
Caso III	1	5.05	0	0.94

Tabla 4.4 Especificaciones de diseño del sistema de control.

4.3.1.2 **Caso I.** Planta de segundo orden sin tiempo muerto.

En primer término se realiza la sintonización del controlador para una planta de segundo orden sin tiempo muerto. La tabla 4.5 muestra el valor de los parámetros de tiempo integral, derivativo y ganacia para un algoritmo PID ideal, y los respectivos valores de los parámetros del bloque de función PID del PLC. Es importante recordar que el método de síntesis de controladores es aplicable para controladores PID serie. Sin embargo, como se dijo, el algoritmo implementado en el PLC es de tipo ideal, por lo que una conversión de parámetros debe ser aplicada.

T_i (segundos)	T_d (segundos)	K_c (sin unidades)	TI (decimal)	TD (decimal)	KC (decimal)
2	0.5	4	3	1	400

Tabla 4.5 Valor de los parámetros del algoritmo PID para el caso I.

La figura 4.8 muestra la respuesta del modelo continuo ante un cambio escalón unitario en el valor deseado bajo las condiciones del caso I. La gráfica fue generada mediante el programa MATLAB 6.5, licencia de la Escuela de Ingeniería Eléctrica de la Universidad de Costa Rica.

Según la figura 4.8, la respuesta del modelo presenta un *sobre paso* de 11%. Esto contradice las especificaciones del método de sintonización, el cual asegura que la respuesta del sistema será de primer orden. La razón de este *sobre paso* radica en que al usar este método se asume que el modo derivativo del algoritmo PID es aplicado sobre el error. Sin embargo esto no se cumple en el caso aquí estudiado, pues como se dijo, el algoritmo del bloque de función PID aplica la acción derivativa sobre la señal realimentada, no sobre el error. El efecto de esta diferencia se analizará en el capítulo 5.

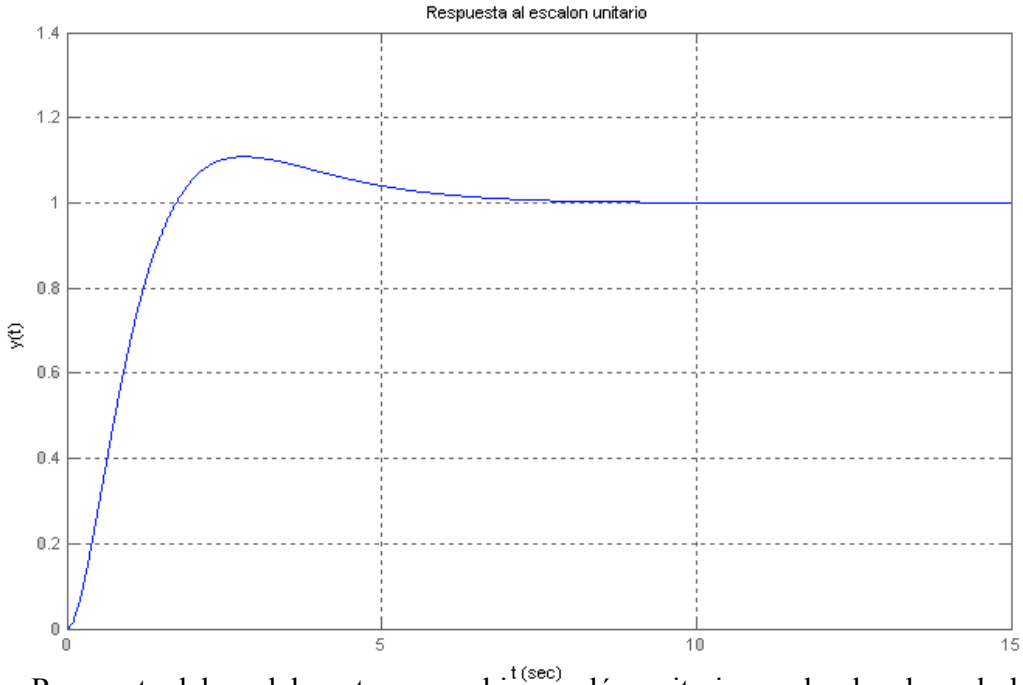


Figura 4.8 Respuesta del modelo ante un cambio escalón unitario en el valor deseado bajo las condiciones del caso I.

4.3.1.3 **Caso II.** Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$ = 10ms'.

Seguidamente se realiza la sintonización del controlador para una planta de segundo orden con tiempo muerto '$\mathbf{t_m}$ = 10ms'. La tabla 4.6 muestra el valor de los parámetros de tiempo integral, derivativo y ganacia para el algoritmo PID y los respectivos valores de los parámetros del bloque de función PID del PLC.

T_i (segundos)	T_d (segundos)	K_c (sin unidades)	TI (decimal)	TD (decimal)	KC (decimal)
2	0.5	3.92	3	1	392

Tabla 4.6 Valor de los parámetros del algoritmo PID para el caso II.

La figura 4.9 muestra la respuesta del modelo ante un cambio escalón unitario en el valor deseado en el caso II, obtenida con el programa MATLAB.

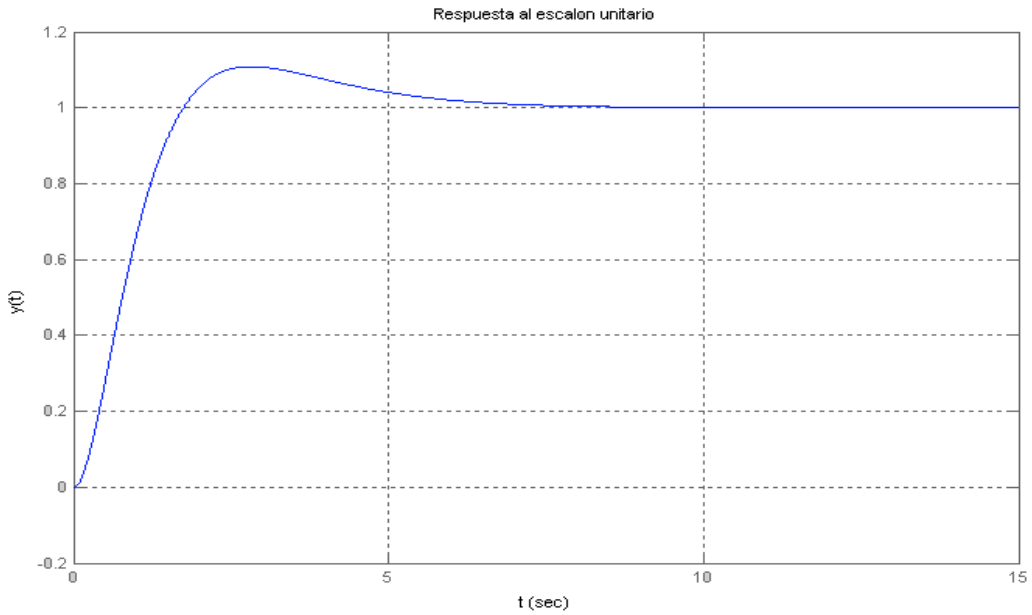


Figura 4.9 Respuesta del modelo ante un cambio escalón unitario en el valor deseado bajo las condiciones del caso II.

4.3.1.4 **Caso III.** Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$ = 1s'.

En el último caso se realiza la sintonización del controlador para una planta de segundo orden con tiempo muerto '$\mathbf{t_m}$ = 1s'. La tabla 4.7 muestra el valor de los parámetros de tiempo integral, derivativo y ganacia para un controlador PID ideal, y los respectivos valores de los parámetros del bloque de función PID del PLC, KC, TI y TD.

T_i (segundos)	T_d (segundos)	K_c (sin unidades)	TI (decimal)	TD (decimal)	KC (decimal)
2	0.5	1.33	3	1	133

Tabla 4.7 Valor de los parámetros del algoritmo PID para el caso III.

La figura 4.10 muestra la respuesta del modelo ante un cambio escalón unitario en el valor deseado para el caso III. De nuevo la gráfica fue obtenida con el programa MATLAB.

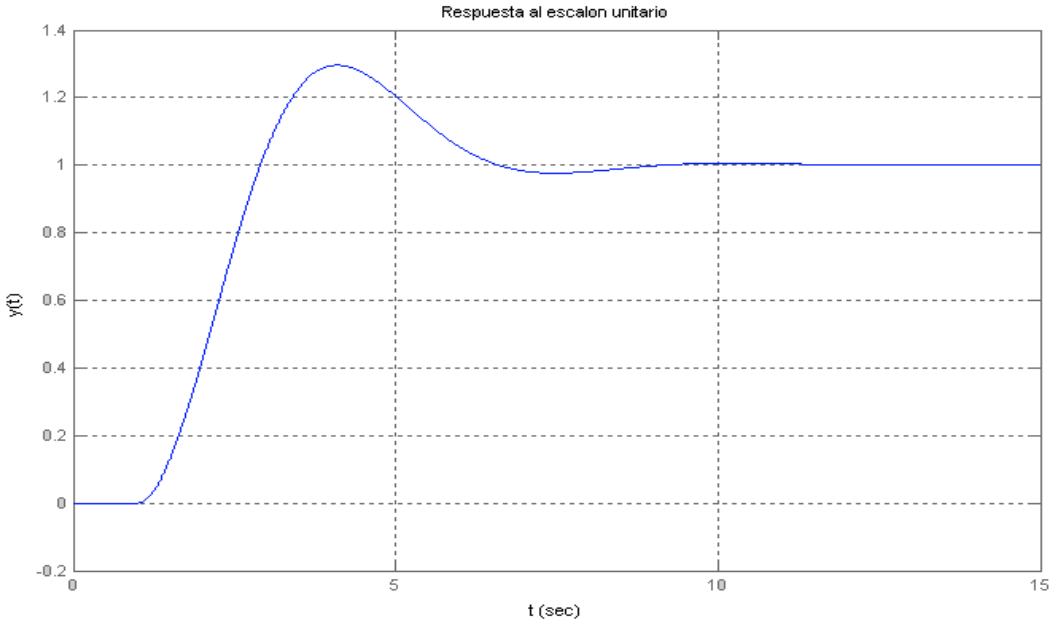


Figura 4.10 Respuesta del modelo ante un cambio escalón unitario en el valor deseado bajo las condiciones del caso III.

4.3.2 Modelado matemático del sistema de control.

El sistema implementado es un sistema de control en tiempo discreto, pues el PLC manipula señales discretas. Para modelar correctamente el sistema, es necesario utilizar un modelo discreto.

A continuación se calculará la función de transferencia discreta del modelo para los casos I, II y III. Además en cada caso se obtendrá esta función de transferencia para diferentes T. Una vez calculada la función de transferencia, se graficará la respuesta del modelo ante un cambio tipo escalón unitario en el valor deseado en cada caso. Estos gráficos se utilizarán para realizar el análisis de los resultados obtenidos de las pruebas experimentales realizadas en el laboratorio al sistema de control.

La figura 4.11 muestra el diagrama en bloques del modelo. La salida analógica del PLC es modelada por un retenedor de orden cero (ZOH), mientras que la entrada analógica se modela por un muestreador. El *parámetro actualización del bloque PID "Loop update"* será modelado por el valor de periodo de este muestreador, que se denotará como T. La planta, como siempre, es representada por el modelo Gp(s).

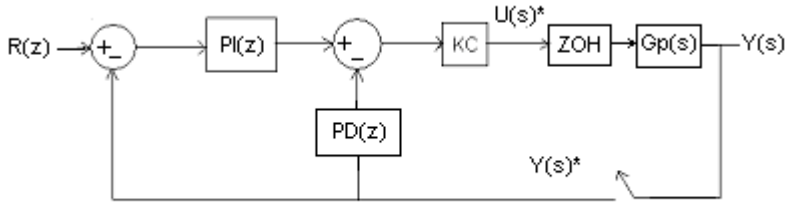


Figura 4.11 Diagrama en bloques del modelo discreto del sistema de control.

Las señales muestreadas se identifican con un asterisco. Es importante notar que tanto la señal del valor deseado como los bloques integral y derivativo de la función PID del PLC son modelados por sus transformadas z, debido a que son señales y algoritmos discretos.

Aquí G(z) es el equivalente retenedor planta. La función de transferencia de la planta es, como se definió,

$$Gp(s) := \frac{e^{-t_m \cdot s}}{(\tau \cdot s + 1)^2} \tag{4.5}$$

Debido a que el retardo de transporte de la planta '$\mathbf{t_m}$' no necesariamente será multiplo del periodo de muestreo se hace necesario utilizar la transformada z modificada para hallar la función de transferencia del equivalente retenedor planta G(z).

Entonces G(z) es:

$$G(z) := \left(1 - z^{-1}\right) \cdot z^{-N} \cdot \left[\left(-\frac{e^{-m \cdot T} \cdot z^{-1}}{1 - e^{-T} \cdot z^{-1}} \right) + \left[\frac{-T \cdot e^{-m \cdot T} \cdot z^{-1} \cdot \left[m + \left[(1 - m) \cdot e^{-T} \cdot z^{-1} \right] \right]}{\left(1 - e^{-T} \cdot z^{-1}\right)^2} \right] + \left(\frac{z^{-1}}{1 - z^{-1}} \right) \right]$$

Con

$$m := 1 - \frac{\rho}{T} \tag{4.6}$$

$$t_m := N \cdot T + \rho$$

.

Aquí, N pertenece a los números enteros y ρ pertenece a los números reales positivos.

La función de transferencia discreta de los bloques PI(z) y PD(z) es:

$$PI(z) := K_c - \frac{K_c \cdot T}{2 \cdot T_i} + \frac{K_c \cdot T}{T_i \cdot \left(1 - z^{-1}\right)} \tag{4.7}$$

$$PD(z) := \frac{K_c \cdot T_d \cdot \left(1 - z^{-1}\right)}{T} \tag{4.8}$$

Con K_c, T_i y T_d los parámetros del algoritmo PID ideal.

Finalmente, la función de transferencia discreta de lazo cerrado del modelo Mr(z) es:

$$Mr(z) := \frac{G(z)\cdot PI(z)}{1 + G(z)\cdot(PI(z) + PD(z))} \tag{4.9}$$

4.3.2.1 Caso I. Planta de segundo orden sin tiempo muerto.

Para este primer caso la función de transferencia de la planta es:

$$Gp(s) := \frac{1}{(s + 1)^2} \tag{4.10}$$

Donde '**tm**' = 0s. Para este caso '**N' =** 0 y **'ρ'** = 0.

Como se dijo, para cada caso se analizará el modelo de control en tiempo discreto para diferentes valores del periodo de muestreo **T**.

4.3.2.1.1 Periodo de muestreo **T** = 0.01segundos.

Con los valores de los parámetros del algoritmo PID ideal de la tabla 4.5, el periodo de muestreo **T** = 0.01 segundos y el retardo de la planta '$\mathbf{t_m}$' = 0 segundos, se obtiene la función de transferencia Mr(z).

$$Mr(z) = \frac{\left(1.9686\,10^{-4}\cdot z + 1.992\,10^{-4}\cdot z^4 - 1.9653\,10^{-4}\cdot z^2 - 1.995\,10^{-4}\cdot z^3\right)}{\left(-9.868\,10^{-3} + z^5 - 3.9699\,z^4 + 5.920\,z^3 + 1.00006\,z - 3.94050\,z^2\right)} \tag{4.11}$$

La respuesta del modelo a un cambio escalón unitario en el valor consigna se muestra en la figura 4.12. El gráfico fue generado con MATLAB.

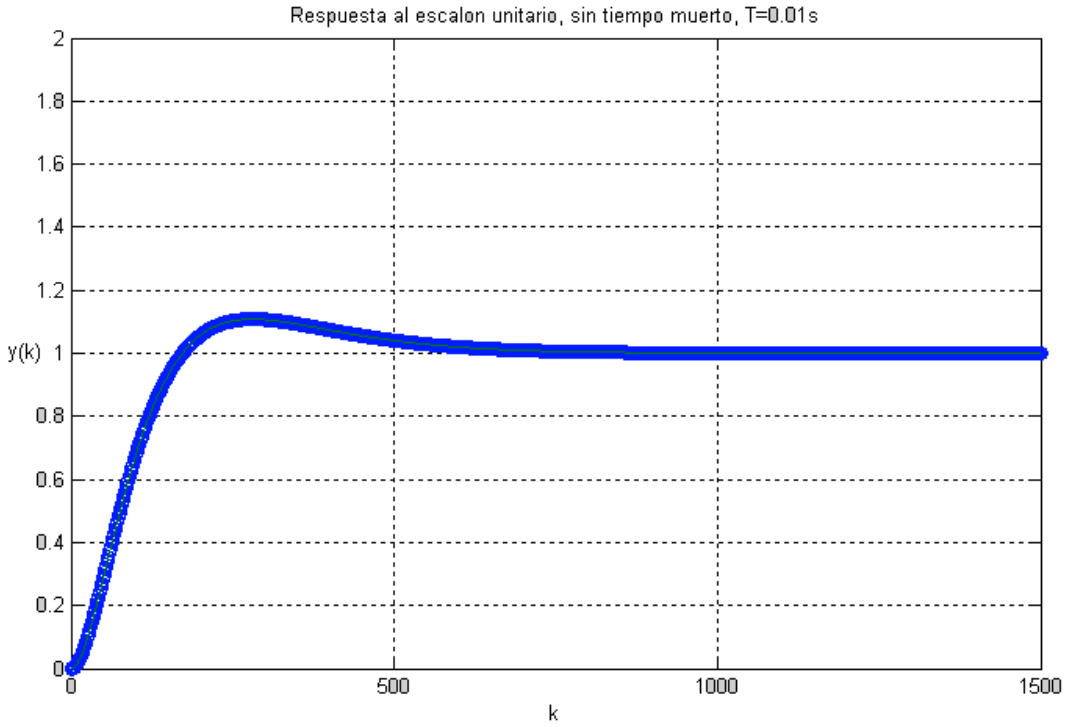


Figura 4.12 Respuesta al escalón unitario del modelo matemático. Planta de segundo orden sin tiempo muerto. Periodo de muestreo T = 0.01 segundos.

4.3.2.1.2 Periodo de muestreo **T** = 0.1segundos.

Con los valores de los parámetros del algoritmo PID ideal de la tabla 4.5, el periodo de muestreo **T** = 0.1 segundos y el retardo de la planta '$\mathbf{t_m}$' = 0 segundos, se obtiene la función de transferencia Mr(z).

$$Mr(z) = \frac{\left(1.707\cdot10^{-2}\cdot z + 1.9183\cdot10^{-2}\cdot z^4 - 1.677\cdot10^{-2}\cdot z^2 - 1.9485\cdot10^{-2}\cdot z^3\right)}{\left(-8.754\cdot10^{-2} + z^5 - 3.6969 z^4 + 5.225 z^3 + 1.0048 z - 3.4458 z^2\right)} \qquad (4.12)$$

La respuesta del modelo a un cambio escalón unitario en el valor consigna se muestra en la figura 4.13. El gráfico fue generado con MATLAB.

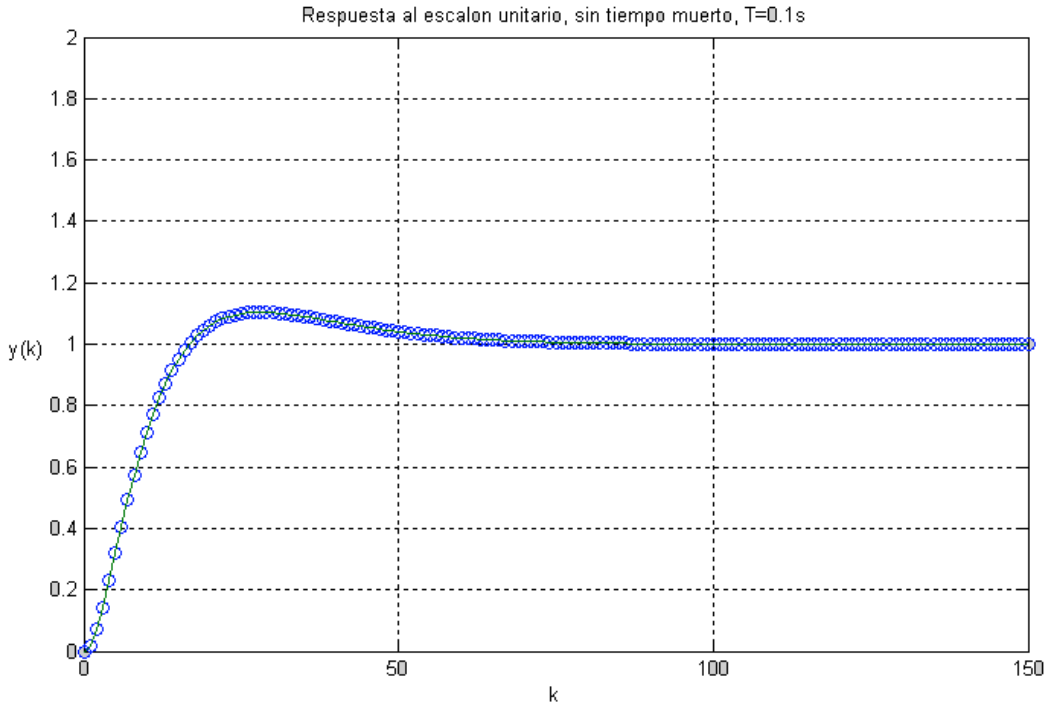


Figura 4.13 Respuesta al escalón unitario del modelo. Planta de segundo orden sin tiempo muerto. Periodo de muestreo T = 0.1 segundos.

4.3.2.1.3 Periodo de muestreo **T** = 0.5 segundos.

Con los valores de los parámetros del algoritmo PID ideal de la tabla 4.5, el periodo de muestreo **T** = 0.5 segundos y el retardo de la planta '$\mathbf{t_m}$' = 0 segundos, se obtiene la función de transferencia Mr(z).

$$Mr(z) = \frac{\left(.226149z + .40592z^4 - .201199z^2 - .43087z^3\right)}{\left(-.258456 - 2.44633z^4 + 2.5391z^3 + z^5 + 1.0086z - 1.8429z^2\right)} \qquad (4.13)$$

La respuesta a un cambio escalón unitario en el valor consigna se muestra en la figura 4.14. El gráfico fue generado con MATLAB.

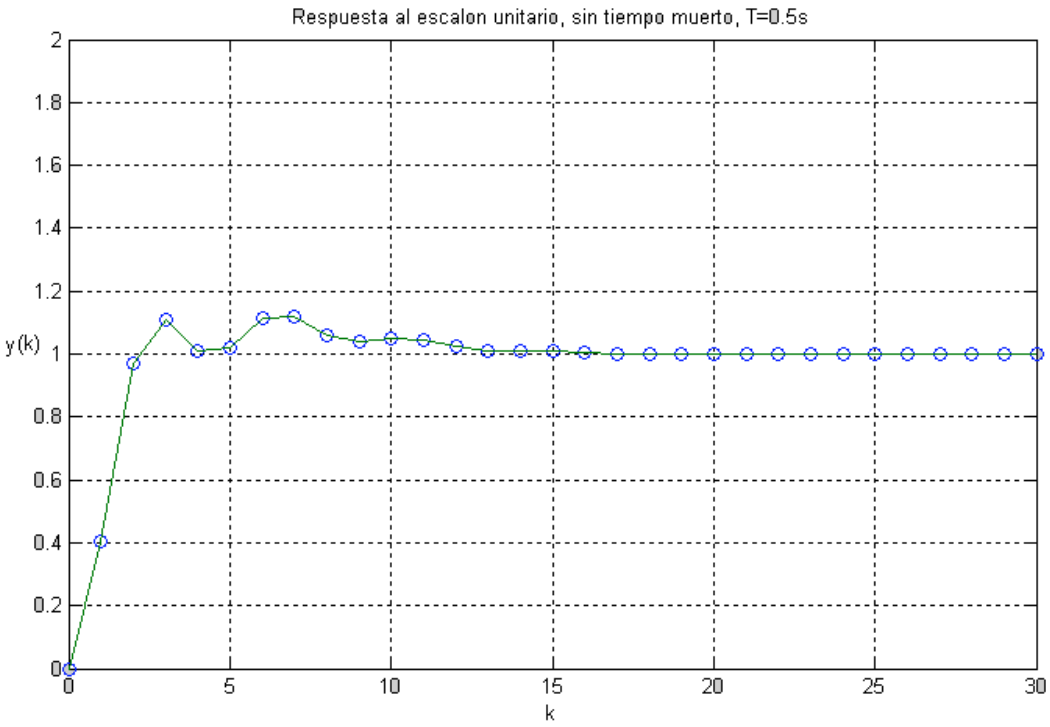


Figura 4.14 Respuesta del modelo al escalón unitario. Planta de segundo orden sin tiempo muerto. Periodo de muestreo T = 0.5 segundos.

4.3.2.1.4 Periodo de muestreo **T** = 0.75 segundos.

Con los valores de los parámetros para el algoritmo PID ideal dados en la tabla 4.5, el periodo de muestreo **T** = 0.75 segundos y el retardo de la planta '$\mathbf{t_m}$' = 0 segundos, se obtiene la función de transferencia Mr(z).

$$Mr(z) = \frac{\left(.341375z + .8235z^4 - .2769z^2 - .8879z^3 + 2.0\cdot10^{-20}\right)}{\left(-.280103 + z^5 - 1.65899z^4 + 1.1179z^3 + .9425z - 1.12133z^2\right)} \qquad (4.14)$$

La respuesta del modelo a un cambio escalón unitario en el valor consigna se muestra en la figura 4.15.

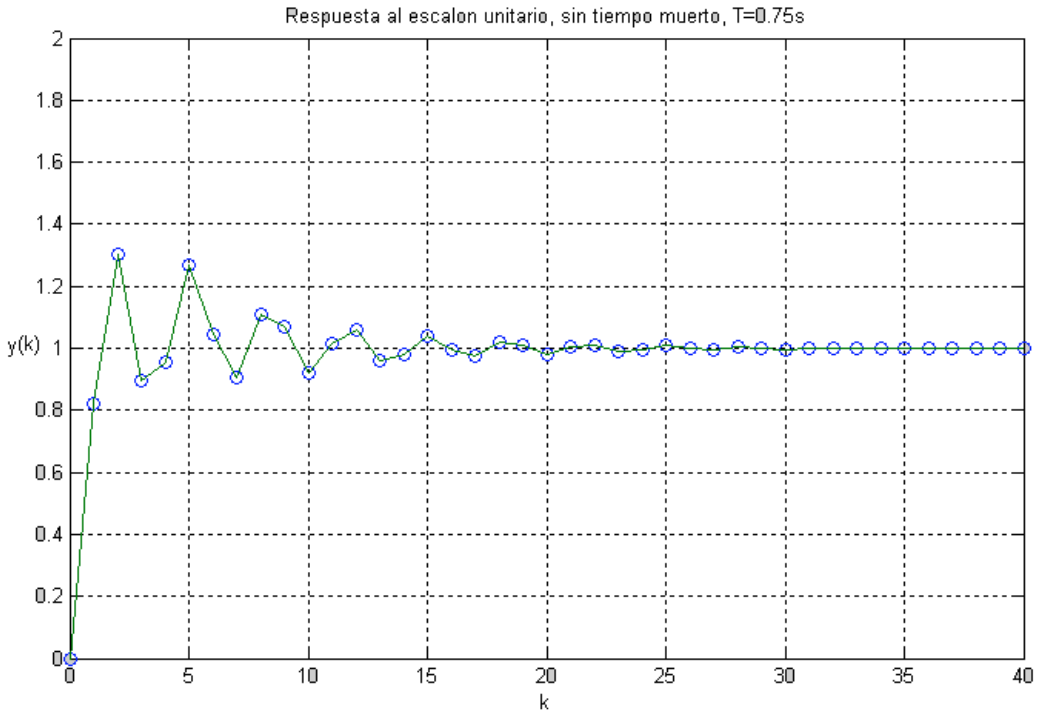


Figura 4.15 Respuesta al escalón unitario del modelo. Planta de segundo orden sin tiempo muerto. Periodo de muestreo T = 0.75 segundos.

4.3.2.1.5 Periodo de muestreo **T** = 1 segundo.

La ecuación 4.15 es la función de trasnferencia de lazo cerrado Mr(z) del sistema para un **T** = 1 segundo y un retardo de la planta '$\mathbf{t_m}$' = 0 segundos. Los parámetros del algoritmo PID ideal se presentan en la tabla 4.5.

$$Mr(z) = .35 \cdot z \cdot \left(377 \cdot 10^{14} \cdot z + 19 \cdot 10^{13}\right) \cdot \frac{(5. \cdot z - 3.)}{\left(5 \cdot 10^{18.} \cdot z^4 + 570 \cdot 10^{13.} \cdot z^3 - 156 \cdot 10^{13.} \cdot z^2 - 277 \cdot 10^{13.} \cdot z + 135 \cdot 10^{13.}\right)} \quad (4.15)$$

La respuesta del sistema a un cambio escalón unitario en el valor consigna se muestra en la figura 4.16.

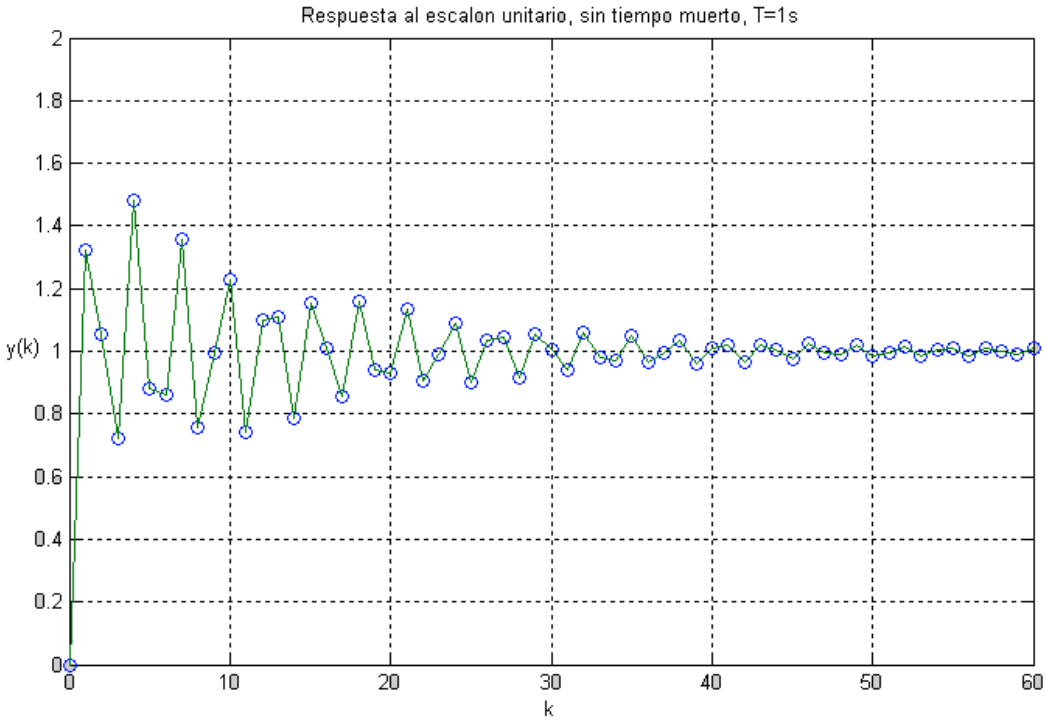


Figura 4.16 Respuesta al escalón unitario del modelo. Planta de segundo orden sin tiempo muerto. Periodo de muestreo T = 1 segundo.

4.3.2.2 Caso II. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 10ms.

Para este caso la función de transferencia de la planta es:

$$Gp(s) := \frac{e^{-0.01 \cdot s}}{(s+1)^2} \tag{4.16}$$

Donde '$\mathbf{t_m}$' = 0.01 segundos.

Ahora se realizará el estudio para diferentes periodos de muestreo T.

4.3.2.2.1 Periodo de muestreo **T** = 0.01segundos.

Para este caso '**N**' = 1 y '**ρ**' = 0. La tabla 4.6 muestra los valores de K_c, T_i y T_d para la sintonización del algoritmo PID en el caso II. Entonces con T = 0.01 segundos; Mr(z) es:

$$Mr(z) = \frac{\left(1.929\cdot10^{-4}\cdot z^2 + 1.95\cdot10^{-4}\cdot z^5 - 1.93\cdot10^{-4}\cdot z^3 - 1.956\cdot10^{-4}\cdot z^4\right)}{\left(-9.674\cdot10^{-3}\cdot z + 5.95\cdot z^5 - 3.96\cdot z^4 + 1.948\cdot10^{-2}\cdot z^2 + .98\cdot z^3 + z^7 - 3.98\cdot z^6 - 1.0\cdot10^{-17}\right)} \qquad (4.17)$$

La respuesta del modelo a un cambio escalón unitario en el valor consigna se muestra en la figura 4.17.

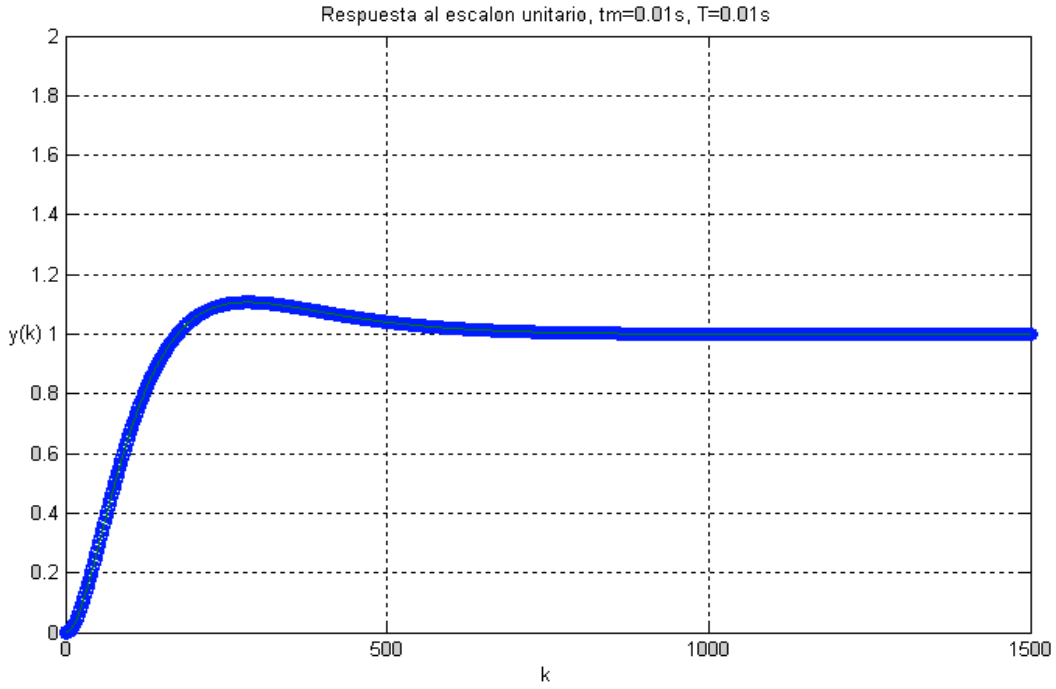


Figura 4.17 Respuesta al escalón unitario del modelo. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 0.01 segundos . Periodo de muestreo T = 0.01 segundos.

4.3.2.2.2 Periodo de muestreo **T** = 0.1segundos.

Para este caso '**N'** = 0 y **'ρ'** = 0.01. Con los valores de K_c, T_i y T_d dados en la tabla 4.6 y T = 0.01 segundos se calcula Mr(z):

$$Mr(z) = \frac{\left(1.96\cdot10^{-2}\cdot z^2 + 1.53\cdot10^{-2}\cdot z^5 - 2.60\cdot10^{-2}\cdot z^3 - 9.021\cdot10^{-3}\cdot z^4 + 1.576\cdot10^{-4}\cdot z\right)}{\left(-9.937\cdot10^{-2}\cdot z - 3.7195\cdot z^5 + 5.307\cdot z^4 + 1.067\cdot z^2 - 3.554\cdot z^3 + z^6 - 8.081\cdot10^{-4}\right)} \qquad (4.18)$$

La respuesta del modelo a un cambio escalón unitario en el valor se muestra en la figura 4.18.

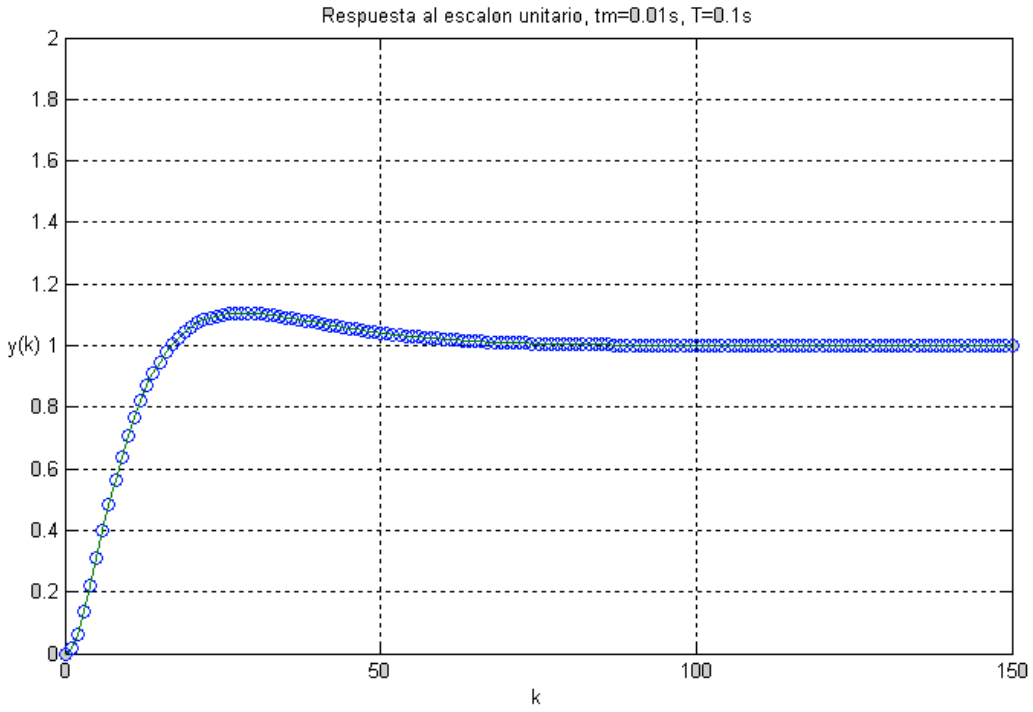


Figura 4.18 Respuesta al escalón unitario del modelo. Planta de segundo orden con tiempo muerto '**tm**' = 0.01 segundos . Periodo de muestreo T = 0.1 segundos.

4.3.2.2.3 Periodo de muestreo **T** = 0.5 segundos.

Aquí '**N'** = 0 y **'ρ'** = 0.01 tal que '$\mathbf{t_m}$' = 0.01 segundos. Dados K_c, T_i y T_d como en la tabla 4.6 y T = 0.5 segundos se calcula Mr(z):

$$Mr(z) = \frac{\left(.232 z^2 + .385 z^5 - .231 z^3 - .386 z^4 + 6.35 10^{-5} z\right)}{\left(-.265 z - 2.49 z^5 + 2.65 z^4 + 1.05 z^2 - 1.955 z^3 + z^6 - 7.262 10^{-5}\right)} \qquad (4.19)$$

La respuesta del modelo a un cambio escalón unitario en el valor consigna se muestra en la figura 4.19.

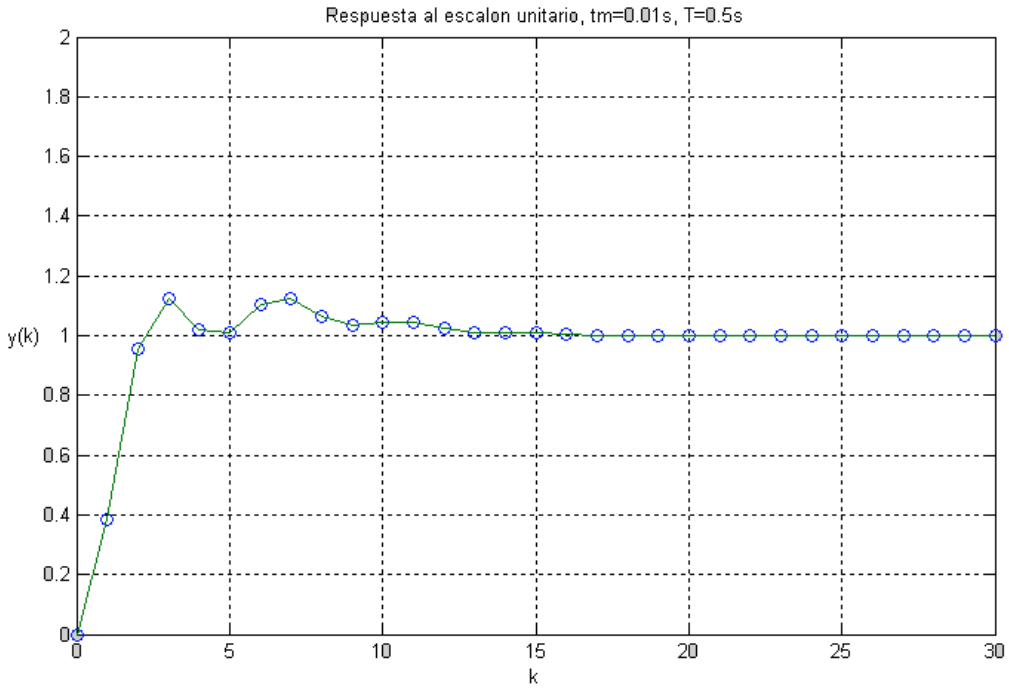


Figura 4.19 Respuesta del modelo al escalón unitario. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 0.01 segundos. Periodo de muestreo T = 0.5 segundos.

4.3.2.2.4 Periodo de muestreo **T** = 0.75 segundos.

Finalmente se realiza el estudio para un periodo de muestreo de 750 ms. Reemplazando los valores de $T_i = 2$, $T_d = 0.5$, $K_c = 3.9216$, **N** = 0, $\rho = 0.01$ y T = 0.75, se calculó la siguiente función de transferencia Mr(z):

$$Mr(z) = \frac{\left(.3458z^2 + .7908z^5 - .3103z^3 - .8264z^4 + 3.579\,10^{-5}\cdot z\right)}{\left(-.2837\cdot z - 1.71\cdot z^5 + 1.238z^4 + .9764z^2 - 1.2208z^3 + z^6 - 2.936\,10^{-5}\right)} \qquad (4.20)$$

La respuesta del modelo a un cambio escalón unitario en el valor consigna se muestra en la figura 4.20.

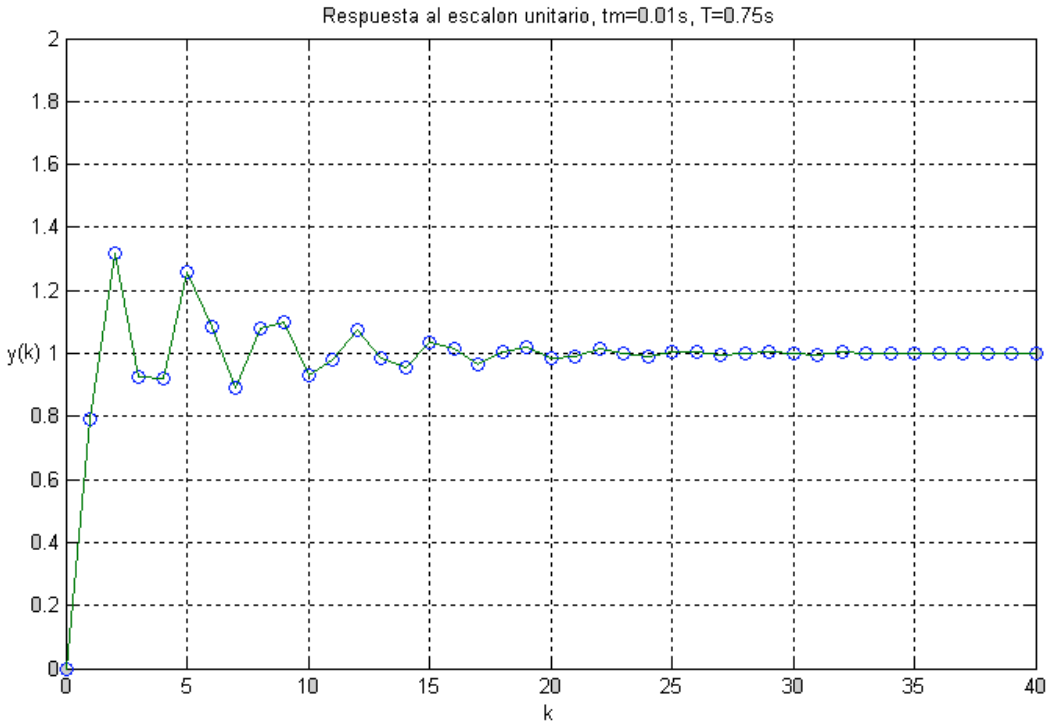


Figura 4.20 Respuesta al escalón unitario del modelo. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 0.01 segundos . Periodo de muestreo T = 0.75 segundos.

4.3.2.3 Caso III. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 1s.

Para este caso la función de transferencia de la planta es:

$$Gp(s) := \frac{e^{-s}}{(s+1)^2} \quad . \tag{4.21}$$

Donde '$\mathbf{t_m}$' = 1s.

Ahora se realizará el estudio de esta planta para diferentes periodos de muestreo T.

4.3.2.3.1 Periodo de muestreo **T** = 0.1 segundos.

La sintonización del controlador PID ideal para el caso de '$\mathbf{t_m}$' = 1 segundo se muestra en la tabla 4.7. Según la tabla, Kc = 1.33, Ti = 2 y Td = 0.5. Como se definió, $t_m = N*T+\rho$. Entonces para este caso N = 10, $\rho = 0$. La función de transferencia Mr(z) es:

$$Mr(z) = \frac{\left(5.676\,10^{-3}\cdot z + 6.378\,10^{-3}\cdot z^4 - 5.576\,10^{-3}\cdot z^2 - 6.4787\,10^{-3}\cdot z^3\right)}{\left(-2.9\cdot 10^{-2} + 3.75\cdot 10^{-2}\cdot z^4 - 7.1\cdot 10^{-2}\cdot z^3 + 6.2\cdot 10^{-2}\cdot z + 4.4\cdot 10^{-4}\cdot z^2 + z^{15} - 3.8\cdot z^{14} + 5.44\cdot z^{13} - 3.45\cdot z^{12} + .82\cdot z^{11}\right)} \quad (4.22)$$

Utilizando el programa MATLAB, se obtuvo la respuesta del sistema modelado por Mr(z) ante una entrada escalón unitario en R(z). La respuesta se muestra en la figura 4.21.

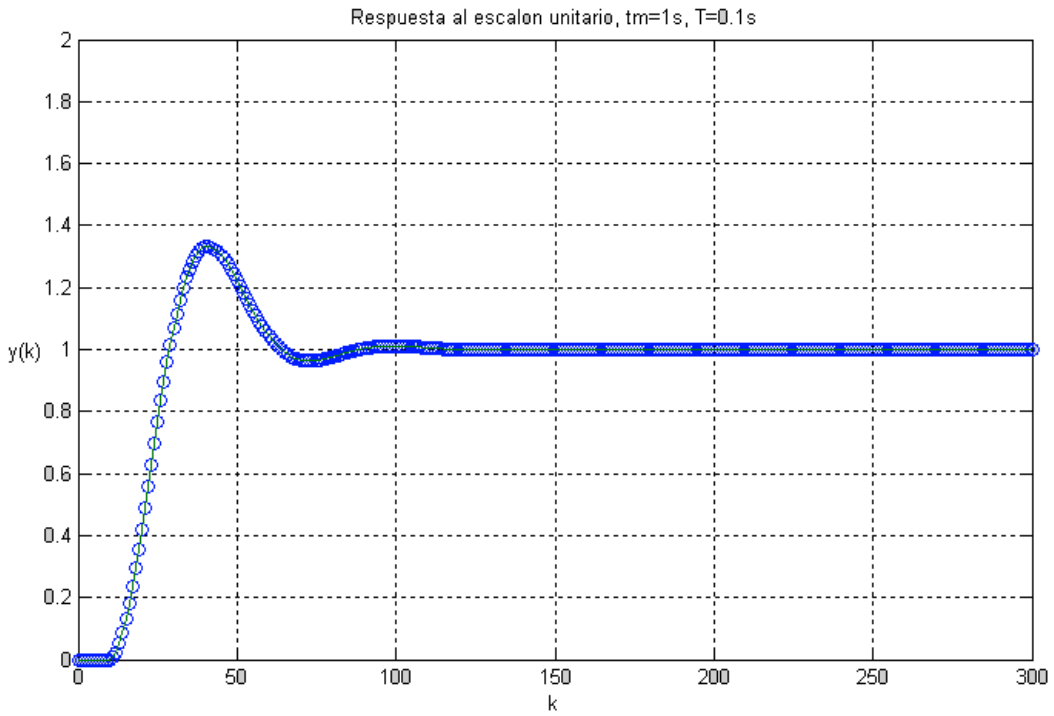


Figura 4.21 Respuesta al escalón unitario del modelo. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 1 segundo . Periodo de muestreo T = 0.1 segundos.

4.3.2.3.2 Periodo de muestreo **T** = 0.5 segundos.

Para este caso N = 2 y ρ = 0, tal que t_m = 1 segundos y T = 0.5 segundos. Entonces, para calcular Mr(z) se reemplazan estos valores y los parámetros del PID ideal mostrados en la tabla 4.7.

La función de transferencia Mr(z) es:

$$Mr(z) = \frac{\left(7.52\cdot10^{-2}\cdot z^2 + .135\cdot z^5 - 6.69\cdot10^{-2}\cdot z^3 - .143\cdot z^4\right)}{\left(-8.59\cdot10^{-2}\cdot z - 1.69\cdot z^5 - 4.94\cdot10^{-2}\cdot z^4 + .213\cdot z^2 + 3.52\cdot10^{-2}\cdot z^3 + z^8 + 1.0\cdot10^{-20} - 3.21\cdot z^7 + 3.79\cdot z^6\right)} \quad (4.23)$$

La respuesta de este modelo ante un cambio escalón unitario en la entrada R(z) del valor deseado se muestra en la figura 4.22.

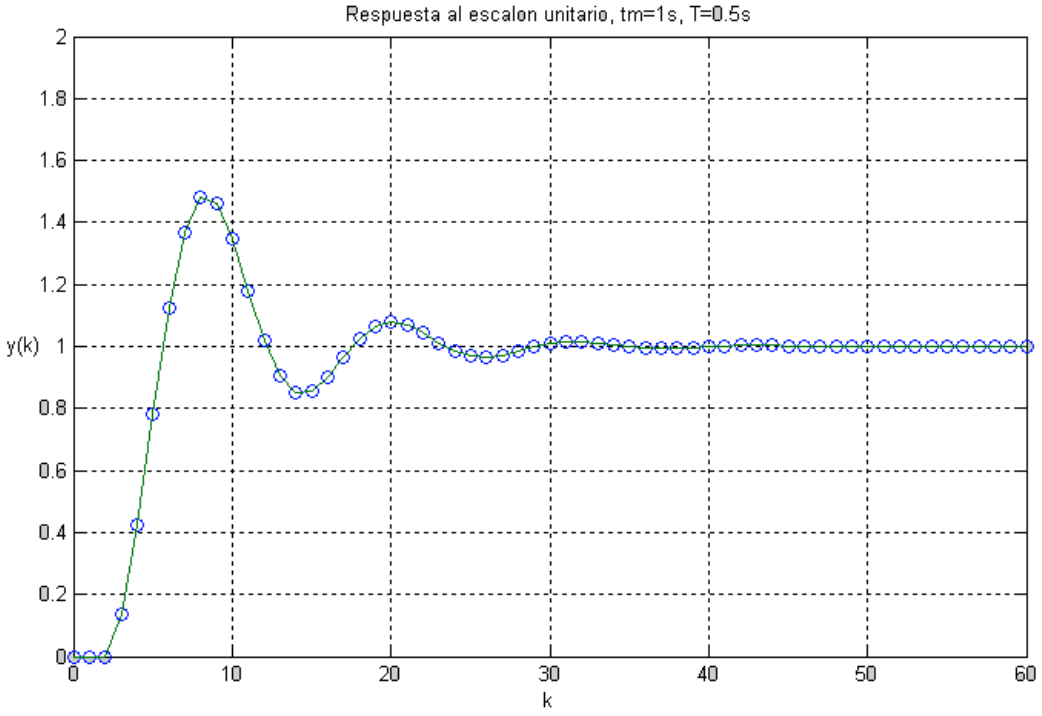


Figura 4.22 Respuesta al escalón unitario del modelo. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 1 segundo . Periodo de muestreo T = 0.5 segundos.

4.3.2.3.3 Periodo de muestreo **T** = 1.2 segundos.

Para este caso N = 0 y ρ = 1, tal que t_m = 1s y T = 1.2 s. Mr(z) se obtiene al reemplazar estos valores y los parámetros del PID ideal mostrados en la tabla 4.7 en la ecuación (4.9) de Mr(z). La función de transferencia Mr(z) es:

$$Mr(z) = \frac{\left(.113 z^2 + 3.03 10^{-2} z^5 - .838 z^3 + .6106 z^4 + 8.446 10^{-2} z\right)}{\left(2.46 10^{-2} z - 2.56 z^5 + 3.088 z^4 + .675 z^2 - 2.17 z^3 + z^6 - 5.027 10^{-2}\right)} \tag{4.24}$$

La respuesta de este modelo ante un cambio escalón unitario en la entrada R(z) del valor deseado se muestra en la figura 4.23.

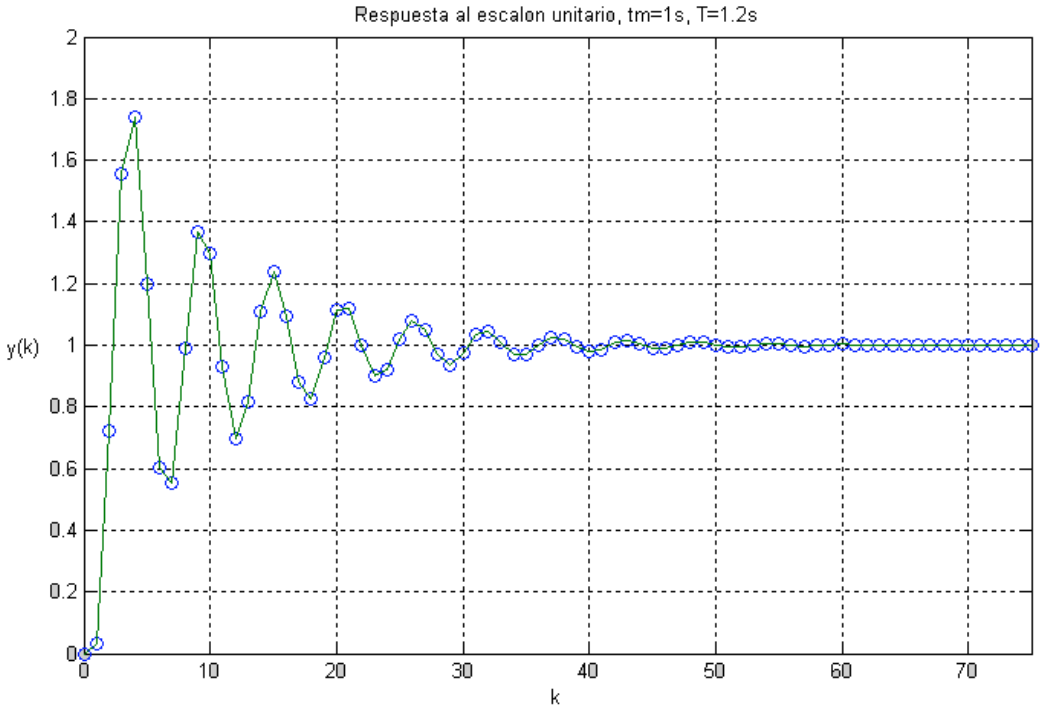


Figura 4.23 Respuesta al escalón unitario del modelo. Planta de segundo orden con tiempo muerto '$\mathbf{t_m}$' = 1 segundos. Periodo de muestreo T = 1.2 segundos.

Capítulo 5: Análisis de resultados

Una vez realizado el modelado matemático del sistema de control, se procedió a realizar la configuración del bloque de función PID del PLC. La tabla 5.1 muestra la configuración de los parámetros. Los parámetros no especificados tienen un valor por defecto.

Parámetro	Valor
Ganancia del controlador	Según el caso
Término de reestablecimiento	Según el caso
Término de taza de cambio	Según el caso
Actualización del bloque PID	Según el caso
Modo de control	E = SP-PV
Control PID	AUTO
Modo de Tiempo	TM
Límite de salida CV	NO
Banda Muerta	0
Valor consigna	0 - 8192
Bandera RG	1

Tabla 5.1 Configuración del bloque de función PID del PLC.

Como se muestra en la tabla, los valores Ganancia del controlador, Término de reestablecimiento, Término de taza de cambio y actualización del Bloque PID dependen del caso específico que se estudie. Es importante anotar que el parámetro del controlador que fue modelado en el análisis discreto como el periodo de muestreo T es el de actualización del Bloque PID.

Las pruebas experimentales consistirán en obtener el comportamiento de la variable controlada ante un cambio tipo escalón unitario en la señal consigna, configurando la planta para cambiar su retardo según cada uno de los casos descritos en el capítulo anterior. Complementariamente, para cada caso, se obtendrán diferentes curvas, cada una para un valor actualización del Bloque PID "loop update" diferente. Los parámetros de la respuesta del sistema experimentalmente se identificarán como **parámetros experimentales**.

Los resultados consisten en una gráfica del valor en decimal de la variable controlada contra el tiempo. El gráfico fue obtenido mediante información del elemento de archivo de entrada I:2.0, correspondiente al módulo de entradas analógicas del PLC. Esa información fue enviada al computador por medio de la interface al ususario y por ello se trata de una medición del valor de la salida de la planta. A continuación se presentan los resultados para cada caso estudiado.

5.1 Resultados.

5.1.1 Caso I. Control de planta de segundo orden sin tiempo muerto.

En este caso se ajustó la planta para que '$\mathbf{t_m}$' = 0. Las figuras de la 5.1 a la 5.5 muestran el comportamiento de la variable controlada en el tiempo, ante un cambio tipo escalón en el valor consigna. Cada gráfica se obtuvo para diferentes valores del parámetro Actualización del bloque PID. Para lograr este tipo de control se debe presionar el botón *comportamiento tipo escalón en el valor deseado 0v-2.5v* en el panel de control de usuario en la interface. Esto deshabilita el valor de consigna dado por el botón deslizable del mismo panel. Una vez hecho esto, se presiona el botón 2.5v para dar un cambio del valor consigna de 0 a 8192 en valor decimal, o de 0 V a 2.5 V, según la conversión vista.

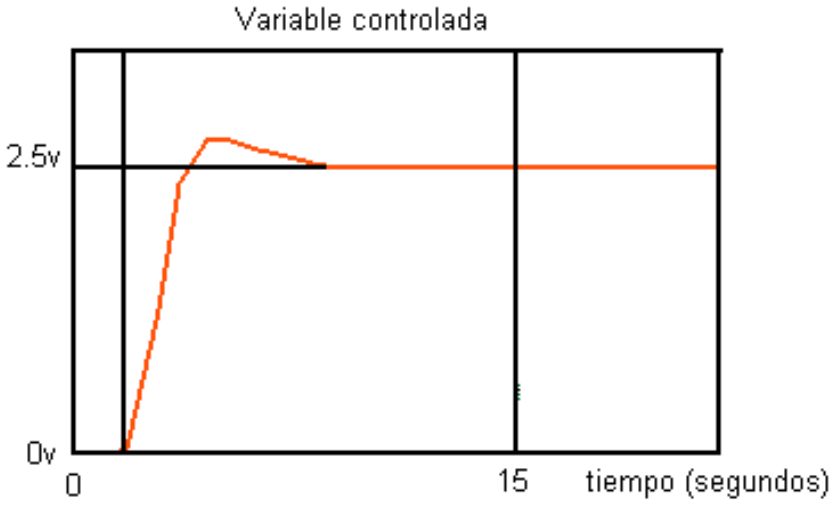


Figura 5.1 Variable Controlada contra el tiempo. Resultado del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 10ms.

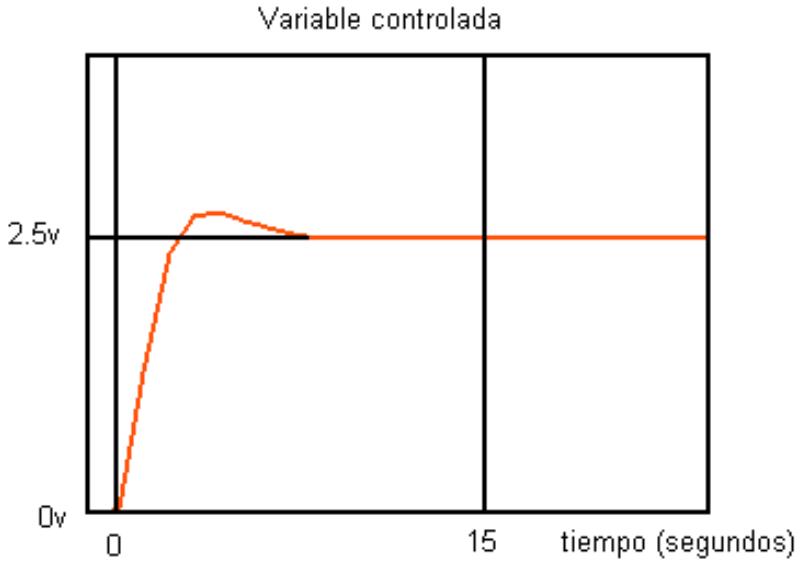


Figura 5.2 Variable Controlada contra el tiempo. Resultado del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 100ms.

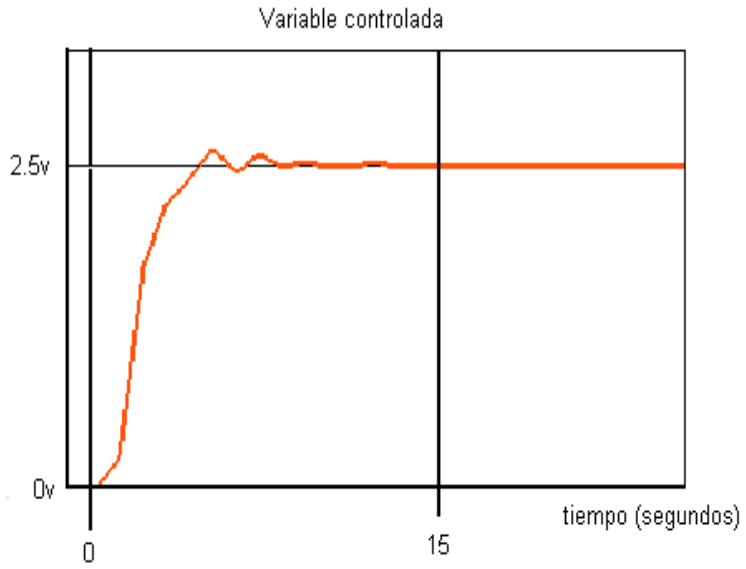


Figura 5.3 Variable Controlada contra el tiempo. Resultado del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 500ms.

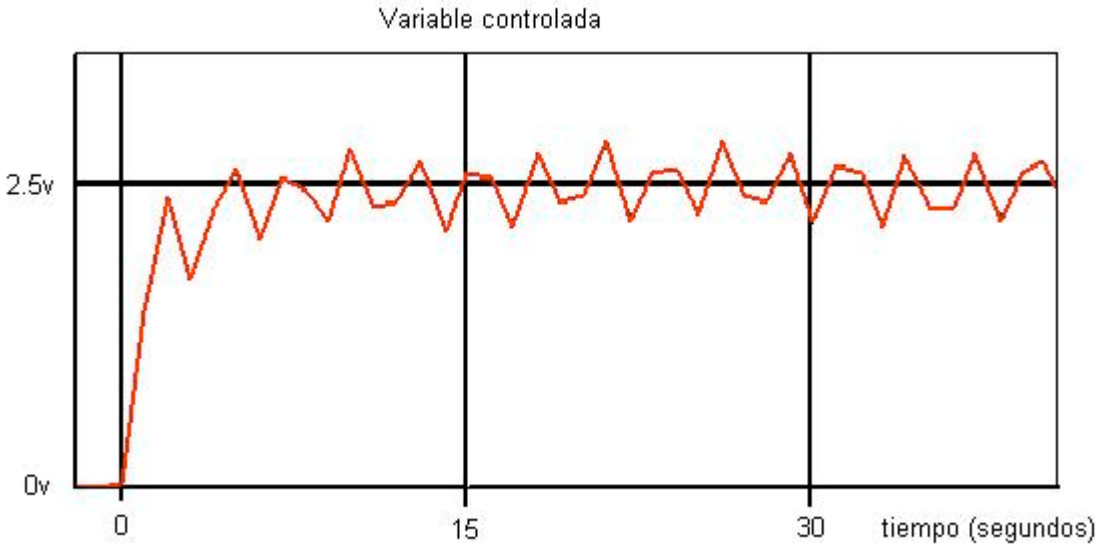


Figura 5.4 Variable Controlada contra el tiempo. Resultado del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 750ms.

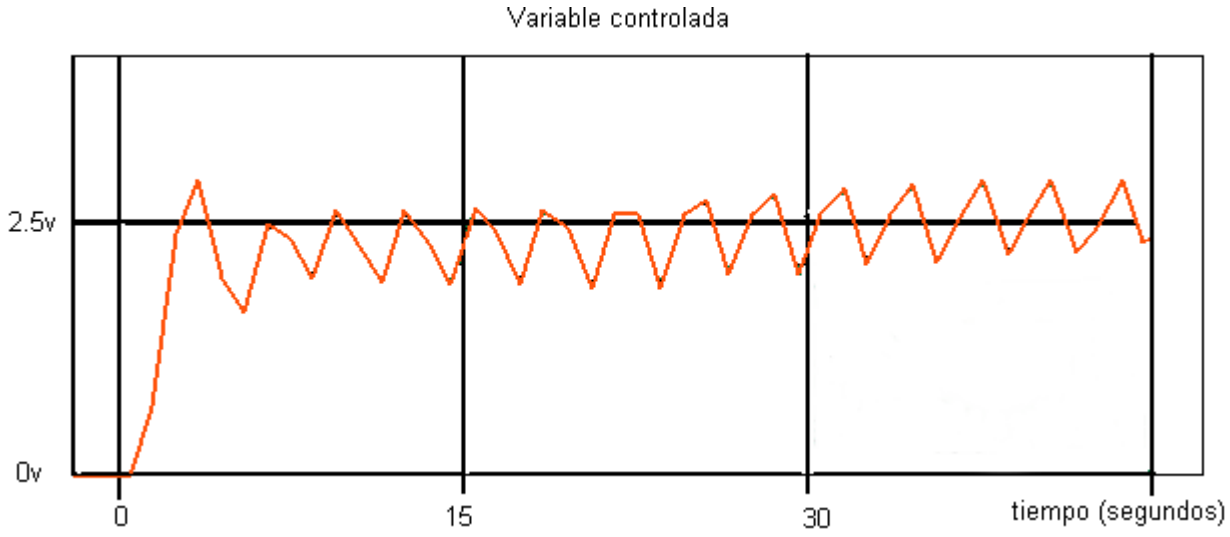


Figura 5.5 Variable Controlada contra el tiempo. Resultado del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 1 segundo.

5.1.2 **Caso II.** Control de planta de segundo orden con tiempo muerto 't_m' = 10ms.

Ahora se ajusta la planta para incorporar un retardo de transporte o tiempo muerto de 10 ms. Los resultados de la respuesta del sistema ante un cambio escalón se obtuvieron, de nuevo, para diferentes valores del parámetro Actualización del bloque PID. Las gráficas de las figuras 5.6 a 5.9 muestran estos resultados.

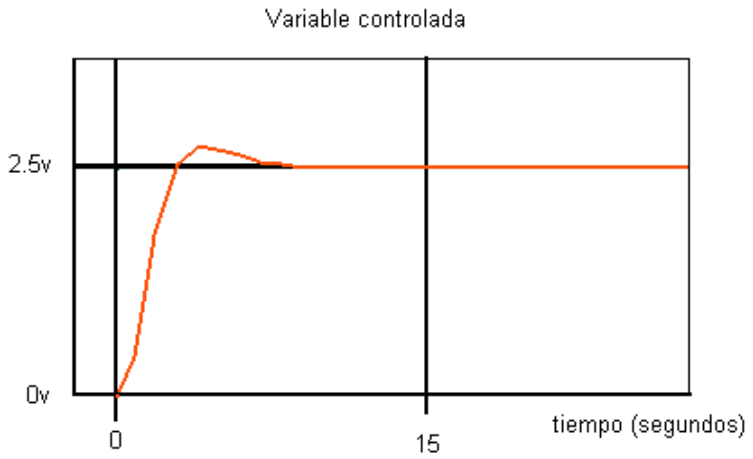


Figura 5.6 Gráfica de la Variable Controlada contra el tiempo. Comportamiento del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID=10ms.

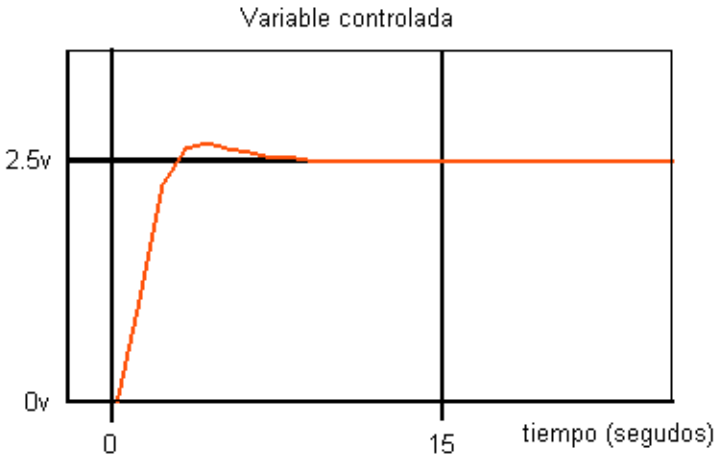


Figura 5.7 Gráfica de la Variable Controlada contra el tiempo. Comportamiento del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID=100ms.

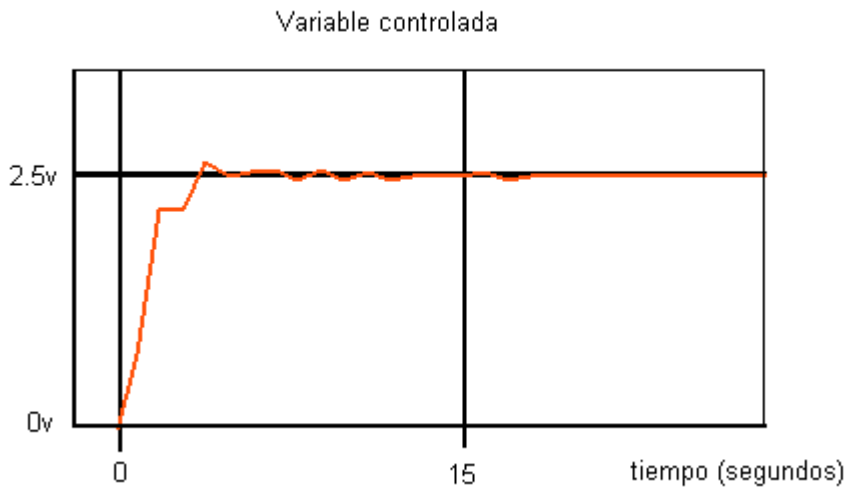


Figura 5.8 Gráfica de la Variable Controlada contra el tiempo. Comportamiento del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID=500ms.

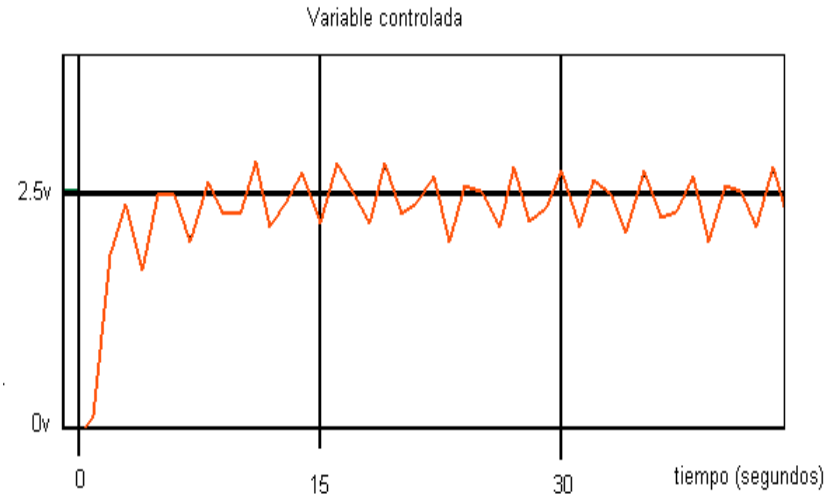


Figura 5.9 Gráfica de la Variable Controlada contra el tiempo. Comportamiento del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID=750ms.

5.1.3 **Caso III.** Control de planta de segundo orden con tiempo muerto 't_m' = 1 segundo.

Finalmente se ajusta la planta para tener un tiempo muerto de 1 segundo.. Las figuras de la 5.10. a 5.12. muestran los resultados obtenidos mediante la interface RSView32 para valores del parámetro Actualización del bloque PID de 100 ms, 500 ms y 1.2 segundos.

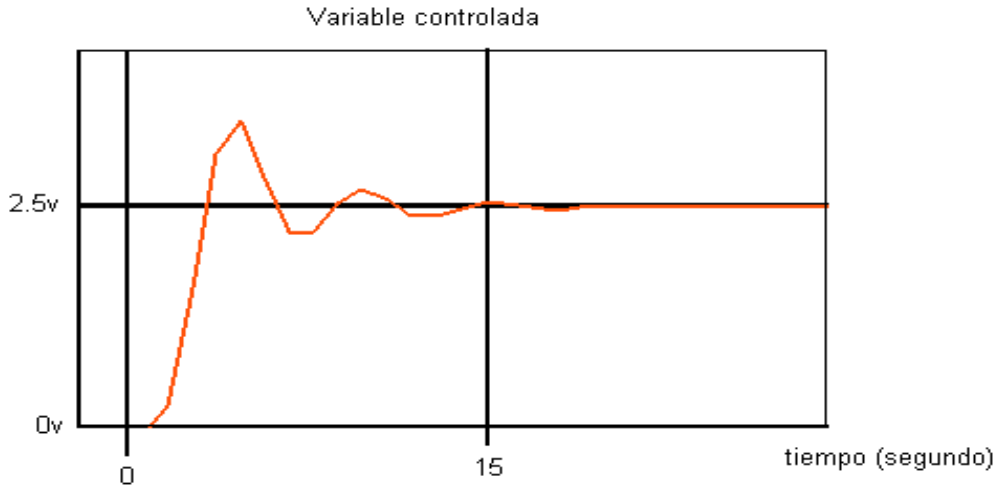


Figura 5.10 Gráfica de la Variable controlada contra el tiempo. Comportamiento del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 100ms.

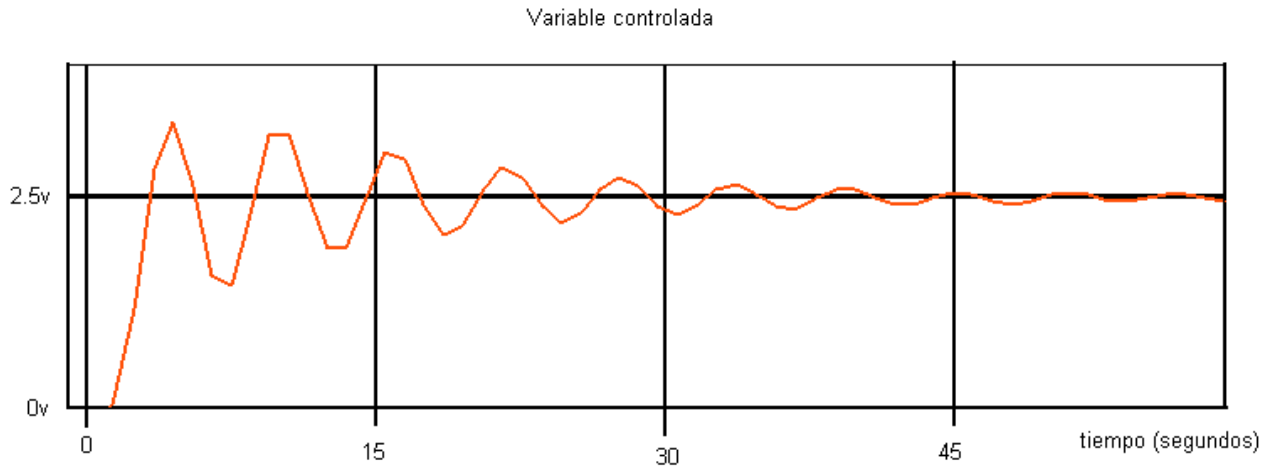


Figura 5.11 Gráfica de la Variable controlada contra el tiempo. Comportamiento del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 500ms.

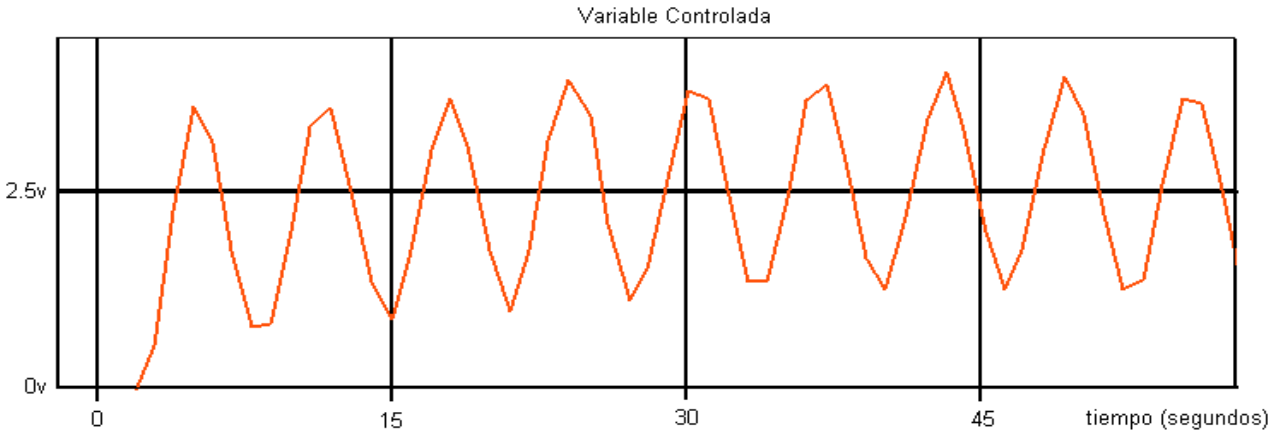


Figura 5.12 Gráfica de la Variable controlada contra el tiempo. Comportamiento del sistema ante un cambio de 0 V a 2.5 V en el valor consigna. Actualización del bloque PID = 1.2 s.

5.2 Análisis de resultados.

En la sección anterior se presentaron los resultados de las pruebas experimentales realizadas

en el laboratorio para cada un de los casos de interés. Seguidamente se compararán los resultados experimentales con los resultados del modelo en tiempo discreto y con las especificaciones de diseño. El análisis se presentará primero por separado en los casos I, II y III según el tiempo muerto de la planta. Posteriormente se analarizarán los resultados globalmente.

5.2.1 Caso I. Planta sin tiempo muerto 't_m' = 0s.

La tabla 5.2 muestra los parámetros de la respuesta*; constante de tiempo del sistema*, *tiempo muerto*, *sobrepaso* y *tiempo de asentamiento al 5%,* según las especificaciones de diseño y el modelo continuo.

Especificaciones de diseño	0.5	0	0	1.5
Modelo continuo	0.94	0	11	4.53

Tabla 5.2 Especificaciones de diseño y resultados de los parámetros continuos para el caso I.

Como se aprecia, existe una diferencia entre los parámetors continuos y las especificaciones de diseño. Específicamente el parámetro continuo *constante de tiempo del sistema* es casi el doble del valor meta de diseño. Además se presenta un *sobrepaso* en el modelo continuo del 11% del valor final. Es, entonces, evidente que las especificaciones de diseño no serán cumplidas por el sistema de control. Como se había apuntado anteriormente, esto se debe a que el algoritmo PID que implementa el PLC; aplica el modo derivativo a la variable realimentada. Por otra parte, el método de sintonización está diseñado tal que el modo derivativo sea aplicado a la señal de error.

Debido a esta diferencia, no se podrán alcanzar las especificaciones de diseño que fueron calculadas mediante el método de síntesis de controladores. Con el fin de poder continuar con el análisis, se asumirá que las nuevas especificaciones de diseño para el caso I son las dadas por el modelo continuo según la tabla 5.2.

La tablas 5.3 muestran los parámetros obtenidos de las respuestas del modelo discreto y de las respuestas experimentales según el periodo de muestreo. Los datos se muestran de la siguiente forma: (parámetro discreto ; parámetro experimental).

Periodo de muestreo (segundos)	**Constante de tiempo del sistema (segundos)**	**Tiempo muerto (segundos)**	**Sobre paso (%)**	**Tiempo de asentamiento al 5% (segundos)**
0.01	(0.917 ; 1.45)	(0 ; 0)	(11 ; 9.57)	(4.63 ; 5.36)
0.1	(0.83 ; 1.45)	(0 ; 0)	(10.5 ; 8.7)	(4.83 ; 6)
0.5	(0.67 ; 2.1)	(0 ; 0)	(10.53 ; 6.1)	(5 ; 5.64)
0.75	(0.7 ; 1.36)	(0 ; 0)	(31.6 ; 11.3)	(9.2 ; indet)
1	(0.6 ; 2.1)	(0 ; 0)	(47.4 ; 17.4)	(32.4 ; indet)

Tabla 5.3 Parámetros discretos y parámetros experimentales para el caso I.

Cuando T = 10ms, los parámetros discretos *constante de tiempo del sistema*, *sobre paso* y *tiempo de asentamiento al 5%* son de 0.917 segundos, 11% y 4.63 segundos respectivamente. Si se comparan con los parámetros continuos de la tabla 5.2, se puede notar que concuerdan. Es decir, el modelo en tiempo discreto es equivalente al modelo en tiempo continuo para este valor de T.

Para este mismo periodo de muestreo, los resultados experimentales describen un sistema en lazo cerrado con una *constante de tiempo del sistema*, un *sobre paso* y un *tiempo de asentamiento al 5%* de 1.45 segundos, 9.57 % y 5.36 segundos respectivamente. De estos datos se puede concluir que el sistema de control se comporta un 15% más lento que el modelo y con un sobrepaso de 0.87 veces el sobrepaso del modelo. Estas diferencias se deben a incertidumbres en la medición, así como limitaciones del la interface.

La figura 4.8 muestra la respuesta del modelo continuo ante un cambio escalón en el valor deseado para el caso I. Esta figura puede ser comparada con la figura 4.12 que muestra la misma infomación pero para el modelo discreto cuando T=10ms. Se nota que son muy similares. Finalmente en la figura 5.1, se muestra el trazo de la variable controlada del sistema ante un cambio tipo escalón de 2.5 voltios de amplitud. La forma es la misma que la de los casos anteriores.

En la tabla 5.3 se presentan los mismos parámetros para periodos de muestreo desde 10ms hasta 1 segundo. Las figuras 5.1 a 5.5 muestran como varía el trazo de la variable controlada del sistema al cambiar T, experimentalmente. La idea es comparar los datos experimentales con los parámetros discretos obtenidos de las figuras 4.12 a 4.16. Estos parámetros se presentan el la tabla 5.3. Por ejemplo, cuando T = 750ms, los parámetros discretos de *constante de tiempo del sistema*, *sobre paso* y *tiempo de asentamiento al 5%* son de 0.7 segundos, 31.6 % y 9.2 segundos respectivamente. Recordando las especificaciones de diseño, se nota que el sobrepaso esperado es 2.9 veces menor que el parámetro discreto. Además, *tiempo de asentamiento al 5%* de diseño es la mitad del parámetro discreto en la tabla 5.3. Se observa que conforme aumenta el periodo de muestreo, el modelo discreto describe un sistema más oscilatorio, que a la vez tarda más en entrar en la franja del 5% de error.

La tabla 5.3 muestra que la *constante de tiempo del sistema* experimental, el *sobrepaso* experimental y el *tiempo de asentamiento al 5%* experimental son de 1.36 segundos, 11.3% e indeterminado, respectivamente cuando T = 750 ms. La figura 5.4 muestra la gráfica de la variable controlada obtenida experimentalmente en este caso. Según se puede notar, la variable controlada no tiende a ningún valor, y más bien es llevada de un valor a otro en el vecindario del valor final deseado. Es por ello que el tiempo de asentamiento del sistema es indeterminado.

Esto indica que para T = 750ms, el sistema está en el límite de estabilidad y se ha vuelto oscilatorio. Las figura 4.15 muestra la respuesta del modelo discreto ante un escalón unitario para el caso I, cuando se está en periodo de muestreo de 750ms. Como se nota según el modelo; el sistema debe volverse inestable con forme T aumenta. Esta misma conclusión se puede extraer de los datos experimentales. Recuerdese que la referencia del caso I es la figura 4.8, que es el gráfico de la respuesta del modelo continuo..

5.2.2 Caso II. Planta de segundo orden con tiempo muerto 't_m' = 10ms.

Ahora se estudiará una planta con un retardo de transporte de 10ms. La tabla 5.4 muestra los parámetros de la respuesta; *constante de tiempo del sistema*, *tiempo muerto*, *sobrepaso* y *tiempo de asentamiento al 5%*, según las especificaciones de diseño y los parámetros continuos.

	Constante de tiempo del sistema (segundos)	Tiempo muerto (segundos)	Sobre paso (%)	Tiempo de asentamiento al 5% (segundos)
Especificaciones de diseño	0.5	0.01	0	1.51
Modelo continuo	0.95	0.01	11	4.57

Tabla 5.4 Especificaciones de diseño y parámetros continuos de la respuesta del modelo ante un escalón unitario para las condiciones del caso II.

Al igual que el caso I, los parámetros continuos no cumplen con las especificaciones de diseño. De nuevo la variación en la aplicación del algoritmo de control PID en el bloque de función del PLC es la causa de esta diferencia. Entonces, para el caso II, los parámetros continuos serán las nuevas especificaciones de diseño.

En la tabla 5.5 se muestran los parámetros discretos del sistema y los resultados experimentales para varios periodos de muestreo T, cuando la planta tiene un retardo de transporte de 10ms. Según el modelo discreto, el sistema se hace más oscilatorio a medida que aumenta T. El mismo comportamiento presentan los datos experimentales. Por ejemplo para T = 10ms, el *tiempo de asentamiento al 5%* es, experimentalmente, de 6.2 segundos, mientras que para T = 750ms, la variable controlada ya no converge a ningún valor y el sistema está en el límite de estabilidad. La figura 5.9 muestra este resultado gráficamente.

La figura 4.20 muestra la misma información que la figura 5.9, para el modelo discreto del sistema. Se nota que aunque la variable converge al valor deseado, ésta lo hace mucho más lento en comparación con el comportamiento de la figura 4.17.

Periodo de muestreo (segundos)	**Constante de tiempo del sistema (segundos)**	**Tiempo muerto (segundos)**	**Sobre paso (%)**	**Tiempo de asentamiento al 5% (segundos)**
0.01	(0.95 ; 1.67)	(0.01 ; 0.01)	(10 ; 7.8)	(4.47 ; 6.2)
0.1	(0.95 ; 1.95)	(0.01 ; 0.01)	(11 ; 6.4)	(4.65 ; 5.5)
0.5	(1.3 ; 1.3)	(0.01 ; 0.01)	(13 ; 4.7)	(10 ; 4)
0.75	(0.95 ; 1.78)	(0.01 ; 0.01)	(31 ; 12.7)	(12.3 ; indet)

Tabla 5.5 Parámetros discretos y resultados experimentales para el caso II. Los datos son dados de la siguiente forma: (parámetro discreto ; resultado experimental).

Si comparamos los resultados experimentales con los parámetros discretos podemos ver que el modelo discreto describe el sistema real hasta que T = 100ms. Después de ese valor el sistema ya no es descrito de forma correcta por el modelo.

5.2.3 **Caso III.** Planta de segundo orden con tiempo muerto **'tm'** = 1s.

Finalmente, se tiene el caso de una planta con un retardo de transporte de 1s. La tabla 5.6 muestra los parámetros de la respuesta; *constante de tiempo del sistema*, *tiempo muerto, sobre paso y tiempo de asentamiento al 5%,* según las especificaciones de diseño y los parámetros continuos.

Las especificaciones de diseño que se tienen son: una *constante de tiempo del sistema* de 0.5 segundos y un *sobre paso* nulo. Sin embargo, en la tabla se puede notar que no es posible sintonizar; mediante el método de **sítesis de controadores**; el algoritmo PID ideal tal que se cumpla con esas especificaciones. En lugar de ello se obtendrá una respuesta del sistema, ante un cambio escalón unitario en el valor consigna, que presentará un *sobre paso* del 18% y una *constante de tiempo* de 0.94 segundos.

	Constante de tiempo del sistema (segundos)	Tiempo muerto (segundos)	Sobre paso (%)	Tiempo de asentamiento al 5% (segundos)
Especificaciones de diseño	0.94	1	18	5
Modelo continuo	1.31	1	30	6

Tabla 5.6 Especificaciones de diseño y parámetros continuos para el caso III.

Esto quiere decir que el sistema se vuelve más osciliatorio y lento conforme el tiempo muerto de la planta aumenta. La explicación de esta limitación del método de sintonización es que el algoritmo que fue utilizado para encontar las ecuaciones que dan los valores de los parámetros T_i, T_d, y K_c no es exacto. El factor exponencial en las funciones de transferencia de los procesos con retardo de transporte es aproximado, en el algoritmo, por medio de un polinomio de Taylor de orden uno. Dicha aproximación es mala para valores del **retardo de transporte** comparables a la *constante de tiempo* del proceso. Entonces, en estos casos, el método de

síntesis de controladores fallará, y el controlador no será debidamente sintonizado para logar que el sistema cumpla con las especificaciones de diseño.

Los parámetros continuos describen una respuesta de un sistema aun más oscilatorio y lento. Por ejemplo, el *sobre paso* del modelo continuo del sistema es del 30%. Como siempre, los parámetros continuos serán las nuevas espacificaciones de diseño, y contra éstos será comparada la respuesta experimental. La figura 4.10 presenta la respuesta del modelo continuo ante un cambio escalón unitario en el valor deseado bajo las condiciones del caso III.

La tabla 5.7 muestra los parámetros discretos y los resultados experimentales para el caso III, cuando el periodo de muestreo varía de 100ms a 1.2 segundos.

Periodo de muestreo (segundos)	**Constante de tiempo del sistema (segundos)**	**Tiempo muerto (segundos)**	**Sobre paso (%)**	**Tiempo de asentamiento al 5% (segundos)**
0.1	(1.45 ; 1.77)	(1 ; 0.77)	(32.5 ; 39.5)	(6.2 ; 10.6)
0.5	(1.3 ; 1.66)	(1 ; 0.89)	(47 ; 35.79)	(20.9 ; 37.3)
1.2	(1.7 ; 1.89)	(1.2 ; 1.67)	(74 ; 42.2)	(35.4 ; indet)

Tabla 5.7 Parámetros discretos y resultados experimentales para el caso III. Los datos son dados de la siguiente forma: (parámetro discreto ; resultado experimental).

En la tabla se nota que el sistema es mucho más sensible al aumento del tiempo de muestreo en el caso III que en los casos anteriores. Por ejemplo, el *tiempo de asentamiento al 5%* de la varibale controlada cuando el periodo de muestreo es igual a 100ms es de 10.6 segundos en el caso III, mientras que para el caso I, bajo las mismas condiciones, *el tiempo de asentamiento al 5%* era de 6 segundos. Es decir, si el controlador está mal sintonizado; entonces el sistema será más sensible a aumentos en el periodo de muestreo, y por ello más propenso a la inestabilidad.

Las figuras 5.10, 5.11, 5.12 muestran los resultados experimentales para el caso III gráficamente.

Capítulo 6: Conclusiones y recomendaciones

6.1 Conclusiones.

1- Los algoritmos PID fueron aplicados, de manera satisfactoria, en el controlador lógico programable Allen Bradley SLC 500, en el sistema de control realimentado. Utilizando los módulos de entradas/salidas analógicas, fue posible que el PLC obtuviera información del proceso (monitorear la variable controlada) y enviara la señal de control al actuador (salida del controlador). Se presentó la necesidad de realizar una conversión de la señal de corriente, del módulo de salidas analógicas, a una señal de voltaje para que alimentara la planta. Un PLC puede ser utilizado en un sistema de control de lazo cerrado siempre que cuente con módulos analógicos de entradas y salidas. Un aspecto interesante es, que es posible utilizar varios bloques de función PID en un mismo programa, por lo que un PLC es capaz de controlar tantas variables como canales entrada/salida posean sus módulos.

2- Se comprobó que el método de sintonización por síntesis de controladores funciona satisfactoriamente para sintonizar los parámetros del bloque de función PID del PLC. Además, se encontró que el método falla al sintonizar el algoritmo PID, cuando el proceso presenta un retardo de transporte de una magnitud comparable a su constante de tiempo. Se recomienda, para estos casos, recalcular las expresiones de los parámetros T_i, T_d y K_c dadas en el apéndice **A.2.1.2.1**; utilizando un polinomio de Padé de orden entre 7 y 15 para la aproximación del retardo de transporte.

3- Se realizó la conexión entre el PLC y la computadora mediante la aplicación RSLinx, de tal forma que los datos de los archivos de salida y entrada pudieron ser monitoreados desde el PC. Esto facilitó la resolución de los problemas propios de la programación del PLC. Además, permite llevar registros del estado del proceso desde cualquier computador conectado a Internet. En las nuevas tendencias de las organizaciones, donde las altas gerencias impulsan el concepto de fábricas transparentes, los PLC se convierten en una gran opción, pues conforman una fuente de datos sobre los procesos, en tiempo real.

4- Se completó con éxito una interface al usuario utilizando un PC y la aplicación RSView32. Esta interface da acceso absoluto al usuario sobre los parámetros del controlador y se convierte en un ambiente de trabajo interactivo que incluso permite desplegar los valores de la variable controlada y la salida del controlador en gráficas en función del tiempo. Esta interface es sólo un ejemplo de la gran cantidad de opciones de despliegue de procesos que esta aplicación brinda, convirtiendo el PLC en un máquina muy versátil. El usuario puede realizar una aplicación RSView32 adecuada a su necesidad, facilitando a los operarios la comprensión de su proceso. Un aspecto muy importante es que la interface da acceso a la configuración del PLC desde puntos remotos, lo que evita la innecesaria movilización de personal por las instalaciones industriales.

5- Experimentalmente se encontró que, para obtener un buen control, es necesario que la constante de tiempo del proceso sea entre de entre 15 y 20 veces más lenta que el tiempo entre evaluaciones sucesivas del bloque de función PID. Sin embargo, el tiempo de actualización del bloque PID no puede ser menor al tiempo de ciclo de programa, por lo que existen limitaciones al uso de un PLC en procesos muy rápidos. El usuario debe tomar en cuenta este aspecto a la hora de decidir entre utilizar un controlador PID electrónico ó un PLC.

6- Tanto el programa de aplicación RSLogix como el programa de aplicación RSView32, realizados para el proyecto, permitirán a cualquier estudiante implementar rápidamente un sistema de control de lazo cerrado con un algoritmo PID utilizando este PLC. Además, la información suministrada en el capítulo 3 le permitirá adquirir conocimientos sobre los componentes de la familia SLC 500, sus conexiones y la lógica del lenguaje en escalera. Gracias a que el uso de las aplicaciones RSLogix y RSView32 se detallan paso a paso en el capítulo 4, el interesado podrá familiarizarse con la programación del PLC y la elaboración de la interface.

7- El modelo en tiempo discreto del sistema, aproximó bien el comportamiento real del equipo, ante los cambios en el periodo de muestreo. Esto comprueba que los modelos matemáticos utilizados; como el muestreador ideal y el retenedor de orden cero; son aproximaciones aceptables de los componentes electrónicos reales que se encuentran en los módulos de entradas y salidas analógicas del PLC SLC 500. Además, se comprobaron los conceptos teóricos sobre estabilidad y modelado de procesos que se imparten en el curso de sistemas en tiempo discreto, en la Escuela.

8- Los resultados del proyecto son muy importantes, pues extienden la amplia gama de aplicaciones de los PLC. Tradicionalmente utilizados en el control secuencial, se concluye que son igualmente efectivos en el control realimentado, extendiendo todas sus capacidades y ventajas a esta rama del control automático.

6.2 Recomendaciones.

Se debe tener suficiente información de la aplicación que se implementará antes de decidir cual es el PLC apropiado para ser utilizado como un controlador en un lazo cerrado. Entre los aspectos que se deben conocer está la constante de tiempo del proceso. El periodo entre actualizaciones de las imagenes de entrada y salida de la instrucción PID del PLC debe ser entre 15 y 20 veces más rápido que la constante de la planta. Esto con el fin de que el controlador digital no afecte la estabilidad del lazo de control.

Otro aspecto al que se le debe poner atención, cuando se escoje el PLC a utilizar, es el nivel de voltaje de las señales analógicas que provienen de los transductores. Dichos niveles debe ser adecuados a las especificaciones de los módulos de entradas y salidas analógicas del PLC. También es importante realizar las conversiones entre tipos de señales. Por ejemplo, en el sistema que se implementó aquí, la señal de salida analógica del controlador es una señal de corriente. Si el actuador que se utilizará recibe una entrada de voltaje, un circuito como el que se muestra en la figura 4.2, debe ser utilizado.

Antes de escoger un método de sintonización, se debe tener claro la ecuación del bloque de función PID. En caso necesario, se deben realizar las transformaciones de parámetros necesarias.

Con el fin de prevenir daños al equipo y accidentes, el parámetro que varía el periodo de actualización del bloque PID, en el programa de aplicación del PLC, debe estar protegido de tal manera que no pueda ser modificado por personal no calificado. Como se concluyó, la estabilidad del sistema se ve afectada por este parámetro.

Cuando se le suspende la alimentación al PLC, su salida de control hacia el actuador de la planta cae a cero. Para procesos delicados donde equipo o personal estén en riesgo, es importante suplir al PLC de energía suplementaria durante una suspensión de energía no programada. Esto con el fin de que el PLC sea capaz de llevar el sistema de manera controlada a un estado seguro.

Bibliografía

1. Allen Bradley. – "SLC 500: 4-Channel Analog I/O Modules", Publicación 1746-UM0058-EN-P, Estados Unidos, 2004.
2. Allen Bradley. – "SLC 500: 4-Channel Analog I/O Modules", Publicación 1747-IN008C-EN-P, Estados Unidos, 2004.
3. Allen Bradley. – "SLC 500: Instruction Set Reference Manual", Publicación 1747-RM001D-EN-P, Estados Unidos, 2003.
4. Allen Bradley. – "SLC 500: Modular Chassis and Power Supplies", Publicación 1746-TD003A-EN-P, Estados Unidos, 2000.
5. Allen Bradley. – "Digital I/O Modules", Publicación 1746-IN005B-EN-P, Estados Unidos, 2002.
6. Allen Bradley. – "SLC 500: Power Supplies", Publicación 1746-IN004D-MU-P, Estados Unidos, 2003.
7. Allen Bradley. – "SLC 500: Modular Hardware Style", Publicación 1747- UM011E-EN-P, Estados Unidos, 2004.
8. Castro, L; Gómez, J. –"Puesta en marcha y elaboración de una plataforma de simulación para el PLC SLC 5/05 de Allen Bradley", Proyecto eléctrico, Universidad de Costa Rica, 2004.
9. Kuo, B.C. - "Sistemas de control automático", Prentice Hall Hispanoamericana, México,1996.
10. Ogata, K. - "Ingeniería de control moderna", Pearson Educación, S.A., España, 4ta edición, 2003.
11. Ogata, K. – "Discrete-time control systems", Prentice Hall, Inc., Estados Unidos, 2da edición, 1995.

Referencias

1. Aguilar, V. –"Análisis de las diferencias entre la CPU de un PLC y la de un computador industrial con Windows CE, en el desempeño de tareas de control y monitoreo", Proyecto eléctrico, Universidad de Costa Rica, 2000.
2. Canales, R., Barrera R. –" Análisis de sistemas dinámicos y control automático", Limusa, México, 1977.
3. Coughlin, R. –" Amplificadores Operacionales", Prentice Hall, México, 1999.
4. Kuo, B.C. - "Sistemas de control automático", Prentice Hall Hispanoamericana, México,1996.
5. Kuo, B.C. - "Sistemas automáticos de control", C.E.C.S.A., México,1970.
6. Loría, G., Mazón, I. –"Modelado y Análisis de Sistemas Dinámicos", Editorial de la Universidad de Costa Rica, Costa Rica, 1989.
7. Mazón, I. –"Sistemas en tiempo discreto", Notas del Curso: Sistemas en tiempo discreto impartido en la Universidad de Costa Rica, Costa Rica, 1999.
8. Murillo, L. –"Uso de controladores programables", Proyecto eléctrico, Universidad de Costa Rica, 1986.
9. Murillo, R. –"Diseño y montaje de un controlador discreto de posición para el carrusel de la celda de manufactura, utilizando el microcontrolador 8051", Proyecto eléctrico, Universidad de Costa Rica, 2000.
10. Nise, N.S. - "Sistemas de Control para Ingeniería", Editorial Continental, México, 2002.
11. Ogata, K. - "Ingeniería de control moderna", Pearson Educación, S.A., España, 4ta edición, 2003.
12. Ogata, K. – "Discrete-time control systems", Prentice Hall, Inc., Estados Unidos, 2da edición, 1995.
13. Orozco, R. –" Introducción al control automático", Publicaciones de la Universidad de Costa Rica, Costa Rica, 1975.
14. Quirós, J. –" Pruebas experimentales de sintonización de controladores PID", Proyecto eléctrico, Universidad de Costa Rica, 2000.

Páginas electrónicas:

1. The MathWorks, Inc., Netick, Massachusetts, EUA, http://www.mathworks.com
   10 de septiembre de 2004.
2. Allen Bradley, EUA, http://www.ab.com.
   25 de septiembre de 2004.

Apéndice

A.1 Introducción a los sistemas de control.

A.1.1 Tipos de señales.

En los procesos industriales modernos existen diferentes tipos de señales las cuales pueden ser de control, medición, etc. A continuación se realizará una clasificación de las señales eléctricas según su comportamineto en el tiempo.

A.1.1.1 Señales continuas.

Una señal en tiempo continuo es aquella que se define sobre un intervalo continuo de tiempo. Esto es, existe un valor de magnitud de la señal para cada instante del tiempo.

La amplitud puede tener un intervalo continuo de valores (*señal analógica*) o solamente un número finito de valores distintos (*señal cuantizada*). El proceso de representar la amplitud de la señal por medio de un conjunto de valores finitos se denomina cuantificación. En la figura A.1 se muestran estos dos tipos de señales.

A.1.1.2 Señales discretas.

Una señal discreta esta definida sólo para ciertos instantes de tiempo (esto es, aquellos en los que la variable independiente *t* está cuantificada). La amplitud de la señal puede tener un intervalo continuo de valores ó un número finito.

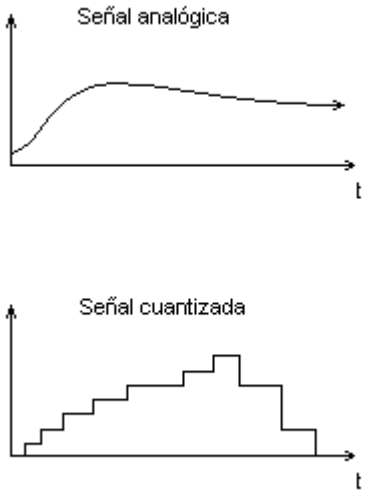


Figura A.1 Señales analógica y cuantizada.

En una señal en tiempo discreto, si la amplitud adopta valores en un intervalo continuo, se denomina *señal de datos muestreados.* Una señal de datos muestreados se puede generar tomando los valores de amplitud de una señal analógica en instantes determinados de tiempo. Una señal muestreada es una señal de pulsos modulada en amplitud.

Una *señal digital* es una señal en tiempo discreto con amplitud cuantificada. Dicha señal se puede representar mediante una secuencia de números, por ejemplo, una secuencia de números binarios. La figura A.2 se muestran estos dos tipos de señal.

En la práctica, muchas señales digitales se obtienen mediante el muestreo de señales analógicas que después se cuantifican. La cuantificación es lo que permite que estas señales analógicas sean leídas como palabras binarias por un procesador digital.

Los términos como “sistemas de control en tiempo discreto”, “sistemas de control de datos muestreados” y “control digital” implican el mismo tipo o tipos muy similares de sistemas de control. Hablando en forma precisa, existen diferencias en estos sistemas. Por ejemplo, en un “sistema de control de datos muestreados” hay tanto señales en tiempo continuo como en

tiempo discreto; las señales en tiempo discreto están moduladas en amplitud por una señal de pulsos. Por otra parte, los sistemas de control digital incluyen señales en tiempo discreto que están codificadas en forma de secuencias numéricas binarias.

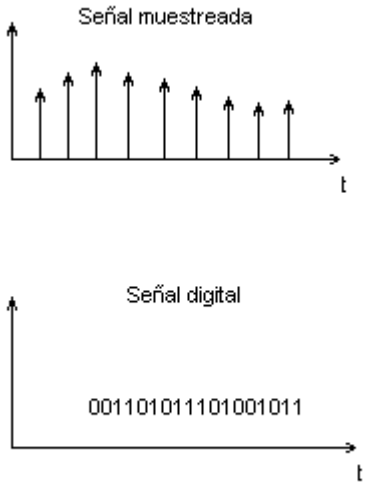


Figura A.2 Señales muestreada y digital.

A.2 Modelado matemático de los sistemas de control.

El análisis matemático de los sistemas de control realimentado se efectúa, generalmente, por medio de las **funciones de transferencia** de cada una de las partes del sistema.

Todo proceso puede ser modelado matemáticamente por una ecuación diferencial. Entonces, aplicando la transformada de Laplace a la ecuación diferencial; con condiciones iniciales cero, se puede, despejado la razón **variable de salida/variable de entrada**, encontrar una relación algebraica en términos de la variable compleja 's'. Esta relación se conoce como *función de transferencia del proceso*.

A.2.1 Sistemas de control en tiempo continuo.

A.2.1.1 Función de transferencia de un sistema de control.

En un sistema realimentado, la función de transferencia del lazo cerrado relaciona la variable controlada con la entrada de valor consina y las pertubaciones. La figura A.3 muestra el diagrama de bloques de un sistema realimentado.

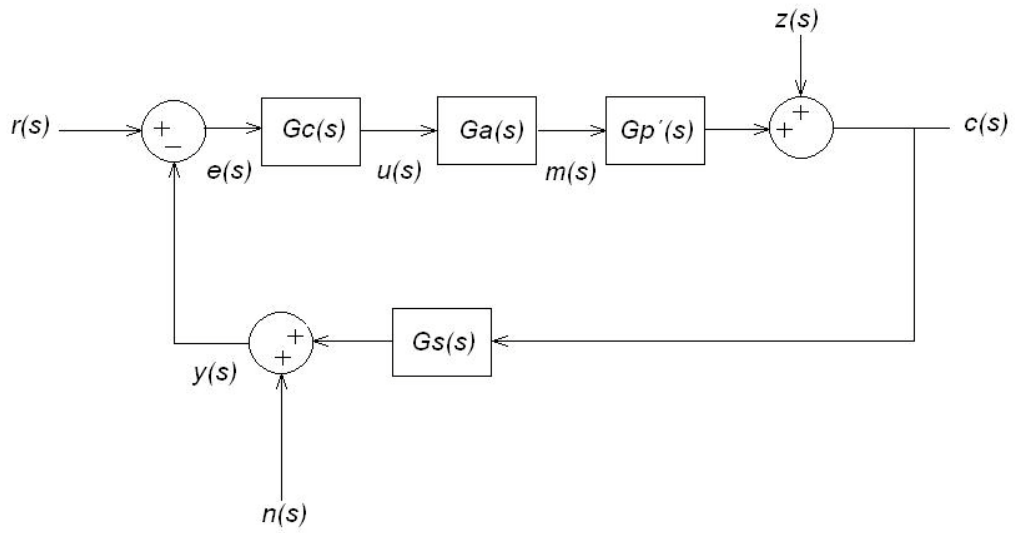


Figura A.3 Sistema de control Realimentado.

Aquí:

-c(s): variable controlada.

-r(s): valor consigna o valor deseado.

-e(s): error, es el valor consigna menos la señal realimentada .

-u(s): salida del controlador.

-y(s): señal realimentada.

-m(s): variable manipulada.

-z(s): perturbación externa.

-n(s): ruido de medición.

Se distinguen varios bloques en la figura. Cada uno de ellos representa un elemento del sistema de control mediante una función de transferencia. La descripción de cada bloque es:

Gc(s): Función de transferencia del controlador. Modela el algoritmo de control.

Ga(s): Función de transferencia del actuador.

Gp'(s): Función de transferencia del proceso o planta.
Gs(s): Función de transferencia del sensor transmisor.

Generalmente existen dos tipos de sistemas: el *servosistema* y el *regulador*. En el *servosistema,* la variable controlada sigue un valor consigna. En el *regulador* se busca que la variable controlada no varíe su valor ante perturbaciones externas que afecten el sistema.

Por ejemplo, la variable controlada pondría tratarse de una presión. Entoces, el objetivo de un *servosistema* sería llevar esa presión de un valor a otro, según lo indica la señal consigna. Esta señal puede ser dada por un operador a través de un panel de mando. Un aspecto importante a tomar en cuenta es que realmente el controlador no monitorea la variable física de presión, sino una señal realimentada que proviene del sensor transmisor. De igual forma, la señal de salida del controlador no afectará de manera directa la variable manipulada, sino que lo hará a través de un actuador. Es posible realizar una serie de simplificaciones al diagrama anterior, de tal manera que sea más sencillo realizar el análisis. Se incluirán los bloques del actuador, del sensor y la planta en un único bloque llamado Gp(s). El diagrama de bloques simplificado se muestra en la figura A.4.

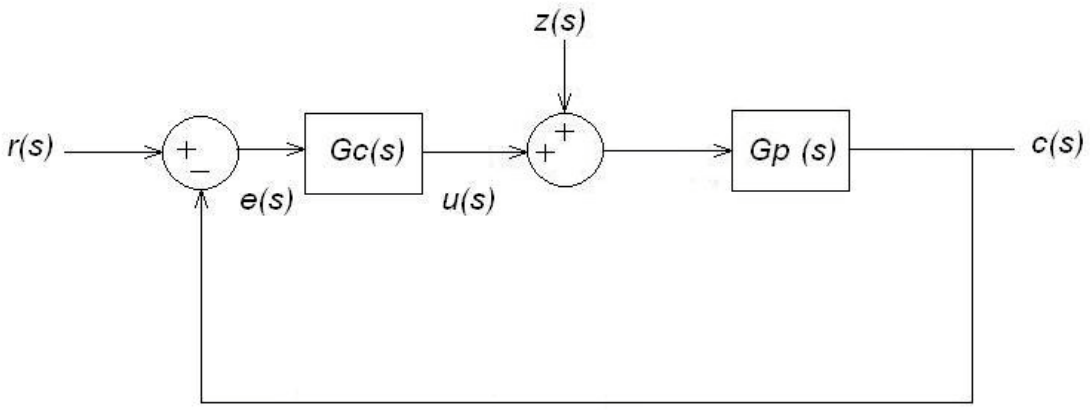


Figura A.4 Diagrama en bloques de un sistema realimentado.

La ecuación (A.1) muestra a Y(s) como función de las entradas R(s) y Z(s).

$$Y(s) := \frac{Gc(s)\cdot Gp(s)}{1 + Gc(s)\cdot Gp(s)}\cdot R(s) + \frac{Gp(s)}{1 + Gc(s)\cdot Gp(s)}\cdot Z(s) \qquad (A.1)$$

Aquí se analizará el comportamiento del sistema solamente como servomecanismo, por lo que en los cambios en la entrada de perturbación Z(s) se desprecian. Con esta simplificación, el segundo término de la ecuación puede ser eliminado. La ecuación (A.2) muestra la relación final entre la variable realimentada y el valor consigna. De aquí en adelante la relación (A.2) es la **función de transferencia del sistema**.

$$Y(s) := \frac{Gc(s)\cdot Gp(s)}{1 + Gc(s)\cdot Gp(s)}\cdot R(s) \qquad (A.2)$$

A.2.1.2 Algoritmo de control proporcional, integral, derivativo ó PID.

El bloque Gc(s) corresponde a la función de transferencia del controlador. Existen diferentes tipos de función dependiendo del tipo de controlador que se implementa y del tipo de comportamiento que se quiere que el sistema tenga. Entre los algoritmos más comunes está el algoritmo PID, el cual utiliza tres operaciones matemáticas que son aplicadas sobre el error o sobre la variable realimentada, El propósito del algoritmo es obtener información del comportamiento de la planta, y poder así aplicar una señal al actuador para lograr el comportamiento deseado de la variable controlada. A cada una de estas operaciones se le llama modo de operación. Específicamente, existen el modo proporcional, el modo derivativo y el modo integral.

Modo proporcional: consiste en aplicar una ganancia al error de tal forma que la salida del controlador sea proporcional al error. El parámetro del controlador asociado a este modo es K_c, un valor sin unidades.

Modo integral: Aplica la operación integral a la señal de error. Sirve para medir la magnitud acumulada del error. Su aporte al valor de la señal de salida del controlador es cero sólo si el error es cero. El parámetro del controlador asociado a este modo es T_i, que se define en unidades de segundos. El T_i se conoce como tiempo integral y es el tiempo que le tardaría a la salida del modo integral alcanzar el valor en magnitud de la salida del modo proporcional.

Modo derivativo: Este modo deriva el error. Por su naturaleza, mide la rapidez con que el error cambia instantáneamente. El parámetro relacionado directamente a este modo es T_d, conocido como tiempo derivativo, y es el tiempo que se adelanta la salida del controlador respecto del modo proporcional. Está dado en valores de segundos.

En ciertas aplicaciones donde se esperan cambios repentinos (llamados de tipo escalón) en el valor deseado, este modo se aplica sólo a la señal realimentada. Luego su salida se resta de la suma de las salidas de los otros modos. Esto se realiza así para evitar sobreesfuerzos del actuador de la planta.

La ecuación que modela la forma en que el algoritmo actúa es:

$$u(t) := K_c \cdot \left[e(t) + \frac{1}{T_i} \cdot \left(\int_0^t e(\tau)\, d\tau \right) + T_d \cdot \frac{d}{dt} e(t) \right] \qquad (A.3)$$

A esta ecuación se le conoce como ecuación del PID ideal. Sin embargo, existen muchas otras ecuaciones que relacionan los modos de manera diferente. Por ejemplo la ecuación del PID serie ó la del PID paralelo.

Aplicando la transformada de Laplace se encuentra la función de transferencia del controlador PID ideal.

$$U(s) := K_c \cdot \left(1 + \frac{1}{T_i \cdot s} + T_d \cdot s \right) \cdot E(s) \qquad (A.4)$$

A.2.1.2.1 Sintonización de un controlador PID.

El diseño de los sistemas de control se basa en lograr un cierto comportamiento del sistema. Existen criterios para medir ese comportamiento basados en cómo cambia la magnitud de la variable realimentada con el tiempo.

La figura A.5 muestra un posible comportamiento de la magnitud de una variable realimentada en el tiempo ante un cambio tipo escalón unitario en la variable consigna. La figura A.5 se utiliza de ejemplo para mostrar las especificaciones de diseño, y cómo se miden.

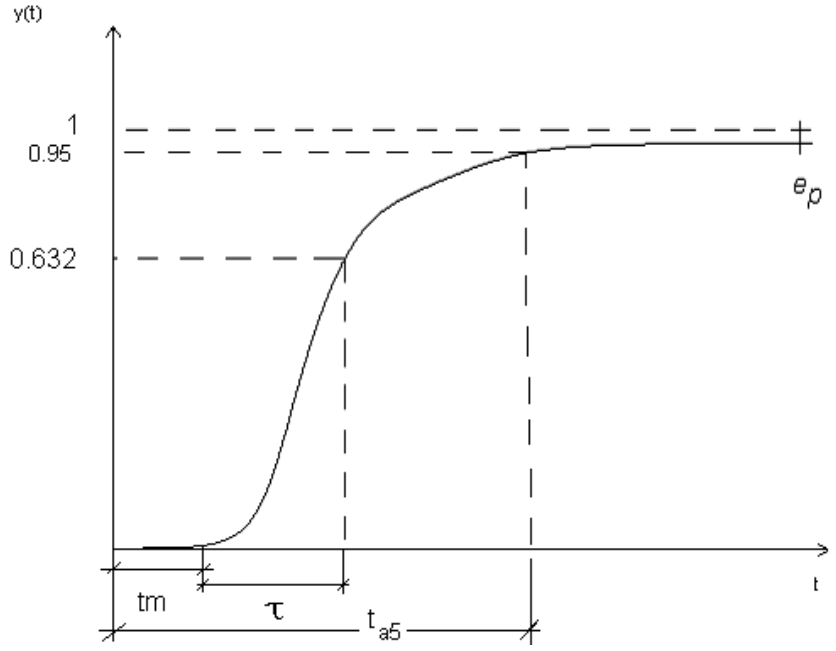


Figura A.5 Cambio de y(t) ante un cambio escalón en el valor consigna r(t).

En el gráfico se distinguen los parámetros de diseño para servomecanismos:

$\mathbf{t_{a5}}$: tiempo de asentamiento al 5%. Es el tiempo que le toma a la variable realimentada entrar en la franja del 5% de error.

$\mathbf{t_m}$: tiempo muerto o retardo de transporte del sistema. Generalmente producto de un tiempo muerto de la planta que se controla.

τ: Constante de tiempo del sistema.

$\mathbf{e_p}$: error en estado permanente. Es la diferencia entre el valor consigna y el valor de la variable realimentada cuando ha pasado un tiempo mucho mayor a τ.

El propósito del controlador **PID** es ayudar a lograr un comportamiento de la variable controlada de manera que se pueda cumplir con las especificaciones de diseño. Para ello, se deben definir los valores K_c, T_i, T_d del controlador. Dicha acción se le llama **sintonización del controlador**.

A.2.1.2.1.1 Método de sintonización por síntesis de controladores.

Una vez que el modelo del proceso es conocido, es posible calcular **una función de transferencia del sistema** que asegure que la variable realimentada cumple con las especificaciones de diseño.

El método de sintonización por síntesis de controladores sigue esta línea de análisis, y plantea el problema de encontrar la función de transferencia del controlador necesaria para que; dado un modelo del proceso; se obtenga la función de transferencia del sistema deseada.

Sea **Mr(s)** la **función de transferencia del sitema** deseada, y **Gp(s)** la función de transferencia del modelo del proceso. La función de transferencia del controlador es:

$$Gc(s) := \frac{1}{Gp(s)} \cdot \left(\frac{Mr(s)}{1 - Mr(s)} \right) \tag{A.5}$$

Por ejemplo; se desea controlar un proceso cuyo modelo es una función de transferencia de segundo orden más tiempo muerto:

$$Gp(s) := \frac{e^{-t_m \cdot s}}{(\tau_1 \cdot s + 1) \cdot (\tau_2 \cdot s + 1)} \tag{A.6}$$

Además, se quiere que el sistema de control se comporte como un sistema de primer orden con constante de tiempo τ_c; más un tiempo muerto t_m. Para simplificar las ecuaciones, se escogerá el t_m del sistema igual al retardo la planta. La función de transferencia del sistema, Mr(s), es:

$$Mr(s) := \frac{e^{-t_m \cdot s}}{\tau_c \cdot s + 1} \qquad (A.7)$$

Aplicando la ecuación A.5, se obtiene la siguiente función de transferencia del controlador:

$$Gc(s) := K_{cs} \cdot \left(\frac{T_{is} + 1}{T_{is} \cdot s} \right) \cdot \left(T_{ds} \cdot s + 1 \right) \qquad (A.8)$$

con:

$$T_{is} = \tau_1 \qquad T_{ds} = \tau_2 \qquad K_{cs} := \frac{\tau_1}{\left(\tau_c + t_m \right)}$$

A.2.1.2.1.2 Conversión de parámetros algoritmos PID.

La ecuación (A.8) es conocida como el algoritmo de control PID serie. Esto significa que el método de síntesis de controladores está diseñado para algoritmos PID serie.

Sin embargo, en caso de que el controlador de que se dispone aplique un algoritmo PID ideal, es posible, aun, utilizar el método de sintonización anterior. Una vez sintonizado el controlador serie para los parámetros de diseño, se puede hallar los parámetros T_i, T_d, K_c de un controlador PID ideal con las siguientes ecuaciones de transformación:

$$K_c := k_{cs} \cdot \left(1 + \frac{T_{ds}}{T_{is}} \right) \qquad T_i := T_{is} \cdot \left(1 + \frac{T_{ds}}{T_{is}} \right) \qquad T_d := \frac{T_{ds}}{1 + \frac{T_{ds}}{T_{is}}} \qquad (A.8)$$

A.2.2 Sistemas de control en tiempo discreto.

Los sistemas de control en tiempo discreto son aquellos sistemas en los cuales una ó más de las variables es una señal discreta. Se habló en el apartado **A.1.1.2** que una señal discreta asigna magnitud sólo a ciertos valores de t. Si esta señal o función discreta tuviera un valor x en momentos periódicos de tiempo, sería posible definirla como una sucesión de valores cada k*T *(k* = 0, 1, 2,...) segundos. Estos instantes pueden especificar los tiempos en los que se lleva a cabo alguna medición de tipo físico o los tiempos en los que se extraen los datos de la memoria de una computadora. El intervalo de tiempo entre estos dos instantes discretos se supone que es lo suficientemente corto, de modo que el dato, para el tiempo entre éstos, se pueda aproximar mediante una interpolación sencilla.

Si en el sistema de control está involucrada una computadora como controlador, los datos muestreados se deben convertir a datos digitales.

Los sistemas en tiempo continuo, cuyas señales son continuas en el tiempo, se pueden describir mediante ecuaciones diferenciales. Los sistemas en tiempo discreto, cuyas señales son datos muestreados o señales digitales, deben ser descritos mediante ecuaciones en diferencias.

A.2.2.1 Proceso de muestreo.

La operación que transforma las señales en tiempo continuo en datos en tiempo discreto se denomina *muestreo* o *discretización.*

El elemento que realiza este muestreo se conoce como muestreador, en el cual una señal continua se muestrea cada T segundos para obtener una secuencia de valores que representa la señal en los instantes de muestreo. Esa secuencia de valores se identificará mediante un asterisco. Por ejemplo para la señal r(t) continua, la secuencia discreta que la representa es r*(t).

Para analizar esta función, se considerará un muestreador ideal, cuya salida es un tren de impulsos, empezando en t=0, cada uno distanciado un t=T, y con una amplitud igual a la amplitud de la señal de entrada en los instantes de muestra. Aquí se considera que r(t)=0 para t<0.

La señal r*(t) se puede representar mediante funciones en tiempo continuo de la siguiente forma:

$$r^*(t)=r(kT)\delta T(t) \tag{A.9}$$

con δT(t) una sucesión de impulsos unitarios definida como:

$$\delta T(t) := \sum_{k=0}^{\infty} \delta T\left(t - k\cdot T\right) \tag{A.10}$$

donde δT(t) es la discretización de la función delta de Dirac para t en el conjunto de los números reales. Esta es la representación matemática de la función del muestreador.

La trasformada de Laplace de la ecuación (A.9) es:

$$X(s) := \sum_{k=0}^{\infty} x(kT)\cdot e^{-k\cdot T\cdot s} \tag{A.11}$$

Ahora si s=(1/T)ln(z) se obtiene:

$$X(z) := \sum_{k=0}^{\infty} x(kT)\cdot z^{-k} \tag{A.12}$$

Lo que significa que la transformada de Laplace de x*(t) evaluada en s = ln(z)/T es la transformada z de x(t).

A.2.2.2 Proceso de retención.

El proceso de retención consiste en generar una señal continua h(t) de una secuencia en tiempo discreto m(kT). Un circuito retenedor convierte la señal muestreada en una señal en tiempo continuo. Se supone que h(t) es igual a m(kT) en los instantes t=kT y t=(k+1)T, tal que la forma de la señal h(t) entre estos dos instantes puede ser aproximada por:

$$h(t) := a_n \cdot \tau^n + .. + a_1 \cdot \tau^n + a_0 \qquad \text{(A.13)}$$

con $t = kT+\tau$.

Entones como h(t) pasa por m(kT), m((k+1)T),…

$$h(kT) = m(kT) \qquad \text{y} \quad a_0 := m(kT) \ . \qquad \text{(A.14)}$$

Si n = 0 se tiene que la ecuación (A.13) se transforma en:

$$h(kT+\tau) = m(kT) \qquad \begin{matrix} 0<\tau<T & \text{para } k=0 \\ kT<\tau<(k+1)T & \text{para } k>0 \end{matrix} \qquad \text{(A.15)}$$

Es la ecuación de un retenedor de orden 0.

Mediante la transformada de Laplace de h(t) se puede hallar la función de transferencia del retenedor de orden cero:

$$G(s) := \frac{1 - e^{-T \cdot s}}{s} \qquad \text{(A.16)}$$

A.2.2.3 Función de transferencia discreta de un sistema realimentado.

Para analizar los sistemas de control de lazo cerrado donde el controlador es un dispositivo digital tal como un PID o un PLC, es conveniente realizar ciertas transformaciones del dominio del tiempo al dominio complejo. Debido a la naturaleza del procesador del controlador digital, las decisiones que el mismo toma sobre la señal de error, para operar sobre el actuador, se realizan en instantes de tiempo discretos marcados por su reloj interno. De este modo, se deben aplicar herramientas de análisis de sistemas en tiempo discreto, aun cuando la planta sea un dispositivo analógico. Es por ello que se utiliza la transformada z como la principal herramienta en el análisis del control digital.

Función de transferencia z de lazo abierto.

La función de transferencia z de los sistemas discretos es la relación de la transformada z de la salida entre la transformada z de la entrada.

Considerese primero cual es la respuesta de un sistema continuo ante una entrada muestreada x*(kT). Se supone que x*(kT) es cero para todo instante t antes de t=0. La función de trasferencia de nuestro sistema continuo es G(s). La salida del sistema es y(t), que es una señal continua. Si se coloca un muestreador ficticio a la salida y(t) se obtendrá un tren de pulsos cuya transformada z es:

$$Y(z) := \sum_{k=0}^{\infty} y(kT)\cdot z^{-k} \qquad \text{(A.17)}$$

Para los sistemas continuos se sabe que la salida está relacionada con la entrada por:

$$y(t) := \int_0^t x(t-\tau)\cdot g(\tau)\, d\tau \tag{A.18}$$

Con g(t) la función respuesta al impulso del sistema. Para sistemas discretos, esta relación es:

$$y(kT) := \sum_{h=0}^{k} x(kT - hT)\cdot g(hT) \tag{A.19}$$

Como los términos de la sumatoria para h>k son cero (pues x(t) = 0 para t<0), la expresión anterior puede ser escrita como:

$$y(kT) := \sum_{h=0}^{\infty} x(kT - hT)\cdot g(hT) \tag{A.20}$$

Como Y(z) es:

$$Y(z) := \sum_{k=0}^{\infty} y(kT)\cdot z^{-1} \tag{A.21}$$

Entonces:

$$Y(z) := \sum_{k=0}^{\infty} \sum_{h=0}^{\infty} x(kT - hT)\cdot g(hT) \tag{A.22}$$

Que se resume como:

$$Y(z) := X(z)\cdot G(z) \tag{A.23}$$

Con G(z) la transformada z de g(t). La ecuación (A.23) es la función de transferencia z del sistema de lazo abierto.

Función de transferencia z del retenedor.

Como se vió, la función de transferencia del retenedor es:

$$Gt(s) = L(gt(t)) = \frac{1 - e^{-Ts}}{s} \qquad \text{(A.24)}$$

Si este retenedor esta en serie con la planta, la transformada z de la combinación es:

$$G(z) = (1 - z^{-1})Z(Gp(s)/s) \qquad \text{(A.25)}$$

Con Gp(s) la transformada de Laplace de la planta y Z( ) la transformada z. La ecuación (A.25) se conoce como equivalente retenedor planta.

Función de transferencia z de lazo cerrado.

Para obtener la función de transferencia de un sistema de lazo cerrado, se debe hacer:
-Visualizar las salidas de los muestreadores como entradas al sistema.
-Todas las otras no entradas del sistema serán salidas.
-Escribir las ecuaciones que representan la salida en términos de las entradas mediante los diagramas de bloques.
-Aplicar la transformada z de estas expresiones para obtener la FT (función de transferencia) z del sistema.

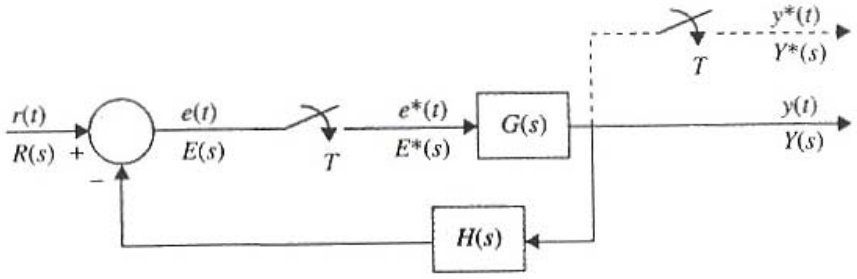


Figura A.6 Sistema discreto en lazo cerrado.

Considere el sistema en tiempo discreto en lazo cerrado mostrado en la Fig. A.6. La salida del muestreador se visualiza como una entrada al sistema. El sistema tiene entradas R(s) y E*(s). Las señales E(s) y Y(s) son las salidas del sistema.

Al escribir las relaciones entre salida y entrada para las salidas E(s) y Y(s) utilizando diagramas de bloques, se obtiene:

$$E(s) = R(s) - G(s)H(s)E^*(s) \tag{A.26}$$

$$Y(s) = G(s)E^*(s) \tag{A.27}$$

Observe que el segundo miembro de las dos últimas ecuaciones contiene solamente las entradas R(s) y E*(s). Si E(s) se muestrea se obtiene

$$E^*(s) = \frac{R^*(s)}{1 + (G(s)H(s))^*} \tag{A.28}$$

Al sustituir la ecuación (A.28) es la ecuación (A.27) se tiene:

$$Y(s) = \frac{G(s)R^*(s)}{1 + (G(s)H(s))^*} \tag{A.29}$$

Discretizando

$$Y^*(s)/R^*(s)=G^*(s)/(1+(G(s)H(s))^*) \tag{A.30}$$

Tomando la transformada z:

$$\frac{Y(z)}{R(z)} = \frac{G(z)}{1 + GH(z)} \tag{A.31}$$

La ecuación (A.31) es la función de transferencia z de lazo cerrado de un sistema discreto.