



UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

FACULTAD DE INGENIERÍA

**IMPLEMENTACIÓN DE UN ALGORITMO DE
CONTROL ADAPTABLE EN UN
MICROCONTROLADOR**

TESIS

QUE PARA OBTENER EL TÍTULO DE:

**Ingeniero en Instrumentación y
Control de Procesos**

P R E S E N T A:

Carlos Alberto Silva Rodríguez

DIRIGIDA POR:

M. en C. Alfonso Noriega Ponce

**CENTRO UNIVERSITARIO
QUERÉTARO, QRO. DICIEMBRE 2006**

BIBLIOTECA CENTRAL
UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

No. ADQ 471264

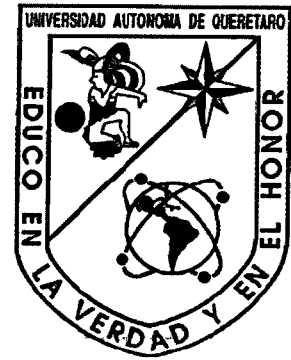
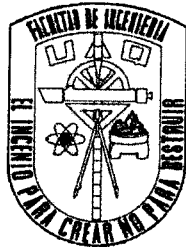
CLAS TS
621.3815
S586i



UNIVERSIDAD AUTÓNOMA DE QUERÉTARO

FACULTAD DE INGENIERÍA

"EL INGENIO PARA CREAR NO PARA DESTRUIR"



**IMPLEMENTACIÓN DE UN ALGORITMO DE CONTROL
ADAPTABLE EN UN MICROCONTROLADOR**

TESIS

Como parte de los requisitos para obtener el título de

Ingeniero en Instrumentación y Control de Procesos

Presenta:

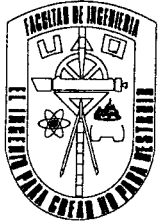
Carlos Alberto Silva Rodríguez

Dirigida por:

M. en C. Alfonso Noriega Ponce



C. U. 20 de septiembre de 2006



C. CARLOS ALBERTO SILVA RODRÍGUEZ
Pasante de Ingeniería Instrumentación y
Control de Procesos

Presente.

Con relación a su oficio enviado al H. Consejo Académico de la Facultad en el que solicita titularse bajo la opción de tesis individual, me permito informarle que en la sesión ordinaria del 20 de septiembre del año en curso, este cuerpo colegiado acordó aceptar la opción de titulación por lo que deberá trabajar en el tema **"Implementación de un Algoritmo de Control Adaptable en un Microcontrolador"**, bajo la dirección del M en C. Alfonso Noriega Ponce.

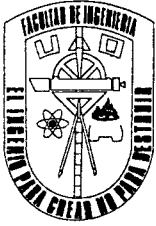
El Contenido Aceptado por el H. Consejo Académico es el siguiente:

ÍNDICE

1. Introducción.
 - 1.1. Planteamiento del problema
 - 1.2. Objetivos del estudio
 - 1.3. Metodología
 - 1.4. Organización de la tesis
2. Conceptos Básicos
 - 2.1. Introducción
 - 2.2. La idea básica de un sistema de control
 - 2.3. La computadora como un elemento de control
 - 2.4. El sistema de control digital de lazo cerrado
 - 2.5. ¿porque el control digital en lugar del analógico?
3. Diseño del algoritmo de control
 - 3.1 Control de seguimiento y regulación
 - 3.2 Identificación recursiva de mínimos cuadrados
 - 3.3 Propiedades del algoritmo de identificación
 - 3.4 Control adaptable directo de sistemas de fase mínima
 - 3.5 Control adaptable con factor de olvido



Universidad Autónoma de Querétaro
Facultad de Ingeniería



4. Topología del sistema de control
 - 4.1 Introducción
 - 4.2 Tarjeta de control y adquisición de datos USB
 - 4.2.1 Microcontrolador PIC18F4550
 - 4.2.2 Protocolo de comunicación USB
 - 4.2.3 Programador Gtp Usb Lite
 - 4.3 Software de soporte
 - 4.3.1 Generación y grabación del firmware
 - 4.3.2 Simulación
 - 4.3.3 Interfaz grafica

5. Desempeño del algoritmo de control
 - 5.1 Modelo de la planta
 - 5.2 Aplicación del controlador lineal
 - 5.3 Aplicación del controlador adaptable

Conclusiones

Bibliografía

También hago de su conocimiento las disposiciones de nuestra Facultad, en el sentido que antes del Examen profesional deberá cumplir con los requisitos de nuestra legislación y deberá imprimir el presente oficio en todos los ejemplares de su tesis.

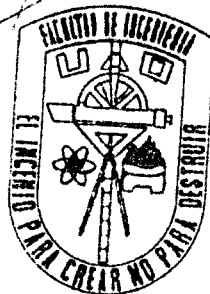
Atentamente

"EL INGENIO PARA CREAR NO PARA DESTRUIR"

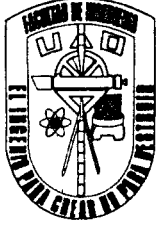
DR. GILBERTO HERRERA RUIZ

Director

c.c.p. Archivo
*GHR/DHM.



DIRECCIÓN



Universidad Autónoma de Querétaro
Facultad de Ingeniería
División de Estudios de Posgrado

CU., 19 de octubre del 2006

Dr. Gilberto Herrera Ruiz
Director de la Facultad de Ingeniería
Presente:

Por este conducto, me permito comunicar a Usted, que una vez revisada la tesis individual titulada "Implementación de un algoritmo de control adaptable en un microcontrolador", del Pasante de la licenciatura en Ingeniería en Instrumentación y Control de Procesos, Carlos Alberto Silva Rodríguez, de acuerdo al artículo 20 del inciso h) del reglamento de Titulación vigente.

Emito mi Voto Aprobatorio.

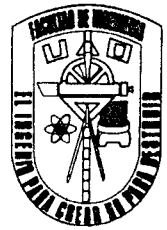
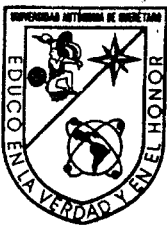


Atentamente,

"EL INGENIO PARA CREAR, NO PARA DESTRUIR"

M. en C. Alfonso Noriega Ponce
Director de Tesis

c.c.p Archivo
*GHR/sar



Universidad Autónoma de Querétaro
Facultad de Ingeniería
División de Estudios de Posgrado

CU., 10 de octubre del 2006

Dr. Gilberto Herrera Ruiz
Director de la Facultad de Ingeniería

Presente:

Por este conducto, me permito comunicar a Usted, que una vez revisada la tesis individual titulada "Implementación de un algoritmo de control adaptable en un microcontrolador", del Pasante de la licenciatura en Ingeniería en Instrumentación y Control de Procesos, Carlos Alberto Silva Rodríguez, de acuerdo al artículo 20 del inciso h) del reglamento de Titulación vigente.

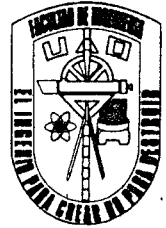
Emito mi Voto Aprobatorio.

Atentamente,

"EL INGENIO PARA CREAR, NO PARA DESTRUIR"

Dr. Rodrigo Castañeda Miranda
Sinodal

c.c Archivo
*GHR/sar



Universidad Autónoma de Querétaro
Facultad de Ingeniería
División de Estudios de Posgrado

CU., 10 de octubre del 2006

Dr. Gilberto Herrera Ruiz
Director de la Facultad de Ingeniería
Presente:

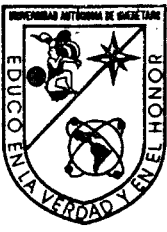
Por este conducto, me permito comunicar a Usted, que una vez revisada la tesis individual titulada "Implementación de un algoritmo de control adaptable en un microcontrolador", del Pasante de la licenciatura en Ingeniería en Instrumentación y Control de Procesos, Carlos Alberto Silva Rodríguez, de acuerdo al artículo 20 del inciso h) del reglamento de Titulación vigente.

Emito mi Voto Aprobatorio.

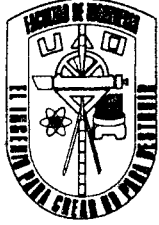
Atentamente,
"EL INGENIO PARA CREAR, NO PARA DESTRUIR"

Dr. Vladimir Rauch Sitar
Sinodal

c.c Archivo
*GHR/sar



Universidad Autónoma de Querétaro
Facultad de Ingeniería
División de Estudios de Posgrado



CU., 10 de octubre del 2006

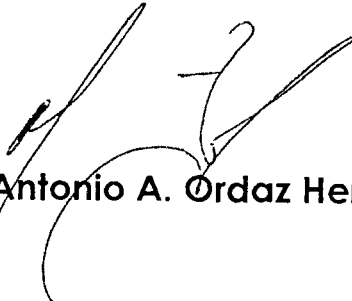
Dr. Gilberto Herrera Ruiz
Director de la Facultad de Ingeniería
Presente:

Por este conducto, me permito comunicar a Usted, que una vez revisada la tesis individual titulada "Implementación de un algoritmo de control adaptable en un microcontrolador", del Pasante de la licenciatura en Ingeniería en Instrumentación y Control de Procesos, Carlos Alberto Silva Rodríguez, de acuerdo al artículo 20 del inciso h) del reglamento de Titulación vigente.

Emito mi Voto Aprobatorio.

Atentamente,

"EL INGENIO PARA CREAR, NO PARA DESTRUIR"


M.D.M. Antonio A. Ordaz Hernández
Sinodal

c.c Archivo
*GHR/sar

RESUMEN

En el presente trabajo se realizó la implantación práctica de un algoritmo de control adaptable en un microcontrolador de propósito general, para controlar en tiempo real una planta variante en el tiempo, continua, una entrada-una salida. El controlador implementado posee las siguientes características: utiliza un algoritmo de identificación recursiva mínimos cuadrados con factor de olvido, se pueden especificar los objetivos de regulación y seguimiento de manera independiente. Para llevar a cabo la implementación tanto del sistema adaptable como del protocolo de comunicación a través de un lenguaje de alto nivel se diseñó una tarjeta de control y adquisición de datos USB, dicha tarjeta trabaja como instrumento necesario para llevar a cabo las pruebas y dar la pauta para poder realizar sistemas embebidos con microcontroladores. Otro de los aspectos importantes fue la implementación de un sistema de muestreo sincronizado y configurable para llevar todo el proceso de adquisición de datos, desarrollo de ecuaciones, operaciones de punto flotante y transferencia de información, de tal manera que se obtuviera un control absoluto para poder obtener la señal de control y la respuesta de la planta. Para la implementación del algoritmo de control adaptable se utilizaron herramientas de diseño bastante robustas tanto en hardware como en software y de última tecnología en sistemas de comunicación que hicieron que la experimentación, las pruebas y la implementación fueran un éxito, demostrando así que una metodología e investigación enfocada y concisa dan resultados adecuados y precisos, como prueba de ello fueron los obtenidos en la implantación del algoritmo adaptable en el microcontrolador comparados con los simulados, ambos con un nivel alto de exactitud.

Palabras clave: Control adaptable, identificación recursiva, tiempo real y microcontroladores.

DEDICATORIAS

A mi Madre:

Con respeto y admiración a Patricia; por su cariño, sacrificio, esfuerzo, enseñanzas, orientación y ejemplo. Porque ha creado en mí sabiduría, haciendo que hoy tenga el conocimiento de lo que soy, por sus cuidados, por creer en mí y por haberme enseñado que por largo que parezca el camino siempre se puede llegar.

No pudiendo expresar todo lo que siento por ella, solo digo GRACIAS, éste trabajo también es suyo.

A mis hermanas:

Agradezco a mis hermanas el apoyo que siempre me han brindado con su impulso, fuerza y tenacidad que son parte de mi formación, gracias por estar ahí para ayudarme. A ti Alma, por tus ganas de salir adelante y por tu fuerza de voluntad.

Como muestra de gratitud les dedico éste trabajo.

ÍNDICE GENERAL

Resumen.....	i
Dedicatorias.....	ii
Índice general.....	iii
Índice de figuras.....	v
Índice de gráficas.....	vii
I. Introducción.....	1
1.1. Planteamiento del problema.....	2
1.2. Objetivos del estudio.....	4
1.3. Metodología.....	5
1.4. Organización de la tesis.....	7
II. Conceptos Básicos.....	8
2.1. Introducción.....	9
2.2. La idea básica de un sistema de control.....	10
2.3. La computadora como un elemento de control.....	14
2.4. El sistema de control digital de lazo cerrado.....	16
2.5. ¿porque el control digital en lugar del analógico?.....	17
III. Diseño del algoritmo de control.....	18
3.1 Control de seguimiento y regulación.....	19
3.2 Identificación recursiva de mínimos cuadrados.....	25
3.3 Propiedades del algoritmo de identificación.....	28
3.4 Control adaptable directo de sistemas de fase mínima.....	29
3.5 Control adaptable con factor de olvido.....	31

ÍNDICE GENERAL (Continuación)

IV.	Topología del sistema de control.....	33
4.1	Introducción.....	34
4.2	Tarjeta de control y adquisición de datos USB.....	36
4.2.1	Microcontrolador PIC18F4550.....	38
4.2.2	Protocolo de comunicación USB.....	41
4.2.3	Programador Gtp Usb Lite.....	48
4.3	Software de soporte.....	50
4.3.1	Generación y grabación del firmware.....	53
4.3.2	Simulación.....	58
4.3.3	Interfaz grafica.....	60
V.	Desempeño del algoritmo de control.....	71
5.1	Modelo de la planta.....	72
5.2	Aplicación del controlador lineal de seguimiento y regulación.....	74
5.3	Aplicación del controlador adaptable.....	83
VI.	Conclusiones.....	96
VII.	Bibliografía.....	99
VIII.	Apéndice.....	100
	Apéndice A. Programa del controlador lineal de seguimiento y regulación.....	101
	Apéndice B. Desempeño del control ante una dinámica variante de la planta.....	103
	Apéndice C. Aplicación del controlador adaptable.....	105
	Apéndice D. Desempeño del control adaptable ante una dinámica variante de la planta.....	108

ÍNDICE DE FIGURAS

2.1.	Planta controlada sin retroalimentación.....	10
2.2.	Sistema multivariable controlado por retroalimentación.....	11
2.3.	Sistema multivariable controlado por retroalimentación con sensores...	12
2.4.	Servomecanismo de posición con señales continuas.....	13
2.5.	Sistema posicionador controlado digitalmente.....	14
2.6.	Sistema multivariable controlado digitalmente.....	15
2.7.	Varias configuraciones de un sistema de control digital.....	16
3.1.	Control de seguimiento y regulación.....	24
3.2.	Esquema de control con identificación.....	29
3.3.	Control adaptable directo.....	32
4.1.	Topología del sistema de control en hardware.....	34
4.2.	Topología del sistema de control en software.....	35
4.3.	Esquemático de la tarjeta de adquisición de datos USB.....	37
4.4.	Tarjeta de control y adquisición de datos USB.....	37
4.5.	PIC18F4550 - empaquetado DIP-40.....	38
4.6.	Diagrama de bloques del PIC18F4550.....	38
4.7.	Pinout del PIC18F4550.....	40
4.8.	Características generales del PIC18F4550.....	40
4.9.	Clases de dispositivo USB.....	46
4.10.	Características del cable USB.....	47
4.11.	Conectores USB.....	47
4.12.	Esquemático del programador GTP USB LITE.....	49
4.13.	Programador GTP USB LITE terminado.....	49
4.14.	Asistente para la instalación de los drivers.....	51
4.15.	Instalando los drivers.....	52
4.16.	Tarjeta de control y adquisición de datos Instalada.....	52
4.17.	PCW C Compiler.....	53
4.18.	WinPic800.....	56
4.19.	PICDEM.....	57
4.20.	Interface Visual C++ 6.0.....	58
4.21.	Interfase Matlab.....	59

ÍNDICE DE FIGURAS (Continuación)

4.22.	Interfaz grafica LabVIEW.....	60
5.1.	Sistema de temperatura.....	72
5.2.	Aplicación del control de seguimiento y regulación.....	75
5.3.	Cambiando los parámetros de la planta.....	79
5.4.	Cambiando los parámetros de la planta en el control adaptable directo.....	88

ÍNDICE DE GRÁFICAS

5.1.	Salida del modelo de referencia para la planta ec. 5.2.7 con $U^M(k)=2$	76
5.2.	Respuesta de la planta ec. 5.2.1.....	77
5.3.	Señal de control ec. 5.2.5.....	77
5.4.	Respuesta de la planta implementada en tiempo real.....	78
5.5.	Señal de control implementado en tiempo real.....	78
5.6.	Dinámica difícil de la planta ec. 5.2.1 con $k_p=2.05$	80
5.7.	Señal del control ec. 5.2.5 a la dinámica difícil de la planta con $k_p=2.05$	80
5.8.	Dinámica difícil de la planta ec. 5.2.1 con $k_p=2.05$ en tiempo real.....	81
5.9.	Señal de control ec. 5.2.5 a la dinámica difícil de la planta con $k_p=2.05$ en tiempo real.....	81
5.10.	Dinámica difícil de la planta ec. 5.2.1 con $k_p=0.2$ en tiempo real.....	82
5.11.	Señal de control ec. 5.2.5 a la dinámica difícil de la planta con $k_p=0.2$ en tiempo real.....	82
5.12.	Respuesta de la planta ec. 5.2.1 con controlador adaptable.....	85
5.13.	Señal de control adaptable ec. 5.3.1.....	85
5.14.	Valores estimados de $\hat{\theta}$	86
5.15.	Respuesta de la planta ec. 5.2.1 en tiempo real.....	87
5.16.	Señal de control adaptable ec. 5.3.1 en tiempo real.....	87
5.17.	Respuesta de la planta con dinámica difícil con $k_p=2.05$	89
5.18.	Señal de control a la dinámica difícil de la planta con $k_p=2.05$	89
5.19.	Valores estimados de $\hat{\theta}$ a la dinámica difícil de la planta con $k_p=2.05$	90
5.20.	Respuesta de la planta con dinámica difícil con $k_p=2.05$ en tiempo real.....	90
5.21.	Señal de control a la dinámica difícil de la planta con $k_p=2.05$ en tiempo real	91
5.22.	Respuesta del la planta con dinámica difícil con $k_p=0.2$	92
5.23.	Señal de control a la dinámica difícil de la planta con $k_p=0.2$	92

ÍNDICE DE GRÁFICAS (Continuación)

vii

5.24. Respuesta de la planta con dinámica difícil con $k_p=0.2$ en tiempo real.....	93
5.25. Señal de control a la dinámica difícil de la planta con $k_p=0.2$ en tiempo real.....	93
5.26. Valores estimados de $\hat{\theta}$ a la dinámica difícil de la planta con $k_p=0.2$	94

CAPÍTULO I
INTRODUCCIÓN

INTRODUCCIÓN

1.1 Planteamiento del problema

Los algoritmos de control convencional como el PID han demostrado un adecuado desempeño cuando se conoce el modelo de la planta; sin embargo, si la planta presenta una dinámica difícil (variante en el tiempo, no lineal, retardos de tiempo grandes, etc.) el desempeño del sistema de control se degrada fuertemente. Por esta razón es necesario utilizar algoritmos de control adaptable los cuales son capaces de adaptarse a las variaciones de la planta.

El principal problema a resolver es el de mostrar el diseño e implementación de un algoritmo de control adaptable de seguimiento y regulación para una planta variante en el tiempo, continua, una entrada-una salida a través de un microcontrolador en tiempo real. A partir de ello se derivan varios puntos a desarrollar antes de conseguir resolver el problema principal:

- a) Obtener un sistema de muestreo exacto para la adquisición de datos y procesamiento de la información.
- b) Implementar el protocolo de comunicación USB dentro del microcontrolador.
- c) Desarrollar el programa de adquisición de información en lenguaje G para obtener los resultados gráficos tanto de la planta como del control.
- d) Implementar la planta dentro del microcontrolador de acuerdo a su definición matemática con el objetivo de analizar la respuesta y ver como actúan en tiempo real.
- e) Implementar diferentes tipos de controladores discretos.
- f) Integrar un sistema de control de seguimiento y regulación lineal a través de un modelo de referencia preestablecido.

Si bien se tienen que desarrollar varios procesos antes de lograr la implementación del algoritmo adaptable, hay otros aspectos a considerar que son también de relevancia como por ejemplo, elaborar un programador óptimo para programar el tipo de microcontrolador con interfase USB, implementar las ecuaciones de diferencias resultantes tanto de la planta como del controlador en el lenguaje del microcontrolador. Disponer de un compilador adecuado al tipo de microcontrolador y al lenguaje en el cual se está programando, configurar el protocolo de adquisición de datos USB a las entradas y salidas tanto analógicas como digitales necesarias para el sistema de control y uno de los aspectos importantes es sincronizar el reloj de todos los procesos.

1.2 Objetivos del estudio

Como sabemos los sistemas de control para plantas de cualquier orden lineales o no lineales implementadas a nivel simulación en plataformas como Matlab son fáciles de analizar y manipular de manera interactiva, pero cuando se desea llevar a la práctica dichos sistemas de control nos encontramos con varios problemas como pueden ser sincronización, ruido, adquisición de datos, resolución, sistemas de muestreo, etc., es por ello que se han planteado los siguientes objetivos para la realización de este trabajo y que construyen la base suficiente para lograrlo y tratar de evitar los problemas anteriores.

- a) Crear nuevos enfoques para la construcción e implementación de controladores adaptables en microcontroladores de propósito general.
- b) El diseño, implementación y análisis de un sistema con el fin de comprobar su fiabilidad y efectividad.
- c) Investigar y adaptar las herramientas con las que se puede implementar el algoritmo adaptable, es decir, los requerimientos en Hardware y Software necesarios para llevarlo a cabo.
- d) Definir claramente cual va a ser el modelo de la planta y con que periodo de muestreo se van a adquirir los datos.
- e) El lenguaje de programación con el que se va a desarrollar la aplicación e interfase con la computadora para la adquisición los datos y posteriormente poder graficarlos.
- f) El tipo de microcontrolador a usar para poder implementar los algoritmos de control, así como su lenguaje de programación y el tipo de programador necesario.
- g) El protocolo de comunicación que se utilizara para poder transmitir la información a la computadora.

1.3 Metodología

Para la realización correcta de este trabajo de investigación y desarrollo propuesto se tuvo que tener primeramente muy claro que es lo que se desea investigar y desarrollar, así mismo poseer bases muy sólidas en áreas de electrónica, control y programación. Ahora se describirá de manera general la metodología empleada a través de las áreas de conocimiento anteriormente mencionadas:

a) Comencemos primero con el área de electrónica, aquí se deben considerar 3 aspectos, el primero de ellos fue investigar el tipo de microcontrolador que se empleara para poder implementar el algoritmo de control, esta investigación se realizó en base a la memoria requerida, velocidad, interfase de comunicación USB, que posea convertidores A/D, de bajo costo y fácil de adquirir; el segundo aspecto es el programador con el que se va a poder grabar el firmware del control, este debe ser económico, fácil de construir y rápido de programar; el tercer y último aspecto a considerar fue la investigación de los requerimientos técnicos y componentes necesarios para poder implementar el protocolo de comunicación USB, así como también todo lo necesario para poder configurar el microcontrolador en la parte de hardware.

b) La investigación en el área de programación fue dividida en 2 rubros:
1) *La programación en la computadora:* en esta parte se investigo en que lenguaje de programación se iba a simular el sistema de control así como también en que lenguaje se iba a implementar la interfase gráfica con el microcontrolador, las consideraciones tomadas se basan en que el lenguaje para la interfaz gráfica fuera de alto nivel.

Donde la funcionalidad de la comunicación del protocolo USB se maneja independientemente, es decir, que la aplicación principal no dependiera en tiempo y respuesta del protocolo mismo, si no al contrario que lo pudiera utilizar en cualquier instante como una función o bloque encapsulado.

Ahora bien para poder simular el sistema de control y después portarlo al microcontrolador se utilizo el lenguaje C y el lenguaje propietario de Matlab, el primero utilizado para obtener los valores estimados del control adaptable y el segundo para poder graficarlos.

II) *La programación en el microcontrolador.* Para poder obtener el firmware necesario del control adaptable y poderlo grabar dentro de la memoria del microcontrolador se necesita un compilador que interprete instrucciones de un lenguaje de programación de alto nivel a instrucciones de bajo nivel. Para esto existen muchos compiladores para microcontroladores que hacen eso, por ejemplo compiladores de C, Basic, Pascal, Niple, etc., ahora las consideraciones que se tomarán para escoger a uno estarán en base a lo siguiente: que el compilador incluya al microcontrolador con interfase USB, que posea librerías de comunicación, que optimice de mejor manera la memoria RAM del microcontrolador y que sea económico de adquirir.

c) En el área de Control se investigo de manera concisa los siguientes temas: el control de seguimiento y regulación, la Identificación recursiva de mínimos cuadrados, las propiedades del algoritmo de identificación, el control adaptable directo de sistemas de fase mínima y el control adaptable con factor de olvido; si bien para llevar a buen puerto este estudio se deben tener buenas bases teóricas de sistemas de control en tiempo discreto, ya que a partir de ello el algoritmo de control adaptable resultante se tendrá que implementar de acuerdo a las propiedades del microcontrolador y a su lenguaje de programación, es por eso que no solo basta obtener matemáticamente dicho algoritmo si no también pensar en la manera de implementarlo.

1.4 Organización de la tesis

Este documento se encuentra dividido en dos partes. La primera es dedicada a la teoría que abarca los requerimientos necesarios para poder implementar con éxito el algoritmo de control adaptable y la segunda está enfocada a la obtención de los datos experimentales.

- En el primer capítulo denominado introducción se presentan los objetivos, el planteamiento del problema y la metodología que son los pasos a seguir para sistematizar la investigación.
- En el segundo capítulo se describe en forma de conceptos básicos la necesidad de la utilización de los sistemas de control digitales frente a los analógicos y la forma en que estos están evolucionado.
- En el tercer capítulo se aborda claramente el estudio del diseño del algoritmo de control adaptable, que es la base teórica de este trabajo de tesis y que a través de su modelo matemático se logrará la implementación final en el microcontrolador.
- En el cuarto capítulo denominado topología del sistema de control esta dividido en dos rubros; el primero de ellos a lo referente a todo el hardware utilizado para la implementación del algoritmo de control adaptable y el segundo rubro a todo el software que permite la simulación, diseño e implantación del mismo.
- En el quinto capítulo se hace un análisis del desempeño del algoritmo de control adaptable que abarca la parte de experimentación, las gráficas del comportamiento real y simulado, que son el medio para interpretar los resultados y poder demostrar la funcionalidad del mismo a través de diferentes análisis que se desarrollaron durante el trabajo.

CAPÍTULO II

CONCEPTOS BÁSICOS

CONCEPTOS BÁSICOS

2.1 Introducción

La Teoría de Control se fundamenta en la búsqueda de mecanismos que permitan interactuar sobre los sistemas con el fin de que ciertas variables observadas de éste se comporten según unas pautas prefijadas. Este objetivo se pretende lograr sobre cualquier sistema, independientemente de su complejidad. Es por ello, que si bien durante el presente siglo se ha establecido una teoría de control muy sólida que permite lograr dicho fin sobre sistemas cuyo comportamiento puede asimilarse a un modelo dinámico lineal, sin embargo todavía no es posible hablar de una solución que permita lograr el control de forma general sobre cualquier sistema.

Las técnicas de control no lineal se han ido extendiendo poco a poco con el fin de contemplar todas las no linealidades que se hallan presentes en dichos sistemas y que lo alejan del modelo lineal de mayor sencillez. Sin embargo, estas herramientas analíticas en muchos casos han sido superadas por otras técnicas que tratan de obtener el control sobre el sistema a través de funciones que de algún modo se inspiran o tratan de emular el intelecto humano. Son estas últimas las que se han englobado bajo el nombre de *Control Adaptable*, considerándose éste como un campo de investigación, que por necesidad se debe hallar ligado a otros enfoques clásicos de la teoría de control.

Como suele ocurrir, tanto el control clásico, como el control Adaptable ambicionan mucho más de lo que hoy en día puede obtenerse de forma práctica de ellos. Sin embargo, del esfuerzo desarrollado en ambos campos, se ha podido obtener una serie de procedimientos que han encontrado gran aplicación, y que en muchos casos dan una solución práctica a las necesidades actuales de sistemas de control.

2.2 La idea básica de un sistema de control

Considérese una planta dinámica la cual va a ser controlada para hacer una tarea dada. Generalmente esta planta tiene entradas y salidas continuas en el tiempo. En general la planta tiene r variables de entrada (variables manipulables) y m variables de salida. Podría haber también variables internas las cuales no son salidas.

Decidir cuales variables son salidas, es materia del diseñador de sistemas. El problema de control es el de manipular las variables de entrada en un intento para influenciar las variables de salida en una manera deseada, por ejemplo para alcanzar ciertos valores o ciertos rangos.

Si nosotros conocemos el modelo del sistema y las condiciones iniciales con razonable precisión, podemos manipular las variables de entrada tal que conduzcamos las salidas en la manera deseada. Esto es lo que llamamos "Control de Lazo Abierto" ya que este es conseguido sin el conocimiento de las salidas en ese momento.

Esta situación es mostrada en la figura 2.1 este tipo de control también asume que el sistema opera en ausencia de perturbaciones, lo que causaría que las salidas del sistema variaran con respecto a aquellas predichas por un modelo de la planta.

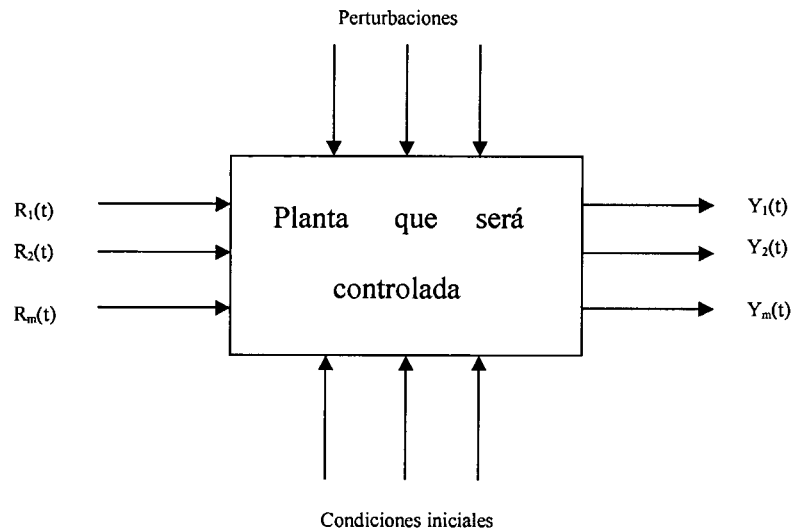


Figura 2.1

Planta controlada sin retroalimentación

Debido a las incertidumbres en el modelo del sistema y condiciones iniciales, así como a las perturbaciones hay una mejor manera de llevar a cabo la tarea de control. Esta técnica involucra medir el funcionamiento de algunos subconjuntos de las variables de salida (aquellas que queremos controlar) $y_1(t) \dots y_k(t)$ y, después de la medición, la comparación con lo que nos gustaría que fuera cada una de ellas al tiempo t y llamando la diferencia entre el valor deseado $r_1(t)$ y el valor actual $y_1(t)$ el "error".

Usando los errores en cada una de las variables, podemos generar las variables de control o esfuerzos tal que lleven los errores a cero. Esta situación esta representada en la figura 2.2 y conocida como "Control de Lazo Cerrado". Debido a la fricción, inercia y otras propiedades dinámicas es posible llevar un sistema instantáneamente a error cero. Debido a esto, las salidas del sistema $y_1(t)$ se retrazaran a la entrada deseada $r_1(t)$ y algunas veces sobrepasarán u oscilaran alrededor de una condición de error cero.

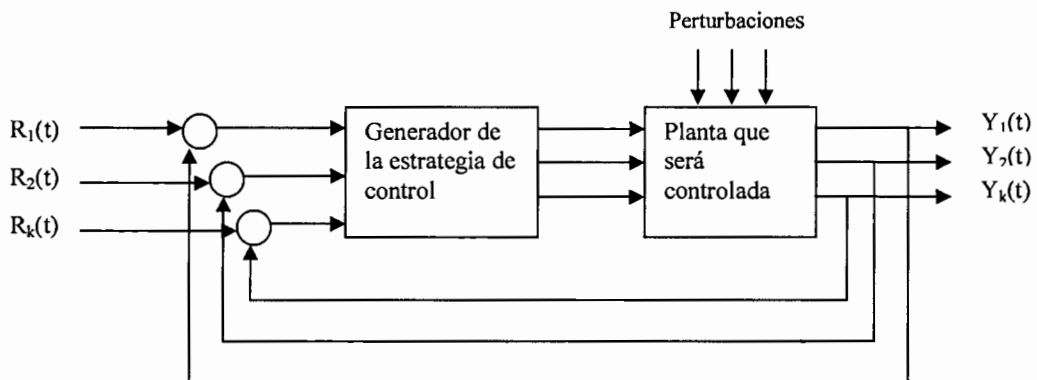


Figura 2.2
Sistema multivariable controlado por retroalimentación

Frecuentemente, no podemos tener físicamente disponibles las variables de salida ni puede ser conveniente comparar físicamente. Por ejemplo, si una de estas fuera la temperatura en el centro de un reactor nuclear, no sería físicamente accesible ni conveniente comparar temperaturas.

Sería más conveniente medir la salida con un transductor o sensor el cual genere una señal de referencia para generar el esfuerzo de control. Esto es frecuentemente, aunque no siempre hecho electrónicamente.

Otra vez, debido a las características dinámicas de los sensores sus salidas no siempre son proporcionales exactamente a las mediciones para las cuales fueron diseñados a medir, y de aquí que la señal retroalimentada no siempre es una replica verdadera de la variable que esta siendo medida. Esperamos, al principio que la información deseada para alcanzar el control esté presente en el sensor de salida o la situación se convierte en un control de "Lazo Abierto". El sistema controlado por retroalimentación con sensores esta mostrado en la figura 2.3 con las señales del sensor de salida denotados por $w_1(t)$, $w_2(t)$... $w_k(t)$.

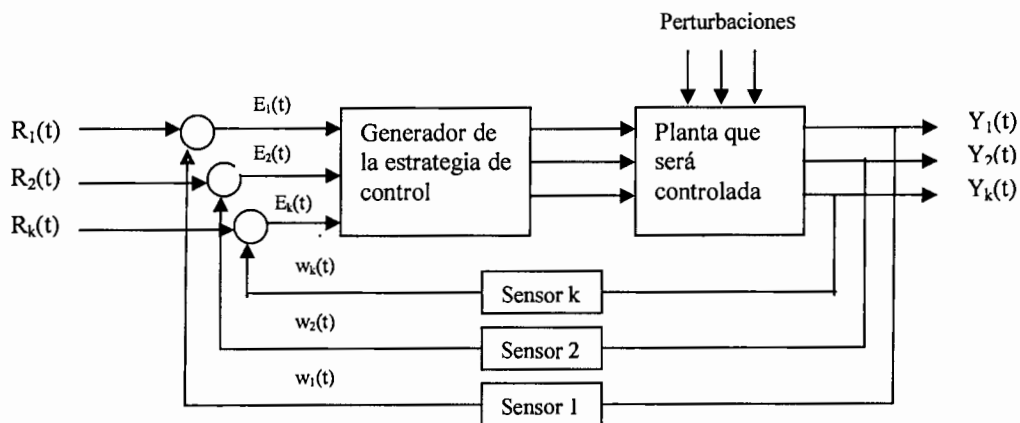


Figura 2.3

Sistema multivariable controlado por retroalimentación con sensores

Por muchos años los sensores y la síntesis del generador de señal de control han sido el centro de considerable actividad ingenieril, y en general estos dispositivos son de naturaleza eléctrica y electrónica, más específicamente filtros, amplificadores, amplificadores de potencia, actuadores, etc. En todos estos casos, las señales han sido funciones continuas de la variable del tiempo t .

Nos referimos comúnmente a tales sistemas como sistemas de control de tiempo continuo. A medida que los sistemas se han vuelto más y más complejos con una multitud de variables de control y variables de salida, el trabajo de implementación se vuelve una tarea difícil.

Con el advenimiento de la computadora digital, los ingenieros empezaron a explorar las posibilidades de mantener la computadora siguiendo las varias señales y hacer decisiones lógicas respecto a las señales de control basada en las señales medidas y los valores deseados de salida.

Sabemos no obstante, que una computadora digital es capaz solamente de operar con números y no con señales, así si una computadora digital es usada para realizar una tarea de control, las señales del sensor deben ser convertidas a números mientras que las salidas de las decisiones del control deben ser salidas en la forma de esfuerzos de control de tiempo continuo.

Como un ejemplo consideremos un servomecanismo de posición de lazo cerrado en la forma de tiempo continuo como el mostrado en la figura 2.4. La señal de referencia es en la forma de un voltaje, como es la señal de retroalimentación, ambas generadas por los potenciómetros gobernados mecánicamente.

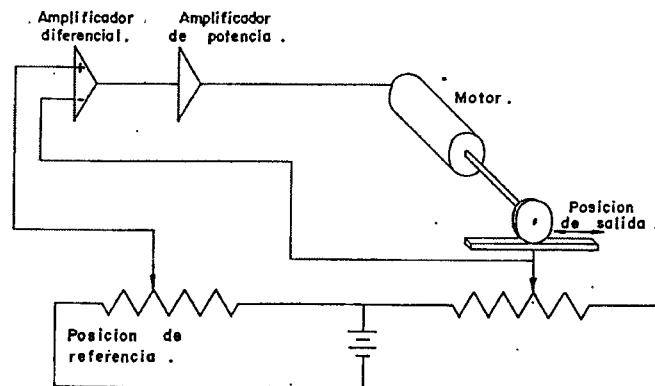


Figura 2.4

Servomecanismo de posición con señales continuas

2.3 La computadora como un elemento de control

Investiguemos ahora como esta tarea relativamente simple, mencionada en el párrafo anterior, podrá ser llevada a acabo mediante una computadora digital para generar la señal al amplificador de potencia. Debemos postular primero la existencia de dos dispositivos, el primero de estos es el convertidor analógico a digital (A/D) el cual muestreara la señal de salida periódicamente y convertirá la muestra en una palabra digital para ser procesada por la computadora digital y en consecuencia generar una estrategia de control en forma de un número. El segundo dispositivo es un convertidor digital a analógico (D/A) el cual convierte la estrategia de control numérico generada por la computadora digital de una palabra digital a una señal analógica.

El servomecanismo de posición esta mostrado en la figura 2.5 controlado por una computadora digital.

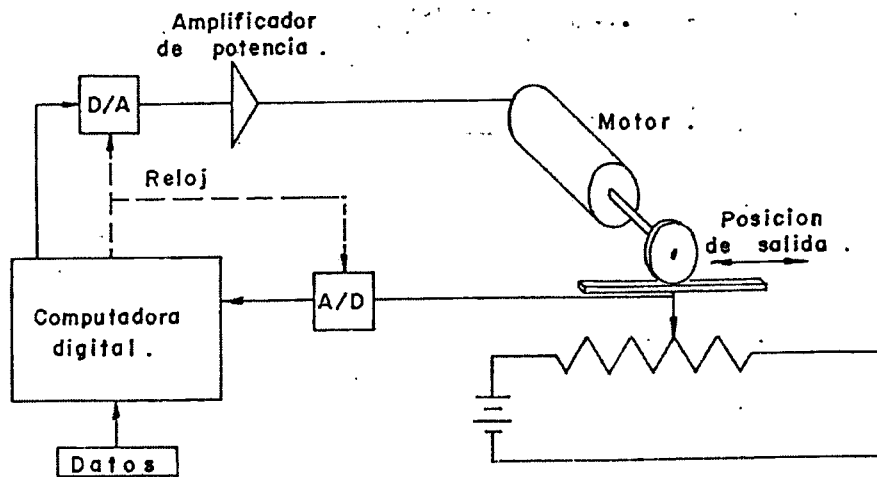


Figura 2.5

Sistema posicionador controlado digitalmente

Este sistema no fue escogido porque fuera real sino más bien para mostrar el principio del control digital de un sistema de lazo simple.

Generalmente los convertidores A/D y D/A operan periódicamente y de aquí entre mas cerca sean tomadas las muestras y más frecuente las salidas del convertidor D/A sean actualizadas, el sistema de control digital se aproximara mas a un sistemas de tiempo continuo.

Como sabremos mas tarde, no siempre es deseable tener el sistema aproximado al sistema continuo donde queremos que haya atributos de un sistema de tiempo discreto. La limitación a la cual el muestreo puede ser hecho es que la señal muestreada debe ser procesada por la computadora antes que esta maneje el convertidor D/A con la nueva estrategia de control.

Cualquier algoritmo para proporcionar la señal de control toma tiempo para ejecutar y debido a esto limita el rango al cual la actualización del esfuerzo de control puede ocurrir. El sistema general multivariable controlado por computadora está mostrado en la figura 2.6.

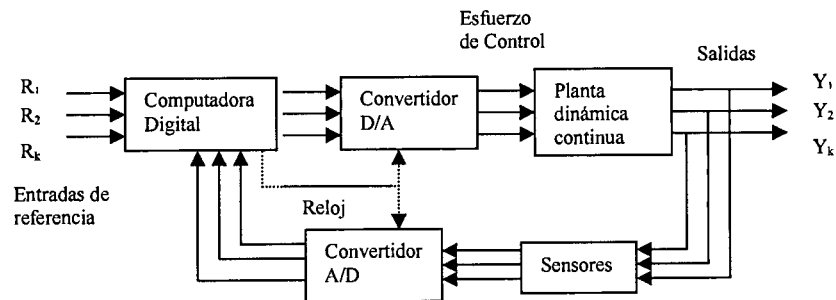


Figura 2.6

Sistema multivariable controlado digitalmente

2.4 El sistema de control digital de lazo cerrado

Frecuentemente el problema es el control de una sola variable de un sistema que puede tener una multitud de otras variables que no necesariamente deben ser controladas. Hay varias configuraciones de un sistema de control de lazo cerrado simple, dos de las cuales son mostradas en la figuras 2.7 a y b. En ambos casos una simple variable de tiempo continuo $y(t)$ esta siendo controlada para seguir una señal de referencia que podría ser cero o constante como en el caso de un regulador.

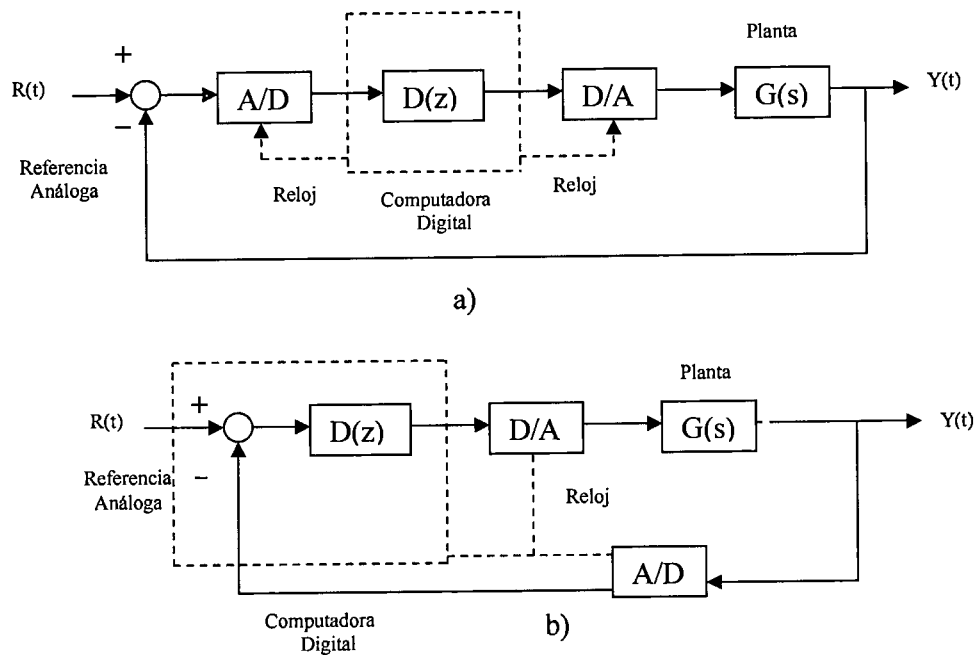


Figura 2.7

Varias configuraciones de un sistema de control digital

La señal que sale de la computadora digital es en ambos casos una secuencia periódica de números que representan la estrategia de control que fue generada por la computadora. La entrada a la computadora digital es una secuencia periódica de números que representan las muestras periódicas de la señal continua que entra al convertidor A/D. El propósito de desarrollar la teoría de control digital es encontrar algoritmos deseables por los cuales la computadora digital convierte la secuencia de entrada en la secuencia de salida que es la estrategia de control numérico.

2.5 ¿Porque el control digital en lugar del analógico?

Ya que casi todas las funciones de control pueden ser alcanzadas por implementación de dispositivos análogos, uno se podría preguntar ¿Porque estudiar teoría de control digital? Los ingenieros han estado interesados por mucho tiempo en la posibilidad de incorporar computadoras digitales a los lazos de control debido a la habilidad de una computadora digital para procesar inmensas cantidades de datos, de esta manera anteriormente encontramos que fueron inicialmente muy grandes y complejos dispositivos que generalmente costaron mucho para la aplicación en sistemas de control de moderado grado de dificultad. Con la aparición de la microcomputadora a mediados de los 60's las posibilidades para el control digital fueron grandemente expandidas debido a la reducción de tamaño y costo. Esto permitió la aplicación de computadoras para controlar sistemas más pequeños y menos costosos. El advenimiento del microprocesador unitario a principios de los 70's ha expandido similarmente los horizontes ya que las capacidades que anteriormente costaron miles de dólares pueden ser compradas, la mayoría por unos cientos de dólares. Estos precios hacen la implementación de control digital competitiva contra la de control análogo, aun para las más simples aplicaciones de control de lazo simple. Ahora vemos que las unidades de microprocesador completas con controlador, unidad aritmética, reloj, ROM y RAM, en un solo circuito integrado están disponibles por menos de cinco dólares.

CAPÍTULO III

DISEÑO DEL ALGORITMO DE CONTROL

DISEÑO DEL ALGORITMO DE CONTROL

3.1 Control de seguimiento y regulación

El propósito de este capítulo es el de mostrar el diseño de un algoritmo de control del tipo de modelo lineal de seguimiento y regulación para una planta invariante en el tiempo, lineal, discreta, una entrada-una salida, descrita por:

$$A(q^{-1})Y(k) = q^{-d}B(q^{-1})U(k) \quad \text{Para } d>0 \quad (3.1.1)$$

Donde:

$$A(q^{-1}) = 1 + a_1q^{-1} + a_2q^{-2} + \dots + a_{nA}q^{-nA} \quad (3.1.2)$$

$$B(q^{-1}) = b_0 + b_1q^{-1} + b_2q^{-2} + \dots + b_{nB}q^{-nB} \quad ; \quad b_0 \neq 0 \quad (3.1.3)$$

$\{q^{-1}\}$ Es el operador de atraso, d representa el retardo de tiempo de la planta. $U(k)$ y $Y(k)$ son la entrada y salida de la planta respectivamente.

Asumimos que los ceros de $B(q^{-1})$ están todos en $|q| < 1$, de ahí que puedan ser cancelados sin conducir a una entrada de control sin límites.

Los objetivos del control que estamos considerando son los siguientes:

(1) El control debe ser tal que en seguimiento, la salida del proceso satisfaga la ecuación.

$$C_r(q^{-1})Y^M(k) = q^{-1}D(q^{-1})U^M(k) \quad (3.1.4)$$

Donde $C_r(q^{-1})$, $D(q^{-1})$ son polinomios en q^{-1} , $C_r(q^{-1})$ es asintóticamente estable y $U^M(k)$ es una secuencia de referencia limitada.

(2) El control deberá ser tal que en regulación ($U^M(k) = 0$), una perturbación inicial ($Y(0) \neq 0$) es eliminada con las dinámicas definidas por:

$$C_r(q^{-1})Y(k+d) = 0 \quad k \geq 0 \quad (3.1.5)$$

Donde

$$C_r(q^{-1}) = 1 + C_1 q^{-1} + \dots + C_{n_{c_2}} q^{-n_{c_2}} \quad (3.1.6)$$

Es un polinomio asintóticamente estable.

Una solución se obtiene usando un modelo de referencia explícito en la ec. 3.1.4 dado por:

$$C_r(q^{-1})Y^M(k) = q^{-1}D(q^{-1})U^M(k)$$

$Y^M(k)$ y $U^M(k)$ son la salida y la entrada del modelo; ambos son limitados.

El error planta-modelo es definido como:

$$\varepsilon(k) = Y(k) - Y^M(k) \quad (3.1.7)$$

Es claro que los objetivos del control son ejecutados si se mantiene la siguiente igualdad:

$$C_r(q^{-1})\varepsilon(k+d) = 0 \quad k > 0 \quad (3.1.8)$$

Usando la identidad:

$$C_r(q^{-1}) = A(q^{-1})S(q^{-1}) + q^{-1}R(q^{-1}) \quad (3.1.9)$$

Donde

$$S(q^{-1}) = 1 + s_1 q^{-1} + s_2 q^{-2} + \dots + s_{n_s} q^{-n_s} \quad (3.1.10)$$

$$R(q^{-1}) = r_0 + r_1 q^{-1} + r_2 q^{-2} + \dots + r_{n_R} q^{-n_R} \quad (3.1.11)$$

Las cuales tienen una única solución

$$S(q^{-1}), R(q^{-1}) \quad n_s = d - 1 \quad y \quad n_R = \max(n_A - 1, n_{C_2} - d)$$

La ecuación 3.1.9 puede ser escrita:

$$C_r(q^{-1})\varepsilon(k+d) = B(q^{-1})S(q^{-1})U(k) + R(q^{-1})Y(k) - C_r(q^{-1})Y^M(k+d) \quad (3.1.12)$$

Ó en forma vectorial:

$$C_r(q^{-1})\varepsilon(k+d) = b_0 U(k) + \theta_0^T \phi_0(k) - C_r(q^{-1})Y^M(k+d) \quad (3.1.13)$$

Ó

$$\varepsilon = \theta^T \phi(k) - C_r(q^{-1})Y^M(k+d) \quad (3.1.14)$$

Donde

$$\phi_0^T(k) = [U(k-1), \dots, U(k-d-n_B+1), Y(k), \dots, Y(k-n_R)] \quad (3.1.15)$$

$$\theta_0^T(k) = [b_0, s_1 + b_1, b_0 s_2 + b_1 s_1 + b_2, b_0 s_3 + b_1 s_2 + b_2 s_1 + b_3, \dots, b_{n_B} s_{d-1}, r_0, \dots, r_{n_R}] \quad (3.1.16)$$

$$\phi^T(k) = [U(k); \phi_0^T(k)] \quad (3.1.17)$$

$$\theta^T(k) = [b_0; \theta_0^T] \quad (3.1.18)$$

Se puede probar que la identidad:

$$C_r(q^{-1}) = A(q^{-1})S(q^{-1}) + q^{-1}R(q^{-1}) \quad (3.1.19)$$

Tiene una solución única. Escribámosla como un conjunto de ecuaciones correspondientes a las potencias de q^{-1}

$$\begin{aligned}
 1 &= 1 \\
 C_1^2 &= a_1 + s_1 \\
 C_2^2 &= a_2 + a_1 s_1 + s_2 \\
 &\cdot \\
 &\cdot \\
 &\cdot \\
 C_{d-1}^2 &= a_{d-1} + \dots + s_{d-1} \\
 C_d^2 &= a_d + \dots + s_d + r_0 \\
 &\cdot \\
 &\cdot \\
 &\cdot
 \end{aligned}$$

El número de coeficientes a calcular es $n_S + n_R + 1$ y el número de ecuaciones del conjunto es $\max(n_A + n_S, n_R + d)$.

Es sabido que para resolver un sistema de ecuaciones de este tipo, el número de coeficientes debe ser igual al número de ecuaciones, es decir:

$$n_S + n_R + 1 = \max(n_A + n_S, n_R + d) \geq n_{C_2} \quad (\text{A})$$

Se puede ver, del conjunto de ecuaciones anteriores, que n_S debe satisfacer:

$$n_S \geq d - 1$$

Tomando el valor mínimo

$$n_S = d - 1$$

Si igualamos el segundo miembro de las ecuaciones 3.1.12, 3.1.13 ó 3.1.14 a cero se alcanzará el objetivo de control.

$$U(k) = \frac{1}{b_0} [C_r(q^{-1})Y^M(k+d) - \theta_0^T \phi_0(k)] \quad (3.1.20)$$

$$= \frac{1}{b_0} [C_r(q^{-1})Y^M(k+d) - R(q^{-1})Y(k) - Bs(q^{-1})U(k)] \quad (3.1.21)$$

Donde

$$Bs(q^{-1}) = B(q^{-1})S(q^{-1}) - b_0 \quad (3.1.22)$$

Debe notarse que con el uso de este algoritmo puede especificarse por separado los objetivos de seguimiento y regulación ya que puede escogerse de manera diferente el polinomio C_r . Se puede ver en estudios más extensos como C_r juega además un papel de filtro y suaviza el proceso de adaptación. El esquema general de este algoritmo se puede apreciar en la figura 3.1.

El uso de este algoritmo esta limitado a sistemas cuyos parámetros no cambian en el tiempo, para los sistemas con parámetros cambiantes se puede diseñar algoritmos de control del tipo de adaptación como veremos en los siguientes apartados.

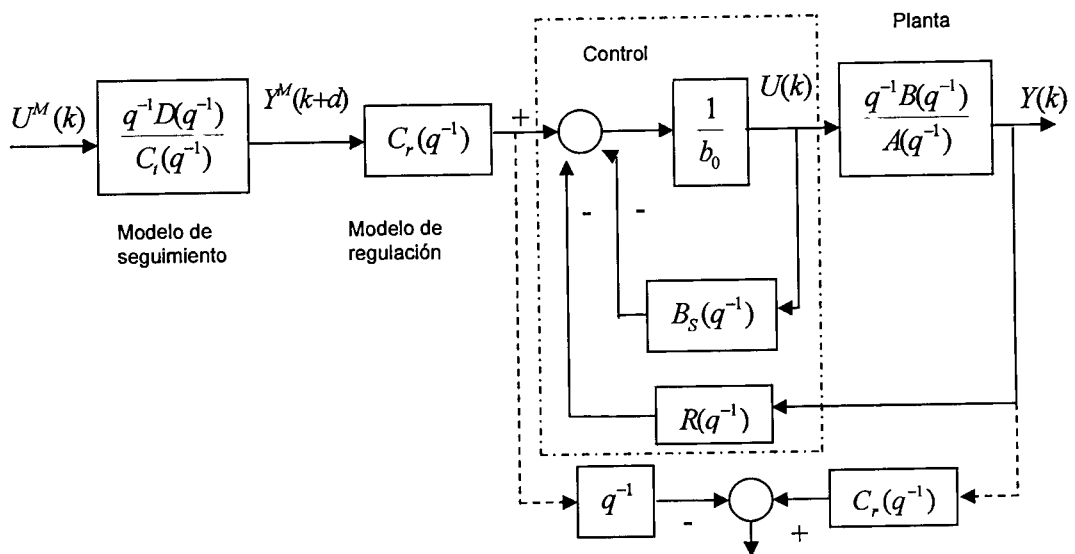


Figura 3.1
Control de seguimiento y regulación

3.2 Identificación recursiva de mínimos cuadrados

Suponga un sistema cuya salida es:

$$y(t) = \theta^T \phi_{t-1} + v_t \quad (3.2.1)$$

θ representa el vector de parámetros

ϕ_{t-1} representa el vector de variables medibles

v_t representa una perturbación

Ley de Control Lineal:

$$u(t) = f_t(u_{t-1}, u_{t-2}, \dots, y_t, y_{t-1}, \dots, y_t^M, y_{t-1}^M, \dots) \quad (3.2.2)$$

El objetivo es obtener el estimado $\hat{\theta}_t$, utilizando las t mediciones y_i , ϕ_{i-1} ; $0 < i \leq t$, que minimicé el criterio de error cuadrático con factor de olvido siguiente:

$$J(\hat{\theta}_t) = \sum_{i=1}^t \left(\prod_{k=i}^t \lambda_k \right) \left(y_i - \hat{\theta}_i^T \phi_{i-1} \right)^2 \quad ; \quad 0 < \lambda_k \leq 1 \quad (3.2.3)$$

El gradiente de J con respecto a $\hat{\theta}_t$ es:

$$J_{\hat{\theta}_t} = 2 \sum_{i=1}^t \left(\prod_{k=i}^t \lambda_k \right) \left(y_i - \hat{\theta}_i^T \phi_{i-1} \right) (-\phi_{i-1}) \quad (3.2.4)$$

Entonces el Hessiano será:

$$J_{\hat{\theta}_t, \hat{\theta}_t} = 2 \sum_{i=1}^t \left(\prod_{k=i}^t \lambda_k \right) (\phi_{i-1} \phi_{i-1}^T) \geq 0 \quad (3.2.5)$$

Una matriz definida positiva, por lo tanto igualando la ec. 3.2.4 a cero obtenemos un mínimo; recordando que:

$$\left(\hat{\theta}_t^T \phi_{i-1} \right) \phi_{i-1} = \phi_{i-1} \left(\phi_{i-1}^T \hat{\theta}_t \right) \quad (3.2.6)$$

Se tiene:

$$\hat{\theta}_t = \left[\sum_{i=1}^t \left(\prod_{k=i}^t \lambda_K \right) \phi_{i-1} \phi_{i-1}^T \right]^{-1} \sum_{i=1}^t \left(\prod_{k=i}^t \lambda_K \right) y_i \phi_{i-1} \quad (3.2.7)$$

Esta expresión se usa para calcular el estimado $\hat{\theta}_t$, fuera de línea, es decir con todas las mediciones hasta el tiempo t.

Existen versiones recursivas o en línea. A continuación se obtiene una de las más utilizadas.

Se define la matriz a invertir en la ec. 3.2.7 como:

$$F_{t+1}^{-1} = \sum_{i=1}^t \left(\prod_{k=i}^t \lambda_K \right) \phi_{i-1} \phi_{i-1}^T \quad (3.2.8)$$

En forma equivalente:

$$F_{t+1}^{-1} = \lambda_t (F_t^{-1} + \phi_{t-1} \phi_{t-1}^T) \quad (3.2.9)$$

Combinando 3.2.7 y 3.2.9 se tiene:

$$\hat{\theta}_t = F_{t+1} \sum_{i=1}^t \left(\prod_{k=i}^t \lambda_K \right) y_i \phi_{i-1} \quad (3.2.10)$$

$$\hat{\theta}_t = F_{t+1} \left[\lambda_t \phi_{t-1} y_t + \sum_{i=1}^{t-1} \left(\prod_{k=i}^t \lambda_k \right) y_i \phi_{i-1} \right] \quad (3.2.11)$$

Reemplazando las ecuaciones 3.2.6 y 3.2.11 se obtiene:

$$\hat{\theta}_t = F_{t+1} \left[\lambda_t \phi_{t-1} y_t + \sum_{i=1}^{t-1} \left(\prod_{k=i}^t \lambda_k \right) \phi_{i-1} \phi_{i-1}^T \hat{\theta}_{t-1} \right] \quad (3.2.12)$$

De la 3.2.8 y 3.2.12 se tiene:

$$\hat{\theta}_t = F_{t+1} \left[\lambda_t \phi_{t-1} y_t + (F_{t+1}^{-1} - \lambda_t \phi_{t-1} \phi_{t-1}^T) \hat{\theta}_{t-1} \right] \quad (3.2.13)$$

ó

$$\hat{\theta}_t = \hat{\theta}_{t-1} + \lambda_t F_{t+1} \phi_{t-1} \left[y_t - \hat{\theta}_{t-1}^T \phi_{t-1} \right] \quad (3.2.14)$$

Utilizando el lema de inversión que dice que si $A_{(m \times m)}$ y $C_{(n \times n)}$ son matrices no singulares, entonces:

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(DA^{-1}B + C^{-1})^{-1}DA^{-1} \quad (3.2.15)$$

Obtenemos de la ecuación 3.2.9 la expresión:

$$F_{t+1} = \frac{1}{\lambda_t} \left[F_t - \frac{F_t \phi_{t-1} \phi_{t-1}^T F_t}{1 + \phi_{t-1}^T F_t \phi_{t-1}} \right] \quad (3.2.16)$$

Que puede ser verificada por multiplicación directa.

3.3 Propiedades del algoritmo de identificación

Para un sistema determinístico, la planta es:

$$y_t = \theta^T \phi_{t-1} \quad (3.3.1)$$

a) $\lambda_t = 1$

El algoritmo de identificación se escribe:

$$\hat{\theta}_t = \hat{\theta}_{t-1} + F_{t+1} \phi_{t-1} \left[y_t - \hat{\theta}_{t-1}^T \phi_{t-1} \right] \quad (3.3.2)$$

b) $\lambda_t < 1$

El algoritmo de identificación será:

$$\hat{\theta}_t = \hat{\theta}_{t-1} + \lambda_t F_{t+1} \phi_{t-1} \left[y_t - \hat{\theta}_{t-1}^T \phi_{t-1} \right] \quad (3.3.3)$$

$$F_{t+1} = \frac{1}{\lambda_t} \left[F_t - \frac{F_t \phi_{t-1} \phi_{t-1}^T F_t}{1 + \phi_{t-1}^T F_t \phi_{t-1}} \right]; \quad F_1 > 0 \quad (3.3.4)$$

Con $0 < \delta \leq \lambda_t \leq 1 - \varepsilon$

Donde δ y ε son números positivos arbitrariamente pequeños, pero distintos de cero;

$$0 < \varepsilon \ll 1 \quad \text{y} \quad 0 < \delta \ll 1$$

3.4 Control adaptable directo de sistemas de fase mínima

Cuando los parámetros del sistema a controlar no se conocen, o cambian su valor en el tiempo, se utiliza un controlador adaptable en base a un algoritmo de identificación véase figura 3.2.

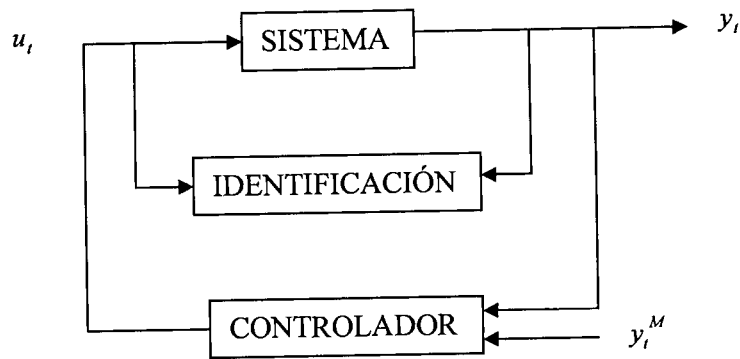


Figura 3.2
Esquema de control con identificación

En el esquema de control adaptable esta basado en la ley de control lineal, pero en este caso los parámetros del controlador se ajustan o actualizan usando los parámetros estimados en la identificación.

Para el caso determinístico ($w_t = 0$) se consideran dos hipótesis:

- El retardo $\{d\}$ es conocido.
- Límites superior n y m (grados de los polinomios A y B) son conocidos.

De la ecuación 3.1.1 y la identidad polinomial 3.1.9 la planta se puede parametrizar como:

$$C_r y_t = (AS + q^{-d}R)y_t = q^{-d}BSu_t + q^{-d}Ry_t \quad (3.4.1)$$

La ecuación 3.4.1 se puede expresar

$$C_r y_t = \theta^T \phi_{t-d} \quad (3.4.2)$$

Donde:

$$\theta^T = [b_0, b_0 s_1 + b_1, b_0 s_2 + b_1 s_1 + b_2, \dots, b_m s_{d-1}, r_0 \dots r_{n_R}] \quad (3.4.3)$$

$$\phi_{i-d}^T = [u_{i-d}, \dots, u_{i-2d+1-m}, y_{i-d}, \dots, y_{i-d-n_R}] \quad (3.4.4)$$

θ se puede identificar con el algoritmo de las ecuaciones 3.3.2 y 3.3.3

$$\hat{\theta}_i = \hat{\theta}_{i-1} + F_{i+1} \phi_{i-d} \left[C_r y_i - \hat{\theta}_{i-1}^T \phi_{i-d} \right] \quad (3.4.5)$$

$$F_{i+1} = F_i - \frac{F_i \phi_{i-d} \phi_{i-d}^T F_i}{1 + \phi_{i-d}^T F_i \phi_{i-d}} \quad (3.4.6)$$

La ley de control lineal se puede expresar como:

$$\theta^T \phi_i = C_r y_i^M \quad (3.4.7)$$

La ley de control adaptable está dada por:

$$\hat{\theta}^T \phi_i = C_r y_i^M \quad (3.4.8)$$

Restando 3.4.2 y 3.4.8 se obtiene:

$$C_r (y_i - q^{-d} y_i^M) = \hat{\theta}_{i-d}^T \phi_{i-d} \quad (3.4.9)$$

3.5 Control adaptable con factor de olvido

Para plantas representadas por la ecuación 3.1 con los parámetros a_i y b_i desconocidos, asumiendo que:

- a) El retardo de tiempo $\{d\}$ es conocido
- b) El orden de los polinomios de $A(q^{-1})$ y $B(q^{-1})$ son conocidos.

La ley de control para en el caso adaptable será:

$$u(t) = \hat{b}_0^{-1}(t) \left[C_r(q^{-1})y^M(t+d) - \hat{\theta}_0^T(t)\phi_0(t) \right] \quad (3.5.1)$$

Ó

$$\hat{\theta}^T \phi(t) = C_r(q^{-1})y^M(t+d) \quad (3.5.2)$$

Donde:

$$\hat{\theta}^T(t) = \left[\hat{b}_0(t); \hat{\theta}_0^T(t) \right] \quad (3.5.3)$$

Introduciendo la ecuación 3.5.2 en la 3.1.14 se tiene el error de seguimiento como una función de la diferencia entre el vector de parámetros verdaderos θ y el de parámetros ajustables $\hat{\theta}(t-d)$.

$$\varepsilon_f(t) = C_r(q^{-1})\varepsilon(t) = \dot{\theta}^T(t-d)\phi(t-d) \quad (3.5.4)$$

Donde $\varepsilon_f(t)$ es el error filtrado modelo-planta: $\dot{\theta}(t) = \theta - \hat{\theta}(t)$ el algoritmo de estimación será:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{F_t \phi(t-d) [C_r(q^{-1})y(t) - \hat{\theta}^T(t-1)\phi(t-d)]}{1 + \phi^T(t-d)F_t \phi(t-d)} \quad (3.5.5)$$

Donde

$$\hat{\theta}(t) = \begin{bmatrix} \hat{b}_0 \\ \hat{b}_1 \\ \hat{r}_0 \\ \hat{r}_1 \end{bmatrix} \quad \phi(t) = \begin{bmatrix} U(k) \\ U(k-1) \\ Y(k) \\ Y(k-1) \end{bmatrix}$$

Con

$$F_{t+1} = \frac{1}{\lambda(t)} \left[F_t - \frac{F_t \phi(t-d) \phi^T(t-d) F_t}{1 + \phi^T(t-d) F_t \phi(t-d)} \right] \quad (3.5.6)$$

En la figura 3.3 se aprecia el esquema general a bloques del sistema de control adaptable directo, donde a través de la identificación de los parámetros de la planta se estiman los valores \hat{B}_s , \hat{b}_0 y \hat{R} . El uso de este algoritmo es aplicable a plantas con dinámicas complicadas (no linealidades, parámetros variantes en el tiempo, retardos, etc.).

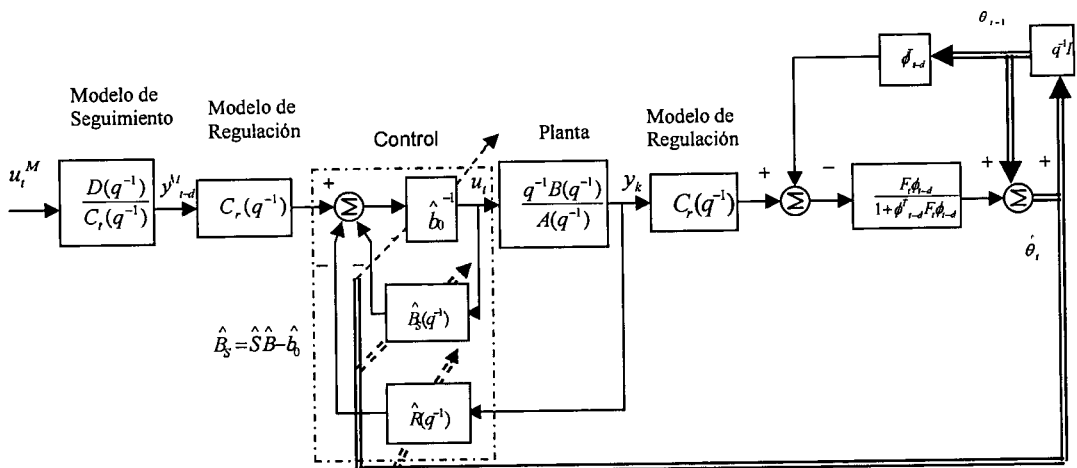


Figura 3.3
Control adaptable directo

CAPÍTULO IV

TOPOLOGÍA DEL SISTEMA DE CONTROL

TOPOLOGÍA DEL SISTEMA DE CONTROL

4.1 Introducción

El esquema general para el sistema de control en hardware se muestra en la figura 4.1, como se puede observar tanto el control, la planta y la comunicación por el protocolo USB se encuentran embebidos dentro del microcontrolador. La referencia es un valor analógico de 0 a 5 volts y se conecta directamente a una de las entradas analógicas del microcontrolador; la planta y el control se desarrollan en ecuaciones de diferencias para ser implementadas en la memoria del microcontrolador y el sistema de comunicación usb es nativo, es decir, esta implementado por hardware y solo se configura por software en el compilador con un lenguaje de programación como C ó Basic, para posteriormente transferir dicha información hacia la computadora a través de un lenguaje de alto nivel como lo es Labview y obtener los resultados gráficos.

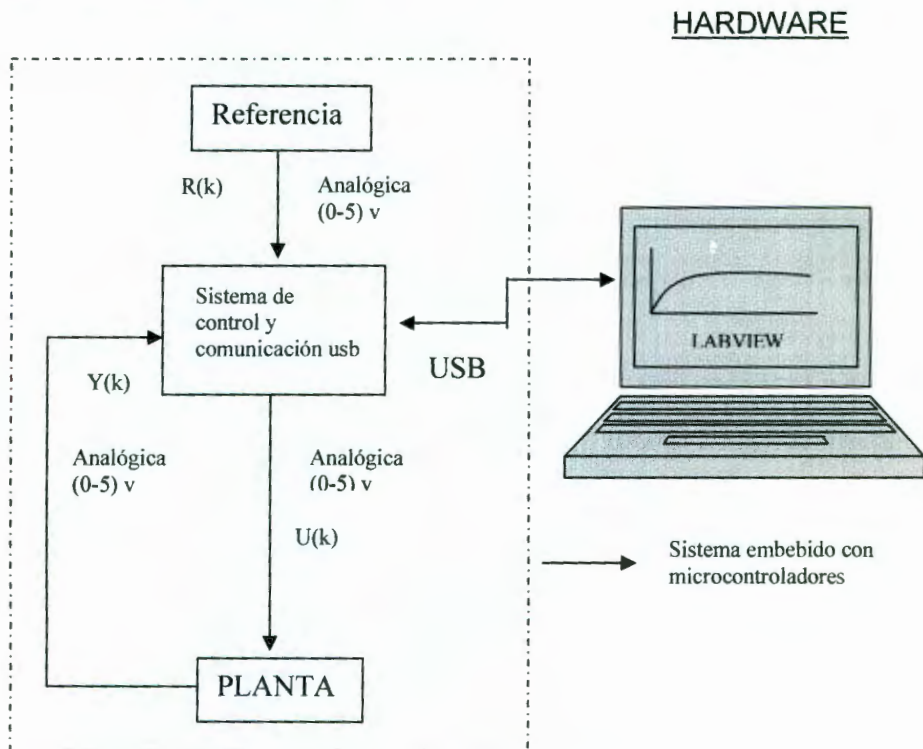


Figura 4.1
Topología del sistema de control en Hardware

La programación de las ecuaciones de diferencias tanto del control como de la planta se implementaron en el lenguaje C, por ello primeramente se opto por utilizar el compilador Visual C++ para poder simular y verificar que funcionaran correctamente, posteriormente se porto dicho código en C a la plataforma del copilador PCWH PIC-C para así poder obtener el firmware necesario tanto de la planta como del controlador, a su vez también se trabajo en paralelo con Matlab para ver gráficamente los datos y saber con anticipación los resultados mostrados por el microcontrolador, después de ello se utilizo el Winpic800 que es un software de programación de firmwares para microcontroladores de la serie Microchip, y finalmente la plataforma de LabVIEW que sirve para obtener los resultados gráficos. Cabe mencionar que el software Picdem es un cargador bootloader, es decir que se puede programar el microcontrolador sin necesidad de un programador como lo es Winpic800, pero con la salvedad de que anteriormente se tiene que grabar un firmware solo una vez para que se pueda realizar las programaciones subsecuentes las veces que se requieran. En la figura 4.2 se observa la topología de configuración en Software para el sistema de control.

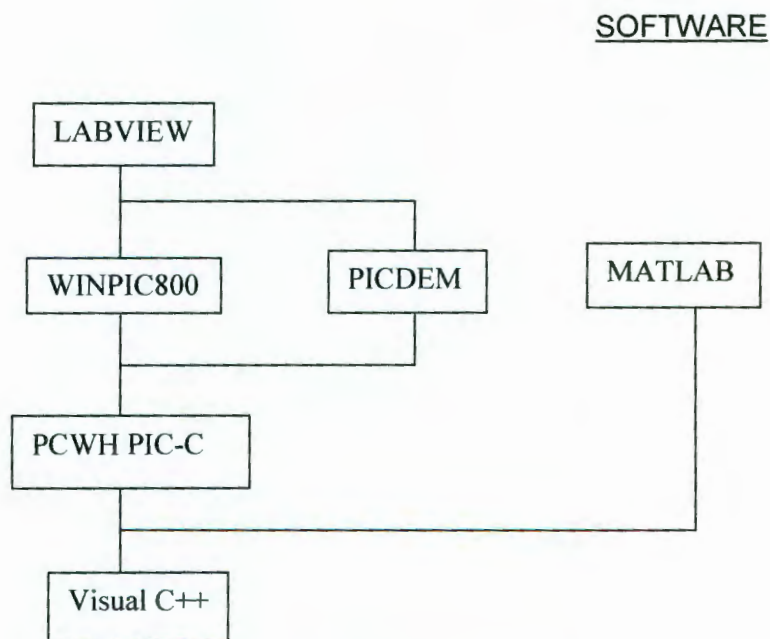


Figura 4.2
Topología del sistema de control en Software

4.2 Tarjeta de Control y Adquisición de Datos USB

Una de las principales herramientas para la elaboración e implantación de los diferentes algoritmos de control que se desarrollaron durante este trabajo es la tarjeta de adquisición de datos que se puede definir como sigue:

Es una tarjeta de control y adquisición de datos configurable y programable de comunicación versátil para la transferencia bidireccional de señales tanto analógicas como digitales por medio del protocolo de comunicación USB.

Se compone principalmente de los siguientes elementos:

- Hardware → Entradas Digitales = 8, Salidas Digitales = 8 y Entradas Analógicas = 8.
- Software → Modulo de control y adquisición de datos para Labview

Características del la tarjeta de adquisición de datos usb:

- No requiere alimentación externa
- Plug and Play
- Bajo Consumo de Energía
- Pequeño
- Fácil de programar
- Versátil

En la figura 4.3 se muestra de forma general el esquemático de la tarjeta de adquisición de datos, cabe mencionar que dentro del firmware del microcontrolador PIC18F4550 se encuentra implementado la planta, el controlador y el sistema de comunicación USB; el diseño se elaboro de esta manera ya que permitía que el sistema fuera inmune a ruidos, fuera más rápido y sobretodo de bajo costo. En la figura 4.4 se muestra la tarjeta ya terminada, los potenciómetros sirven para simular entradas analógicas, los leds para simular salidas digitales y los push button para entradas digitales.

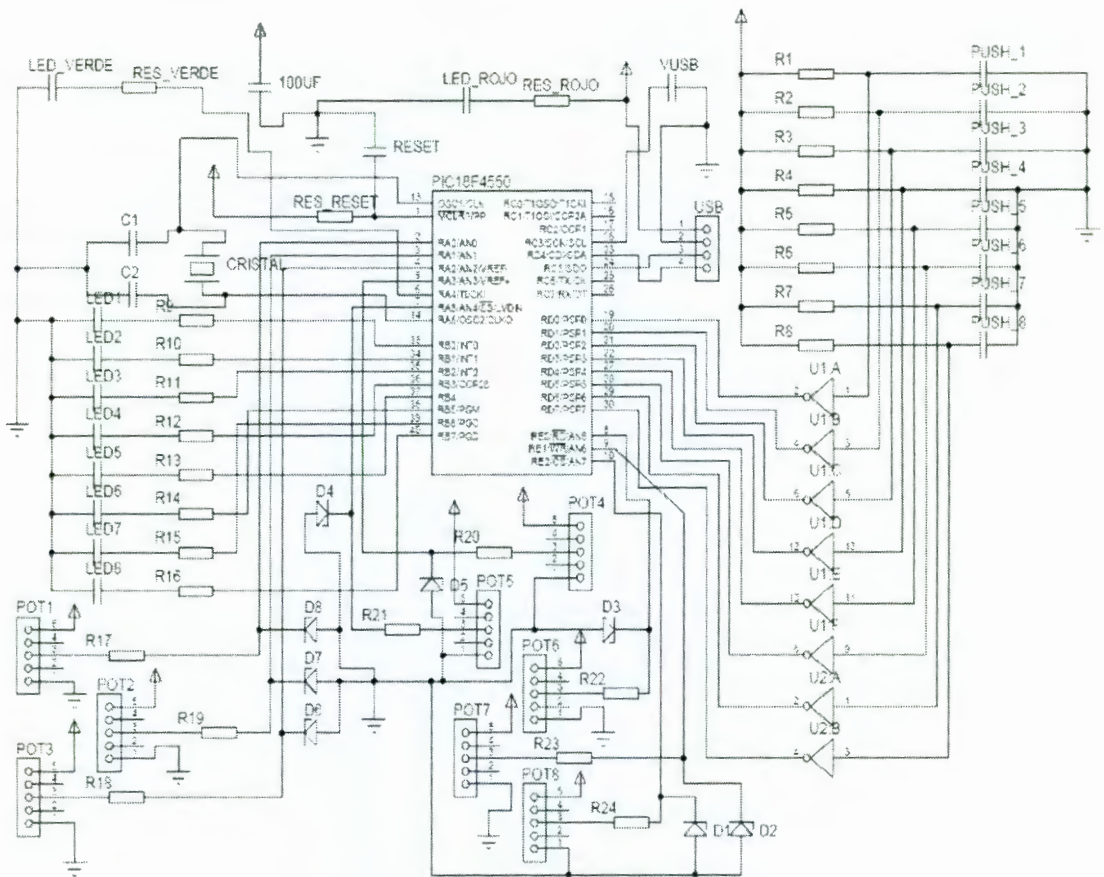


Figura 4.3
Esquemático de la tarjeta de adquisición de datos USB

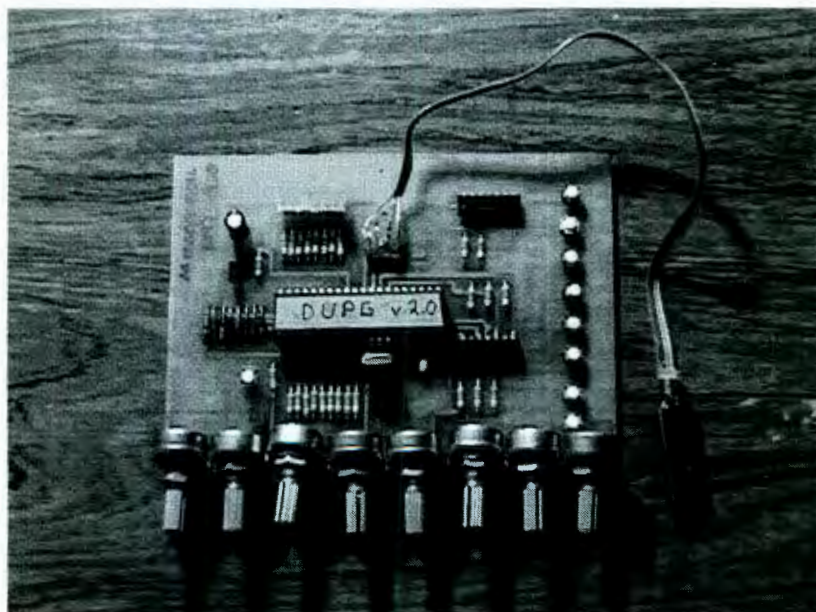


Figura 4.4
Tarjeta de control y adquisición de datos USB

4.2.1 Microcontrolador PIC18F4550

El PIC18F4550 es un microprocesador de propósito general versátil y económico. Pertenece a la popular familia de procesadores PICmicro de la empresa norteamericana Microchip cuya sede se ubica en Chandler, Arizona (USA).



Figura 4.5
PIC18F4550 - empaquetado DIP-40

Lo particular del procesador PIC18F4550 es que es uno de los PICs que viene con soporte nativo para USB, lo cual quiere decir que incluyen un controlador USB interno que ya brinda patas de salida para conectar directo a la PC, sin la necesidad de *pull-ups* o ninguna circuitería externa.

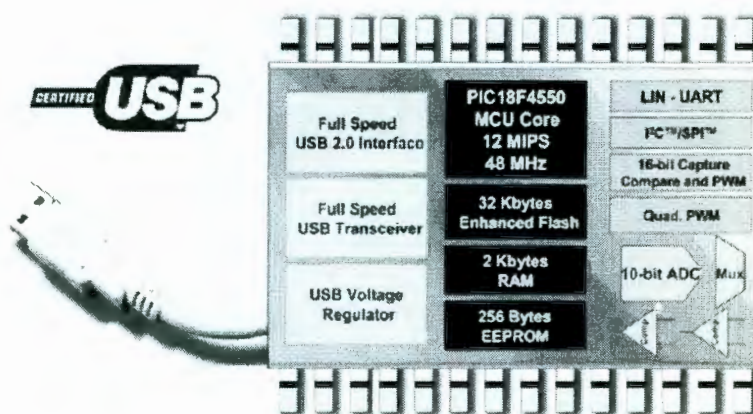


Figura 4.6
Diagrama de bloques del PIC18F4550

Soporta cristales y osciladores de varias frecuencias como entrada y tiene post-scaler de manera que el procesador pueda trabajar a una frecuencia de 48 Mhz, independiente del oscilador que se conecte. Para ello debe configurarse (a través de los configuration bits) el oscilador que se le ha conectado. Trabajar a 48 Mhz es un requisito para poder transferir a full-speed por el puerto USB, es decir, a 1.5 Mbytes/seg y es compatible con el estándar USB 2.0.

También cuenta con 35 patas de entrada/salida digitales de propósito general (figura 4.7) y viene disponible en varios empaquetados, entre ellos DIP-40 lo cual lo hace una alternativa muy popular entre desarrolladores y electrónicos. Los puertos de entrada/salida son todos compatibles con la tecnología TTL. Cuando se los utiliza como salida, se comporta como un CMOS, siendo compatible con TTL, de modo de poder manejar cualquier tipo de tecnología. Sin embargo cuando son configurados los puertos como entrada, hay dos comportamientos posibles: puede ser exclusivamente TTL, o puede ser configurado para TTL o CMOS.

En cuanto a memoria, posee 32Kb de flash para almacenamiento de programas, 2Kb de SRAM para memoria volátil, y 256 bytes de EEPROM (memoria no-volátil) para almacenamiento permanente de datos como configuraciones y demás.

Otras características interesantes que posee son timers, interrupciones (externas e internas por timers) con dos niveles de prioridad y disparadas tanto por nivel como por flanco, un comparador analógico con un generador de voltaje de referencias.

Por último, el PIC también cuenta con un convertor analógico de 10-bit pero con una frecuencia de muestreo de 200 ksps. Ya que, si bien el oscilador es de 48 Mhz, entre los tiempo de ejecución de las interrupciones y otros delays (bucles, etc) no se pueden obtener velocidades de captura mayores a 200 ksps.

A continuación se presenta el pinout del PIC18F4550, en empaquetado DIP-40. En particular se puede reconocer las pines D- y D+ de la conexión USB (patas 23 y 24).

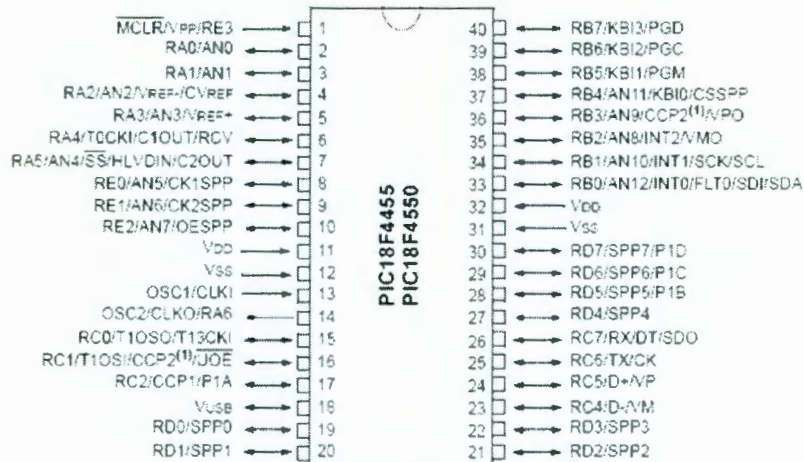


Figura 4.7
Pinout del PIC18F4550

Características generales del microcontrolador pic18f4550

Device	Program Memory		Data Memory		I/O	10-bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EUSART	Comparators	Timers 8/16-bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI™	Master I ² C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

Figura 4.8
Características generales del PIC18F4550

4.2.2 Protocolo de comunicación USB

Introducción

La tecnología USB ha sido promovida principalmente por Intel, aunque le han seguido todos los grandes fabricantes, de forma que se ha convertido en un estándar importante. En sus comienzos los interesados en esta tecnología se agruparon en un foro, el USB Implementers Forum Inc., USB-IF, que agrupa a más de 460 compañías, y ha publicado diversas revisiones de la norma:

- USB 0.9: Primer borrador, publicado en Noviembre de 1995.
- USB 1.0: Publicada en 1996 establece dos tipos de conexión: La primera, denominada velocidad baja ("Low speed"), ofrece 1.5 Mbps, y está pensada para periféricos que no requieren un gran ancho de banda, como ratones o joysticks. La segunda, denominada velocidad completa ("Full speed"), es de 12 Mbps, y está destinada a los dispositivos más rápidos.
- USB 1.1: Publicada en 1998, añade detalles y precisiones a la norma inicial; es el estándar mínimo que debe cumplir un dispositivo USB.
- USB 2.0: Su versión final fue publicada en Abril del 2000; es una extensión de la norma compatible con las anteriores. Permite velocidades de hasta 480 Mbps, denominada alta velocidad ("High speed").

Historia

El primer ordenador que incluyó un puerto USB de forma estándar fue el iMac de Apple, presentado en Marzo de 1998, que utilizaba esta conexión para el teclado y el ratón. Por su parte el mundo del PC solo comenzó a utilizarlo cuando Microsoft introdujo los controladores correspondientes en la versión OSR 2.1 de Windows 95. Fue a partir de Windows 95C cuando los sistemas de MS incorporan de forma estándar soporte para este bus. En el ámbito de servidores la incorporación se produjo en Windows 2000.

Los primeros dispositivos que empezaron a utilizar este tipo de conexión fueron las cámaras de video-conferencia, aunque actualmente (2006) pueden encontrarse todo tipo de dispositivos. El resultado es que, junto con los dispositivos inalámbricos (algunos de los cuales se conectan también a través de esta interfaz), la conexión USB se ha convertido en el método universal de conexión de periféricos, incluyendo dispositivos de almacenamiento y los denominados HID ("Human Interface Device") principalmente ratones y teclados.

Topología

Los dispositivos USB adoptan una topología de estrella y se organiza por niveles a partir de un controlador host instalado en la placa base, que actúa de interfaz entre el bus de ésta y el primer dispositivo USB, el denominado concentrador raíz ("Root hub"), instalado también en la placa. El controlador de host es único; suele ser un chip Intel con una denominación como 82371AB/EB; 82801DB, etc. Dada la proliferación de este tipo de dispositivos, las placas modernas pueden disponer de varias concentradoras raíces, cada uno con su propia salida (generalmente 2 conectores del tipo "A" por cada uno de ellos). Cada uno de estos concentradores se considera el origen de un bus (numerados sucesivamente a partir del 0), del que cuelgan los dispositivos en el orden en que son detectados por el Sistema.

El bus USB soporta intercambio simultáneo de datos entre un ordenador anfitrión y un amplio conjunto de periféricos. Todos los periféricos conectados comparten el ancho de banda del bus por medio de un protocolo de arbitraje basado en testigos ("Tokens"). El bus permite conexión y desconexión dinámica, es decir, que los periféricos se conecten, configuren, manipulen y desconecten mientras el sistema anfitrión y otros periféricos permanecen en funcionamiento.

En un bus USB existen dos tipos de elementos: Anfitrión ("host") y dispositivos; a su vez, los dispositivos pueden ser de dos tipos: concentradores y funciones.

Los *concentradores* ("Hubs") son el centro de una estrella, y sirven para conectar con el sistema anfitrión, con otro hub o con una función. Cada hub puede conectar hasta 7 dispositivos, aunque lo normal es que sean de 4 salidas, y proporcionar 500 mA de energía de alimentación (hasta 2.5 W) a cada uno de ellos, ya que el cable de conexión tiene hilos de señal (datos) y de alimentación (5 V. CC \pm 0.25 V).

Una *función* es un dispositivo capaz de transmitir o recibir datos o información de control en un bus USB, suele conectarse como un dispositivo independiente enlazado por un cable de menos de 5 metros, a un puerto del hub o directamente al sistema anfitrión. De esta descripción se desprende que cada segmento del bus representa una conexión apunto a punto de alguno de los tipos siguientes:

Sistema anfitrión \leftrightarrow Función

Sistema anfitrión \leftrightarrow Concentrador

Concentrador \leftrightarrow Concentrador

Concentrador \leftrightarrow Función.

Que un hub pueda estar conectado a otro hub, significa que pueden conectarse dispositivos en cascada; el sistema soporta un total de 127 dispositivos. Una característica importante es que el concentrador (hub), proporcionan la energía necesaria a la función por el cable de conexión (que transporta fuerza y datos), lo que evita la necesidad de fuentes de alimentación independientes a las funciones.

Funcionamiento

El bus serie USB es *síncrono*, y utiliza el algoritmo de codificación *NRZI* ("Non Return to Zero Inverted"). En este sistema existen dos voltajes opuestos; una tensión de referencia corresponde a un "1", pero no hay retorno a cero entre bits, de forma que una serie de unos corresponde a un voltaje uniforme; en cambio los ceros se marcan como cambios del nivel de tensión, de modo que una sucesión de ceros produce sucesivos cambios de tensión entre los conductores de señal.

A partir de las salidas proporcionadas por los *concentradores raíz* (generalmente conectores del tipo "A") y utilizando concentradores adicionales, pueden conectarse más dispositivos hasta el límite señalado.

El protocolo de comunicación utilizado es de testigo, que guarda cierta similitud con el sistema Token-Ring de IBM. Puesto que todos los periféricos comparten el bus y pueden funcionar de forma simultánea, la información es enviada en paquetes; cada paquete contiene una cabecera que indica el periférico a que va dirigido. Existen cuatro tipos de paquetes distintos: *Token*; *Datos*; *Handshake*, y *Especial*; el máximo de datos por paquete es de 8; 16; 32 y 64 Bytes. Se utiliza un sistema de detección y corrección de errores bastante robusto tipo CRC ("Cyclical Redundancy Check").

El funcionamiento está centrado en el host, todas las transacciones se originan en él. Es el controlador host el que decide todas las acciones, incluyendo el número asignado a cada dispositivo (esta asignación es realizada automáticamente por el controlador "host" cada vez que se inicia el sistema o se añade, o elimina, un nuevo dispositivo en el bus), su ancho de banda, etc.

Cuando se detecta un nuevo dispositivo es el host el encargado de cargar los drivers oportunos sin necesidad de intervención por el usuario.

El sistema utiliza cuatro tipos de transacciones que resuelven todas las posibles situaciones de comunicación. Cada transacción utiliza un mínimo de tres paquetes, el primero es siempre un Token que avisa al dispositivo que puede iniciar la transmisión.

1) *Transferencia de control* ("Control transfer"): Ocurre cuando un dispositivo se conecta por primera vez. En este momento el controlador de host envía un paquete "Token" al periférico notificándole el número que le ha asignado.

2) *Transferencia de pila de datos* ("Bulk data transfer"): Este proceso se utiliza para enviar gran cantidad de datos de una sola vez. Es útil para dispositivos que tienen que enviar gran cantidad de datos cada vez, como escáneres o máquinas de fotografía digital.

3) *Transferencia por interrupción* ("Interrupt data transfer"): Este proceso se utiliza cuando se solicita enviar información por el bus en una sola dirección (de la función al host).

4) *Transferencia de datos isócrona* ("Isochronous data transfer"): Este proceso se utiliza cuando es necesario enviar datos en tiempo real. Los datos son enviados con una cadencia precisa ajustada a un reloj, de modo que la transmisión es a velocidad constante.

Nota: Las comunicaciones asíncronas ponen más énfasis en garantizar el envío de datos, y menos en su temporización ("cuando" llegan); por su parte las comunicaciones isócronas son justamente lo contrario, ponen más énfasis en la oportunidad de la transmisión que en la velocidad. Esta sincronización es importante en situaciones como la reproducción de video, donde no debe existir desfase entre las señales de video y audio.

Clases de dispositivos

Los dispositivos que se conectan pueden tener sus propios drivers personalizados. Sin embargo, existe lo que se llaman clases de dispositivos que son pequeños estándares para distintos tipos de dispositivos y especifican como deben compartirse los dispositivos en términos de los descriptores de interfaz y de dispositivo, endpoints, etc. Esto permite que todos los dispositivos sean fácilmente intercambiables y/o sustituibles puesto que hablan el "mismo idioma". Por su parte, los sistemas operativos solo tienen que implementar drivers genéricos para todas las clases conocidas de dispositivos lo cual alivia el alto costo de desarrollar y mantener un driver particular para cada producto y modelo. En conclusión, las clases de dispositivos son una estandarización de funcionalidades a un nivel superior al del bus USB y que utiliza a éste último como medio de comunicación e intercambio de datos. Tanto los descriptores de dispositivos como los descriptores de interfaz tienen un byte que identifica la clase. En el primer caso, todo el dispositivo/función pertenece a la misma clase, mientras que en el segundo un mismo dispositivo puede tener diferentes clases, cada una asociada a su descriptor de interfaz.

Dado que el identificador de la clase es un byte, existe un máximo de 253 clases diferentes (0x00 y 0xFF están reservados). Los códigos de las clases son asignados por el USB Implementers Forum, y a continuación en la figura 4.9 se presenta una lista de los más comunes.

Código	Clase
0x00	Reservado. Usado en el descriptor de dispositivo para indicar que la clase está identificada en el (o los) descriptores de interfaz
0x01	Dispositivo de audio. Por ejemplo; tarjetas de sonidos
0x03	Dispositivo de interfaz humana (HID). Por ejemplo: mouses, teclados, joystick
0x07	Impresoras
0x08	Dispositivo de almacenamiento masivo. Por ejemplo: discos duros, lectores de memoria, cámaras digitales, reproductores MP3
0x09	Hubs
0x0A	Dispositivo de comunicación (CDC por sus siglas en inglés). Por ejemplo: módems, tarjetas de red
0x0E	Dispositivo de video. Por ejemplo: webcams
0xE0	Controladores inalámbricos. Por ejemplo: adaptadores Bluetooth
0xFF	Reservado. Usado para indicar que el dispositivo tiene un driver personalizado propio que no pertenece a ninguna clase

Figura 4.9
Clases de dispositivo USB

La tarjeta de control y adquisición de datos USB en particular pertenece a la clase (0xFF), ya que utiliza drivers personalizados, estos drivers son proporcionados por el fabricante del microcontrolador PIC18F4550, en este caso por la empresa Microchip; cabe mencionar que también se pueden diseñar los propios drivers con herramientas propietarias como las DDK de MS.

Cables y conectores

El cable de bus USB es de 4 hilos, y comprende líneas de señal (datos) y alimentación, con lo que las funciones pueden utilizar un único cable. Existen dos tipos de cable: apantallado y sin apantallar.

En el primer caso el par de hilos de señal es trenzado; los de tierra y alimentación son rectos, y la cubierta de protección (pantalla) solo puede conectarse a tierra en el anfitrión.

En el cable sin apantallar todos los hilos son rectos. Las conexiones a 15 Mbps y superiores exigen cable apantallado.

AWG	mm Ø	long. máx.
28	0.321	0.81 m
26	0.405	1.31 m
24	0.511	2.08 m
22	0.644	3.33 m
20	0.812	5.00 m

Pin	Nombre	Descripción	Color
1	VBUS	+ 5 V. CC	rojo
2	D-	Data -	azul
3	D+	Data +	amarillo
4	GND	Tierra	verde

Figura 4.10
Características del cable USB

Se utilizan diámetros estándar para los hilos de alimentación del bus. Para cada sección se autoriza una longitud máxima del segmento. En la figura 4.10 izquierda se muestran estas distancias y a la derecha se muestra la disposición de pines y colores de identificación.

Se usan dos tipos de conectores, A y B. Ambos son polarizados (solo pueden insertarse en una posición) y utilizan sistemas de presión para sujetarse. Los de tipo A utilizan la hembra en el sistema anfitrión, y suelen usarse en dispositivos en los que la conexión es permanente (por ejemplo, ratones y teclados). Los de tipo B utilizan la hembra en el dispositivo USB (función), y se utilizan en sistemas móviles (por ejemplo, cámaras fotográficas o altavoces). En general podemos afirmar que la hembra de los conectores A están en el lado del host (PC) o de los concentradores (hubs), mientras las de tipo B están del lado de los periféricos.

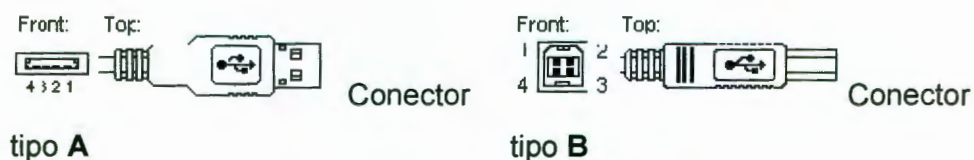


Figura 4.11
Conectores USB

4.2.3 Programador gtp usb lite

Una de las herramientas que se utilizaron para programar el microcontrolador Pic18f4550 de Microchip, fue el programador GTP USB LITE en conjunto con el software Winpic800.

Al inicio de la investigación que se realizó, fue que la mayoría de los programadores no contaban con la opción de poder programar el PiC18F4550, en caso concreto el programador PICSTART plus, jdm y propic II que son con los que se contaba, por eso se tubo que implementar el GTP USB LITE ya que de lo contrario se tendría que comprar uno comercial como lo es el MPLAB PM3 Universal Device Programmer o el PRO MATE II ambos de Microchip.

El GTP USB Lite es un grabador ICSP para PIC's y memorias que hace uso del puerto USB. Permite grabar todos los PICs de la serie F que soporten el modo de grabación ICSP (In-Circuit Serial Programming) y que estén incluidos en el WinPIC800. Al ser un grabador ICSP no tiene placa de zócalos, esto se ha hecho con la intención de que el diseño fuera compacto. Para grabar un PIC se deben conectar cada una de las líneas de esa salida ICSP a los pines correspondientes en el PIC (o memoria) a grabar.

En la figura 4.12 se muestra el esquemático del programador, la situación de componentes en la placa y en la figura 4.13 unas imágenes del programador terminado.

Ventajas de usar el programador gtp usb lite:

- 1.- Económico de construir
- 2.- Rápida programación gracias al puerto Usb 2.0
- 3.- Pequeño y versátil.
- 4.- No necesita alimentación externa.
- 5.- Programa de la mayoría de Pic's de la serie 16F, 18F, Dspic's y memorias.
- 6.- Compatible con todos los sistemas operativos Windows.

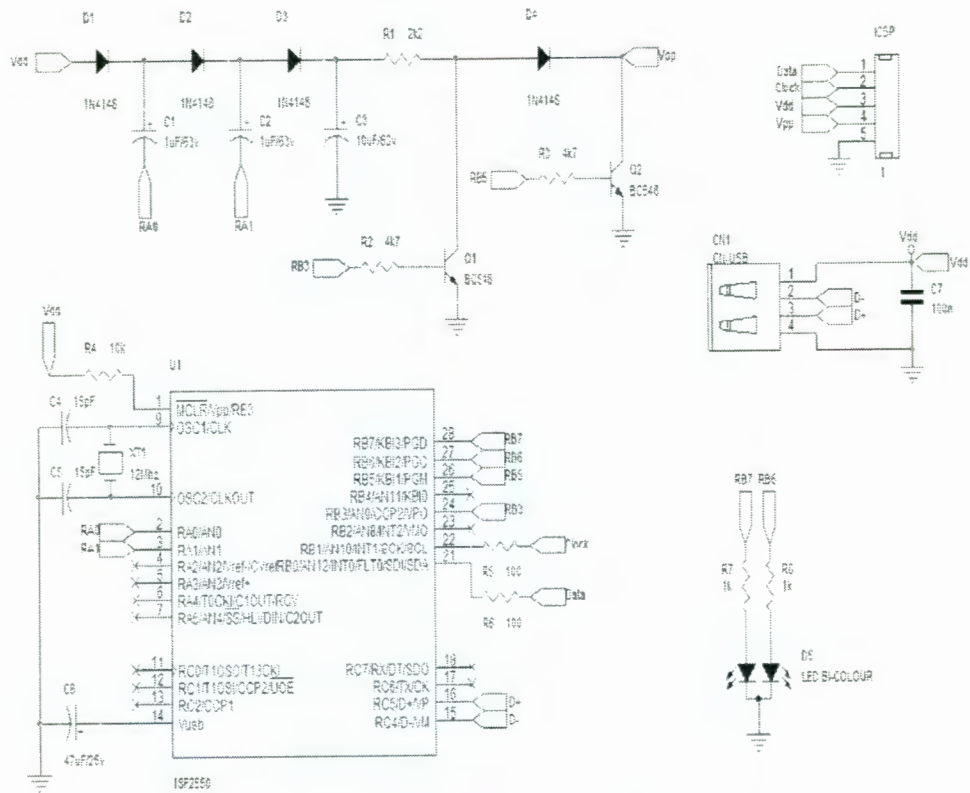


Figura 4.12
Esquemático del programador GTP USB LITE

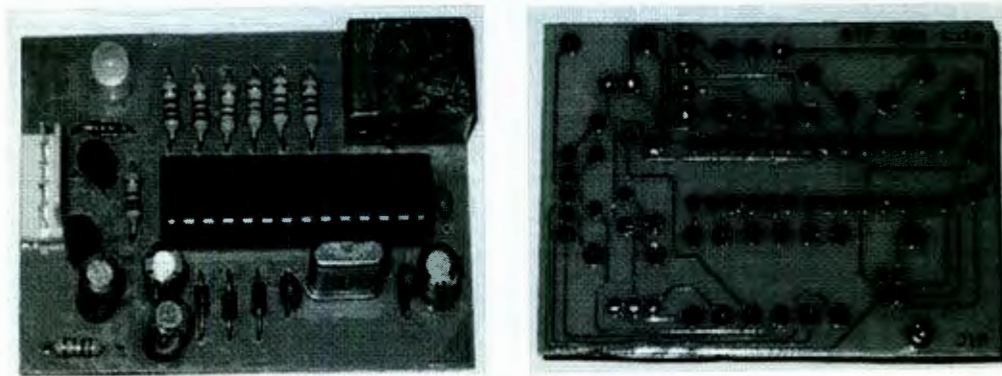


Figura 4.13
Programador GTP USB LITE terminado

4.3 Software de soporte

Driver (Vendor ID y Product ID)

El estándar USB exige que todos los dispositivos, durante la etapa de enumeración, se identifiquen con un *Vendor ID* y un *Product ID* (en adelante, VID y PID). Dicho par de valores sirve para conocer el fabricante del dispositivo (VID) y el modelo particular del producto que se ha conectado (PID). Por lo tanto, modelos diferentes de un mismo producto generalmente tienen PIDs diferentes.

La utilidad principal de estos valores no es solamente la de identificar el dispositivo, sino la de encontrar y cargar los drivers apropiados para el mismo. Por consiguiente, cada driver que viene con Windows viene programado (al igual que el dispositivo) con uno o más PID y VID para los cuales sirve dicho driver. Esta es la forma que tiene Windows (o el sistema operativo en cuestión) de saber si el driver seleccionado es correcto. En el caso de que el driver ya venga con el sistema operativo, el par VID/PID bastará para identificar el driver que es necesario cargar y por lo tanto cuando se conecta un dispositivo con VID/PID conocido el sistema lo detecta automáticamente e inmediatamente queda listo para usar.

Sin embargo, en el caso de que el VID/PID no sea reconocido, el sistema operativo solicitará al usuario que suministre los drivers. Un ejemplo de esto es la conocida pantalla de Nuevo Hardware Detectado de Windows.

Driver para Windows 98/2000/XP

Como ya mencionamos anteriormente, la tarjeta de control y adquisición de datos USB utiliza drivers únicos para comunicarse con la PC por el puerto USB.

Sin embargo, Windows a priori no sabe que driver debe asignar a nuestra tarjeta USB porque no reconoce el par VID/PID con el cual éste se identifica. Por lo tanto, es necesario indicarle a Windows que driver debe utilizar para nuestro VID/PID. Esto se hace a través de un archivo USB.inf donde se especifica, entre otras cosas:

- PID/VID del dispositivo (nuestro caso PID = 0x4D8 y VID = 0x004)
- El driver a utilizar (mchpusb.sys, wdmstub.sys)
- Nombre con el que aparecerá la tarjeta en la lista de dispositivos conectados (en este caso como una clase Dispositivo usb de propósito general).

Proceso de instalación

Al conectar por primera vez la tarjeta de control y adquisición de datos USB a la computadora aparecerá la siguiente pantalla donde se solicitan los drivers de la tarjeta, los cuales son: mchpusb.sys y wdmstub.sys, que están definidos dentro de la declaración de un archivo principal USB.ini, éste archivo es el que es solicitado por el sistema operativo para proceder a la instalación a través de un asistente como el de la figura 4.14

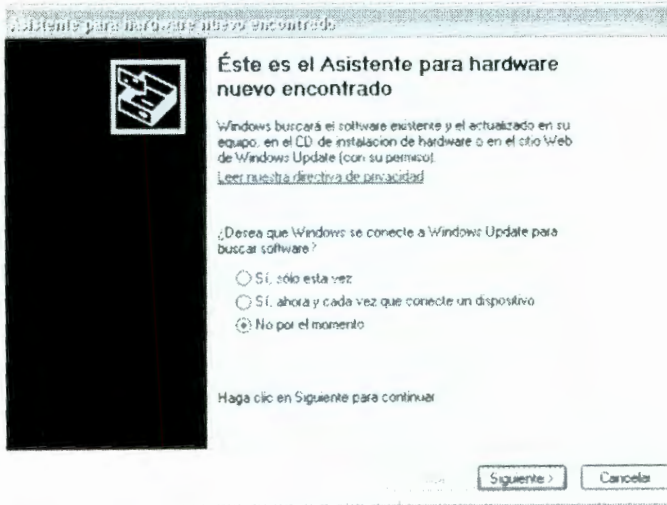


Figura 4.14
Asistente para la instalación de los drivers

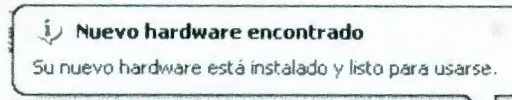
Una vez que se da la ubicación de dichos drivers se procede automáticamente a la instalación de la tarjeta por parte del sistema operativo.

Instalando...



Figura 4.15
Instalando los drivers

Al final aparece este anuncio:



Y en el hardware de la tarjeta el led verde estará prendido anunciando que la tarjeta esta adecuadamente configurada con el host del usb de la pc.

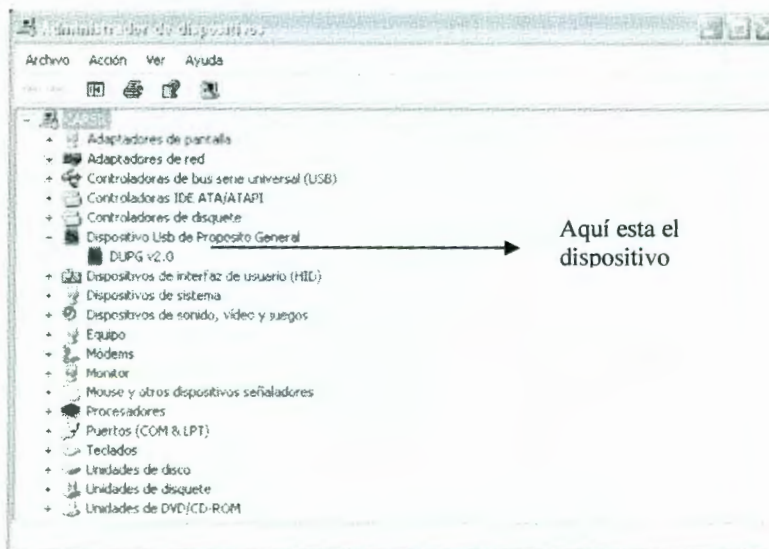


Figura 4.16
Tarjeta de control y adquisición de datos Instalada

4.3.1 Generación y grabación del firmware

Este subcapítulo trata sobre los temas relacionados con la implementación del firmware, es decir, el programa que corre en el microcontrolador y controla el funcionamiento de la tarjeta de control y adquisición de datos USB. Incluye información sobre la estructura del código y detalles particulares de la implementación.

Herramientas de trabajo

El firmware es el programa que corre internamente en el PIC, fue escrito enteramente en lenguaje C utilizando el PCW C Compiler, un compilador de C provisto por el fabricante CCS (Custom Computer Services, Inc.), en la figura 4.17 se aprecia la interfaz IDE del compilador. Una característica destacable del PCW C Compiler es la posibilidad de generar binarios optimizados (tanto en espacio, como cantidad de instrucciones) para PICs de la familia PIC18F (por ejemplo, nuestro PIC18F4550) utilizando las instrucciones extendidas provistas por dicha arquitectura.



```
PCW C Compiler IDE
File Project Edit Options Compile View Tools Debug Help
Microchip PIC18
prueba.c
20 struct matriz
21 {
22     float elemento[4][4];
23 };
24 typedef struct matriz mat;
25
26
27 int contrRTCC=0,flag=0;
28
29 #int RTCC // interrupción por desbordamiento
30 void interrupcion() // cada 250 milisegundos
31 {
32     ++contrRTCC;
33     if(contrRTCC==2)
34         set_TIMER0(8000);
35     if(contrRTCC==3)
36     {
37         flag=1;
38         contrRTCC=0;
39     }
40 }
41
42
43 void main(void)
44 {
45     int out_data[10]; //solo es posible usar registros de 8 bit para usb
46     int in_data[10]; // no se puede usar long int
47     int n=0,m=0,u=0,k=0,s=1,w=0,j=1,a=0,b=0,l=0,o=0;
48     float Y=0,V_1=0,V_2=0,U=0,U_1=0,U_2=0,Vn_1=0,Vn=0,Vnt=0,R_1=0,R=0;
49     mat l,F1,Ft,Ft1,Te,Te_1,T1,T2,T3,T4;
50
51     Vn=(0.2*Vn_1)+(0.8*R_1);
52
53     For(n=0;n<4;n++)
54     {
55         For(m=0;m<4;m++)
56         {
57             l1.elemento[n][m]=0;
58             l2.elemento[n][m]=0;
```

Figura 4.17
PCW C Compiler

Todo código en el compilador PCW C Compiler inicia con un encabezado donde se declara el tipo de microcontrolador a utilizar, la velocidad del CPU, los puertos de entrada-salida y los fusibles de programación como se muestra en la lista de abajo del 1 al 4; para la especificación del protocolo USB es necesario agregar una librería, ésta incluye todas las especificaciones, capas, descriptores, buffers, tokens del protocolo USB, además se deben agregar algunas definiciones (#define) para la especificación de los endpoints y modos de transferencia así como también el tamaño de los mismos.

```
1) #include <18F4550.h>
2) #fuses HSPLL, NOWDT, NOLVP, NODEBUG, USBDIV, PLL5, CPUDIV1, VREGEN
3) #use delay(clock=48000000)
4) #use fast_io(B)
5) #DEFINE USB_HID_DEVICE FALSE
6) #define USB_EPl_TX_ENABLE USB_ENABLE_BULK
7) #define USB_EPl_RX_ENABLE USB_ENABLE_BULK
8) #define USB_EPl_TX_SIZE 1
9) #define USB_EPl_RX_SIZE 1
10) #include <libreria_Usb.h>
```

Ahora bien se agrega la función principal main() y dentro de ella se especifica lo siguiente: la declaración de variables (línea 13) cuyo función es almacenar un dato que se va a enviar o recibir respectivamente, también se declara la inicialización del protocolo USB (líneas 15 y 16), después de ello se enlaza un bucle infinito (línea 17) y se verifica constantemente si el bus USB esta enumerado (línea 19), si es afirmativo se procede a enviar el numero 22 que fue almacenado (línea 21) al host de la pc a través de la función de la línea 22, de lo contrario no hace nada el programa, por otro lado si del host de la computadora envía un dato este es recibido a través de una interrupción (línea 23) y se guarda en la variable in_data y posteriormente se envía el dato al puerto de salida B del microcontrolador, esto se hace en las líneas 25 y 26 respectivamente.

La programación de las ecuaciones de diferencias tanto del controlador adaptable así como de la planta se basaron en las mismas configuraciones comentadas anteriormente, con la salvedad de que habrá que definir una estructura matricial y una función timer que sincronizara los procesos para la adquisición de datos, interrupciones, envió de información, etc.

```

11) void main(void)
12) {
13)     int out_data,in_data;
14)     set_tris_b(0);
15)     usb_init();
16)     usb_wait_for_enumeration();
17)     while (TRUE)
18)     {
19)         if(usb_enumerated())
20)         {
21)             out_data = 22;
22)             usb_put_packet(1, out_data, 1, USB_DTS_TOGGLE);
23)             if (usb_kbhit(1))
24)             {
25)                 usb_get_packet(1, in_data, 1);
26)                 output_b(in_data);
27)             }
28)         }
29)     }
30) }

```

Una vez que se haya programado todo el código en C, se procede a compilarlo con el PCW C Compiler, éste software nos proporcionará un archivo hexadecimal necesario para poderse grabar dentro de la memoria del microcontrolador; ahora bien para poder grabar ese archivo hexadecimal necesitamos el software WinPic800 junto con el programador Gtp Usb Lite anteriormente visto, en la figura 4.18 se muestra el software WinPic800; en dicho programa se encuentran opciones de detección del programador así como también del microcontrolador a programar, una vez hecho lo anterior se procede a cargar el archivo hexadecimal y a grabarlo en el microcontrolador, al final el propio software indicara si la grabación fue exitosa o si hubo algún error en alguna parte del archivo. Para asegurarse de que el firmware es el correcto se puede hacer una verificación del firmware interno de la memoria del microcontrolador contra el archivo hexadecimal original, una vez hecho todo ese proceso de grabado y verificación se retira el microcontrolador del programador y se inserta en el zócalo de la tarjeta de control y adquisición de datos USB. Se comprueba si el firmware funciona bien con la aplicación gráfica de la computadora en este caso referente a la plataforma de programación LabVIEW que se vera más adelante.

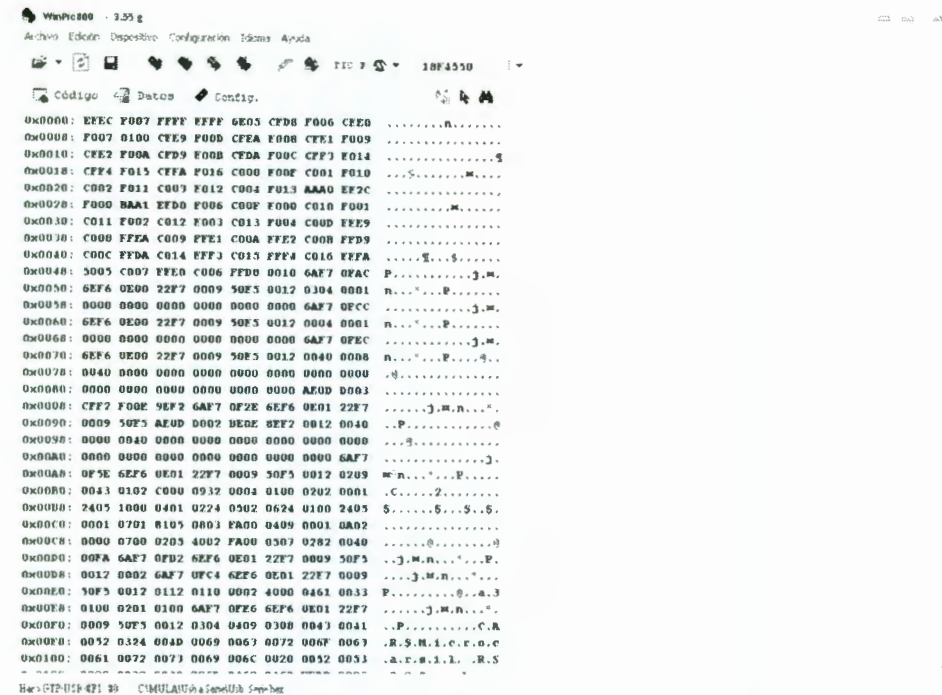


Figura 4.18
WinPic800

Otra manera de grabar el archivo hexadecimal generado por el PCW C Compiler es a través de un bootloader, el cual es un pequeño programa que se carga dentro de un intervalo de la memoria del microcontrolador y sirve para transferir programas desarrollados por medio del bus USB sin la necesidad de un programador externo. Esto ofrece la comodidad a la hora de desarrollar ya que se compila y se manda inmediatamente el programa generado por medio del USB, todo esto sin sacar el micro de la tarjeta de control y adquisición de datos. Ese archivo bootloader generalmente lo proporciona la compañía fabricante de los microcontroladores. Una vez grabado el firmware bootloader en el microcontrolador se tendrán que agregar 2 instrucciones en los programas que deseemos cargarle, es decir, que antes de compilar el programa principal debemos agregarle las siguientes instrucciones:

```
#build(reset=0x800, interrupt=0x808)
#org 0x000, 0x7ff { }
```

Quedando ubicadas esas instrucciones en las siguientes posiciones 5 y 6 del programa principal:

```

1) #include <18F4550.h>
2) #fuses HSPLL, NOWDT, NOLVP, NODEBUG, USBDIV, PLL5, CPUDIV1, VREGEN
3) #use delay(clock=48000000)
4) #use fast_io(B)
5) #build(reset=0x800, interrupt=0x808)
6) #org 0x000, 0x7ff { }
7) #DEFINE USB_HID_DEVICE FALSE
8) #define USB_EP1_TX_ENABLE USB_ENABLE_BULK
9) #define USB_EP1_RX_ENABLE USB_ENABLE_BULK
10) #define USB_EP1_TX_SIZE 1
11) #define USB_EP1_RX_SIZE 1
12) #include <libreria_Usb.h>

```

Al construir la aplicación esas 2 instrucciones le dicen al compilador que guarde el programa a partir de la dirección 0x0400 hexadecimal, ya que el bootloader se encuentra ubicado en el intervalo de direcciones que abarca de la 0x0000 a 0x03E8. Ahora que se tiene el archivo hexadecimal de la aplicación se utilizará el software PICDEM como se muestra en la figura 4.19, el software reconocerá a la tarjeta de control y adquisición de datos USB al momento de conectarla gracias al bootloader grabado anteriormente, el software permite cargar, borrar o verificar el firmware que se implementará en el microcontrolador. Los beneficios de utilizar esa técnica de programación son variados: No necesitas programador y por lo mismo no retiras tu microcontrolador de la tarjeta, es rápido por ser de tecnología USB y ahorra tiempo de programación.



Figura 4.19
PICDEM

b) La herramienta de programación y simulación Matlab utilizada para este trabajo fue sin lugar a dudas muy provechosa, porque permitió utilizar una interfaz grafica y un lenguaje muy intuitivo de tal manera que cualquier cambio que se deseaba hacer a los coeficientes del algoritmo de control adaptable se hacían sin ningún problema, es decir, de manera rápida y fácil; a diferencia de la plataforma en visual C++ cuya programación fue más complicada por el hecho de que Matlab posee funciones matriciales nativas al lenguaje y toolbox de interfase gráfica cosa que visual C++ no posee, de esta manera las simulaciones en esta plataforma de matlab fueron exitosas y permitieron hacer un análisis muy profundo del algoritmo de control adaptable. En la figura 4.21 se aprecia la interfaz de programación de Matlab a través de un editor M-File en el cual se escribe todo el código de la aplicación que se va a simular, en este caso la programación del control adaptable, la planta, la interfaz grafica, etc.

```

Editor - C:\MATHESS\Adaptable\ADAPTABLE\Visual C++ - Adaptable para Fic1B\439 con cambio de planta\adaptable.m
File Edit Text Cell Tools Debug Desktop Window Help
[Icons] Stack

22 - end
23 - Ft(1,1)=0; Ft(1,2)=0; Ft(1,3)=0; Ft(1,4)=10;
24 - Ft(2,1)=0; Ft(2,2)=0; Ft(2,3)=0; Ft(2,4)=10;
25 - Ft(3,1)=0; Ft(3,2)=0; Ft(3,3)=0; Ft(3,4)=10;
26 - Ft(4,1)=0; Ft(4,2)=0; Ft(4,3)=0; Ft(4,4)=10;
27 - if(Teta_reales== 0)
28 - Te_1(1,1)=0.025;
29 - Te_1(2,1)=0.0204;
30 - Te_1(3,1)=0.08;
31 - Te_1(4,1)=-0.502;
32 - else
33 - Te_1(1,1)=0.03;
34 - Te_1(2,1)=0.03;
35 - Te_1(3,1)=1;
36 - Te_1(4,1)=-0.6;
37 - end
38 - while(Ecmuestras)
39 - Ym1=(0.2*Ym)+(0.8*E);
40 - if(k<25)
41 - Y=(-0.502*Y_2)+(1.46*Y_1)+(0.0204*E_2)+(0.025*E_1);
42 - else
43 - Y=(-0.502*Y_2)+(1.46*Y_1)+((Perturbacion*0.0204)*E_2)+((Perturbacion*0.025)*E_1);
44 - end
45 - Ft1 = (operacion1(Ft,F1))/lamda;
46 - Te = operacion2(Ft,F1,Te_1,Y,Y_1);
47 - U=(1/Te(1,1))*(Ym1-(0.5*Ym)-(Te(3,1)*Y)-(Te(4,1)*Y_1)-(Te(2,1)*U_1));
48 - F1(1,1)=U;
49 - F1(2,1)=U_1;
50 - F1(3,1)=Y;
51 - F1(4,1)=Y_1;
52 - Ft=Ft1;
53 - Te_1=Te;
54 - Yg(k)=Y;

```

Figura 4.21
Interfase Matlab

4.3.3 Interfaz grafica

La primera decisión a la hora de desarrollar la aplicación con interfaz gráfica para utilizar la tarjeta de control y adquisición de datos USB fue la selección de los siguientes aspectos a considerar:

- El lenguaje de programación a usar
- El toolkit gráfico (librería para el manejo de ventanas y creación de la interfaz gráfica)
- Las librerías a utilizar para controlar diversos aspectos de la aplicación
- El entorno de desarrollo a utilizar para crear la aplicación gráfica
- La portabilidad de la aplicación principal
- La aplicación debe ser de rápido desarrollo
- Compatibilidad con las librerías de la tarjeta

Por los puntos anteriormente comentados la decisión que se tomo fue que el lenguaje en el que se desarrollaría dicha aplicación sería la plataforma de National Instrument LabVIEW. En la figura 4.22 se aprecia en el lado izquierdo el panel frontal donde se encuentra la interfaz grafica del usuario y del lado derecho el diagrama de bloques donde se programa la aplicación.

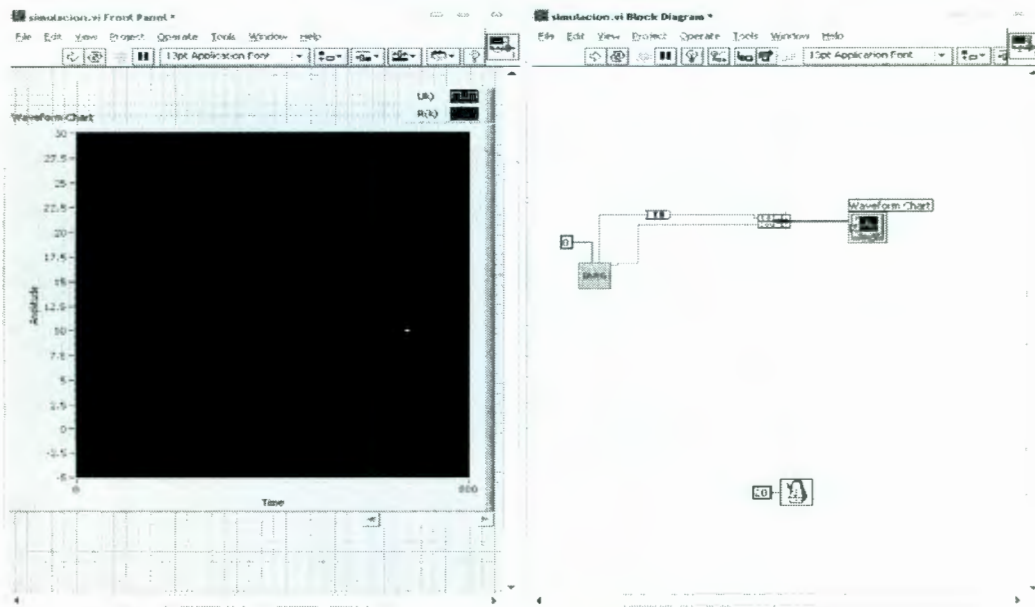


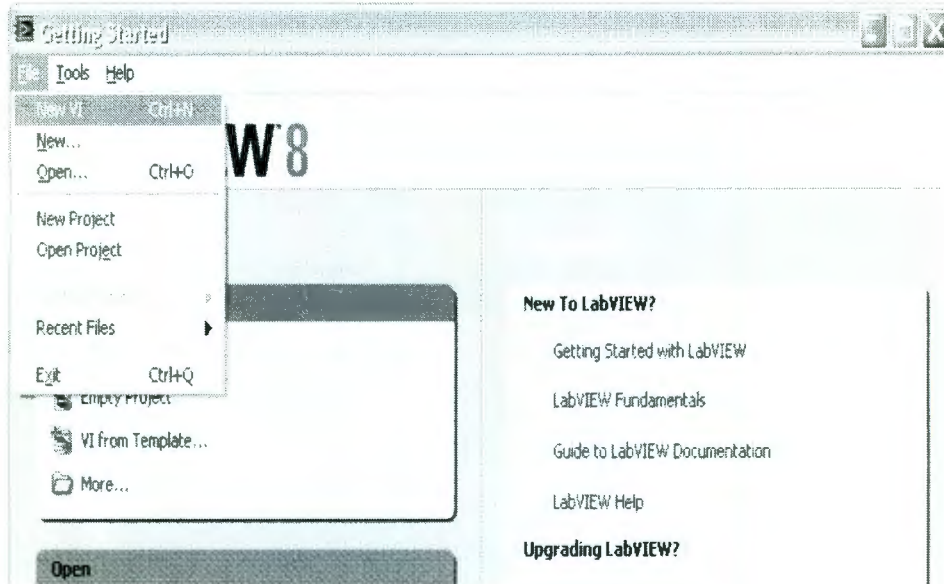
Figura 4.22
Interfaz grafica LabVIEW

Programación LabVIEW

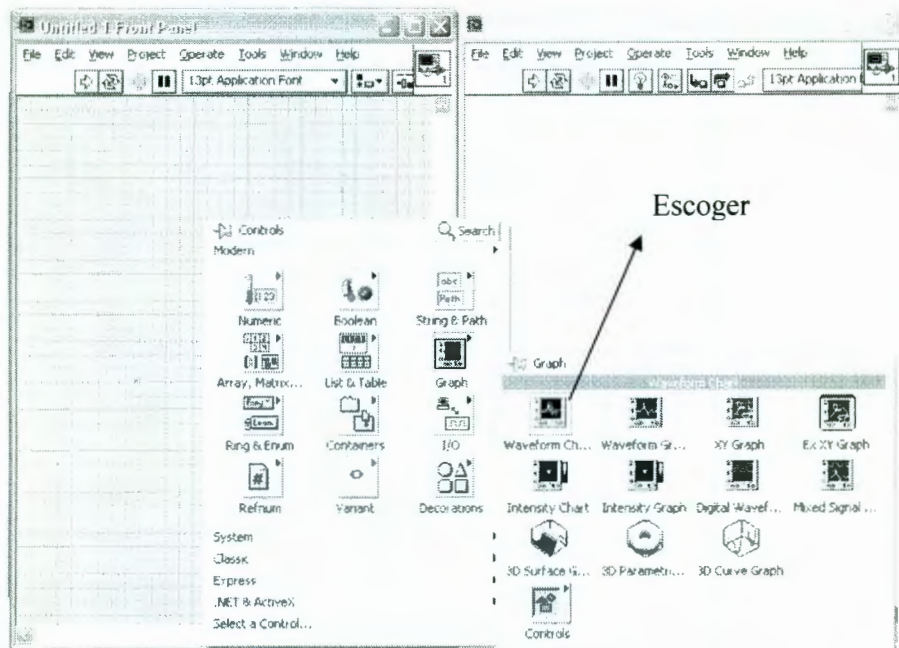
En este apartado se describirá de manera de ejemplo el uso y configuración de LabVIEW para la tarjeta de control y adquisición de datos USB utilizando 2 entrada analógicas y 1 salida digital ambas de 8 bits.

Pasos realizados:

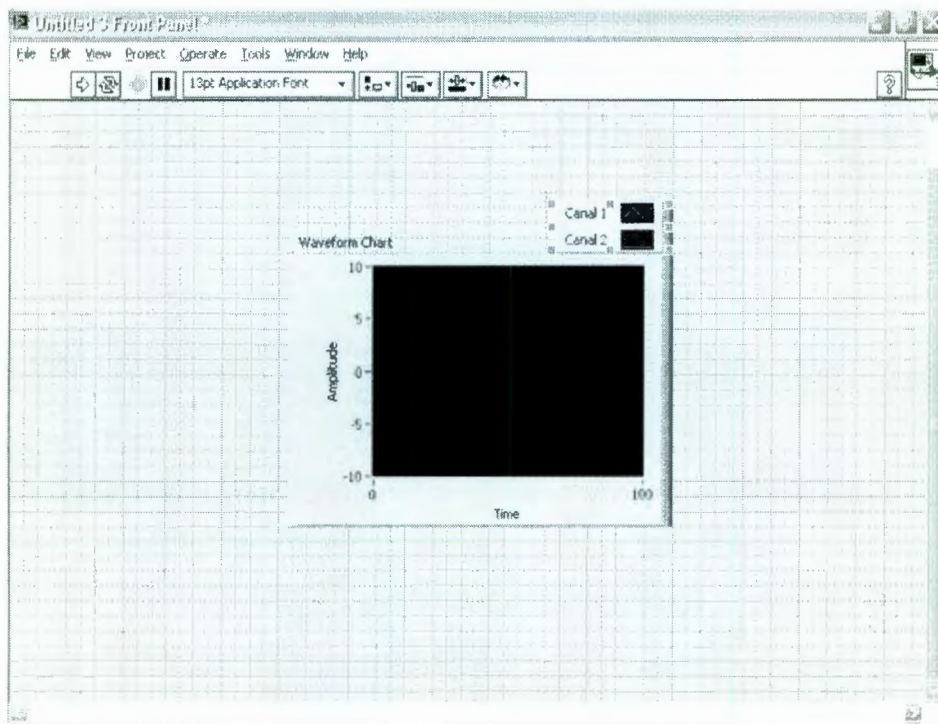
1.- Se abrió el programa LabVIEW, y posteriormente se hizo click en la ubicación **File/New VI**.



2.- En el **panel frontal** se hizo click derecho y se escogió una **Waveform chart**

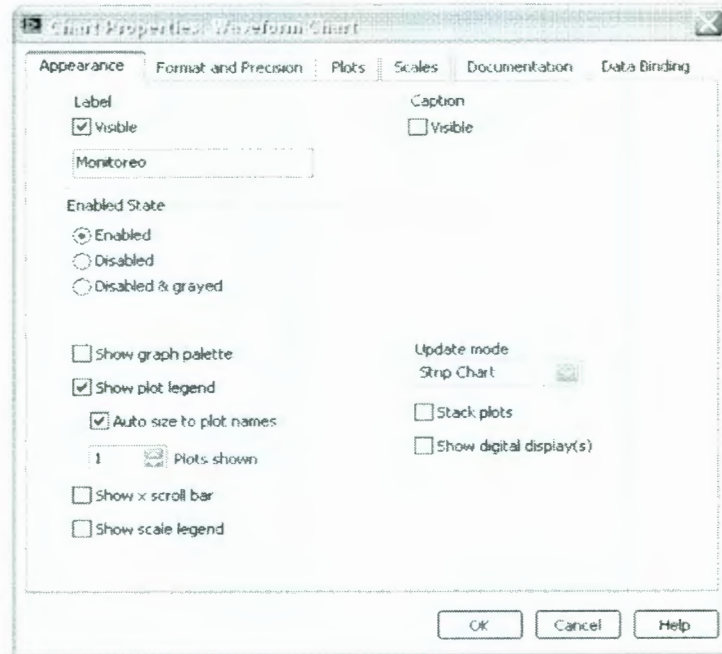


3.- Se expandió el número de canales como se aprecia en la imagen:

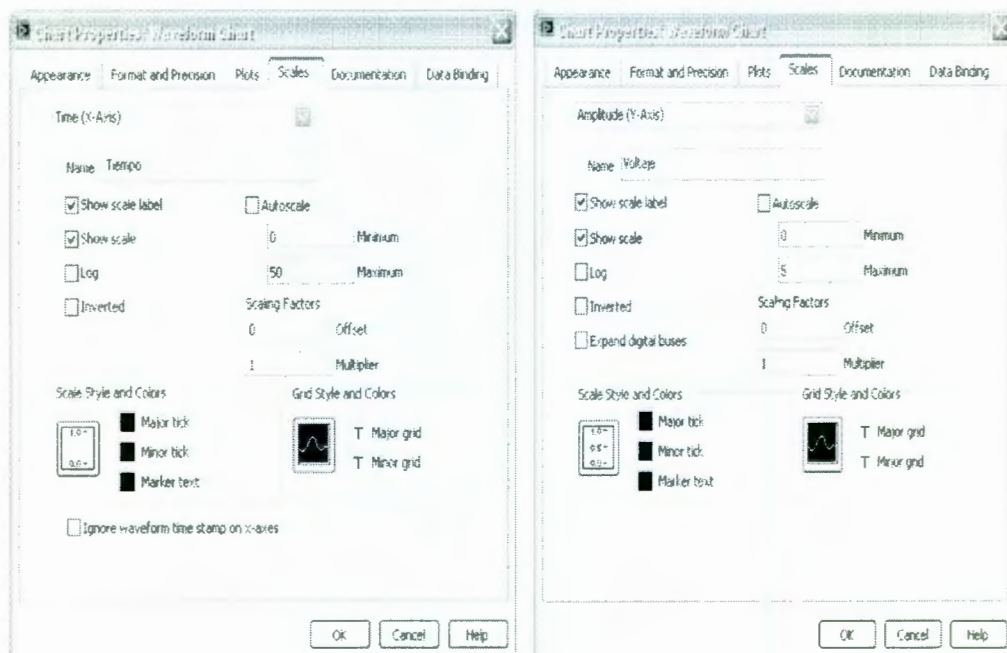


4.- Se hizo click derecho sobre la gráfica y se escogió **properties**, después se configuró lo siguiente:

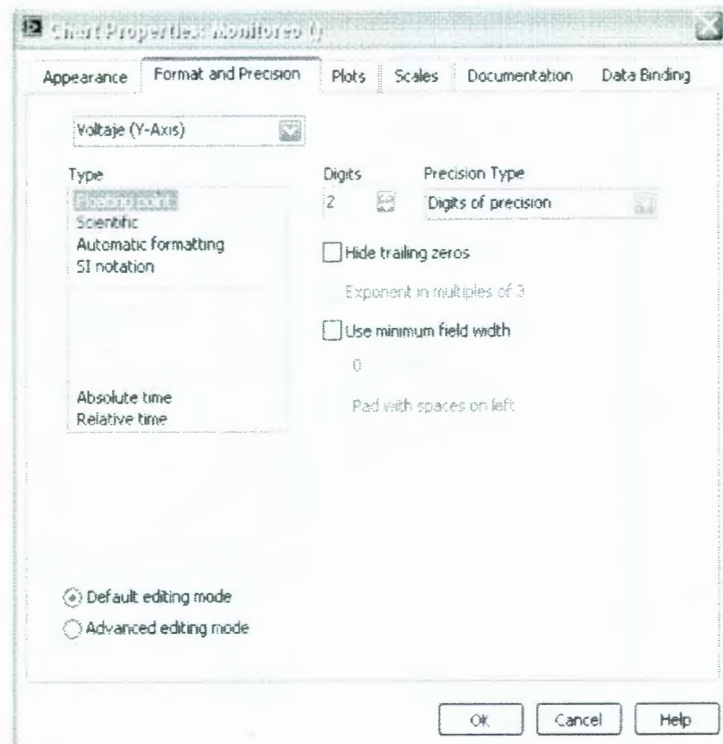
a) En la pestaña **appearance**, como sigue:



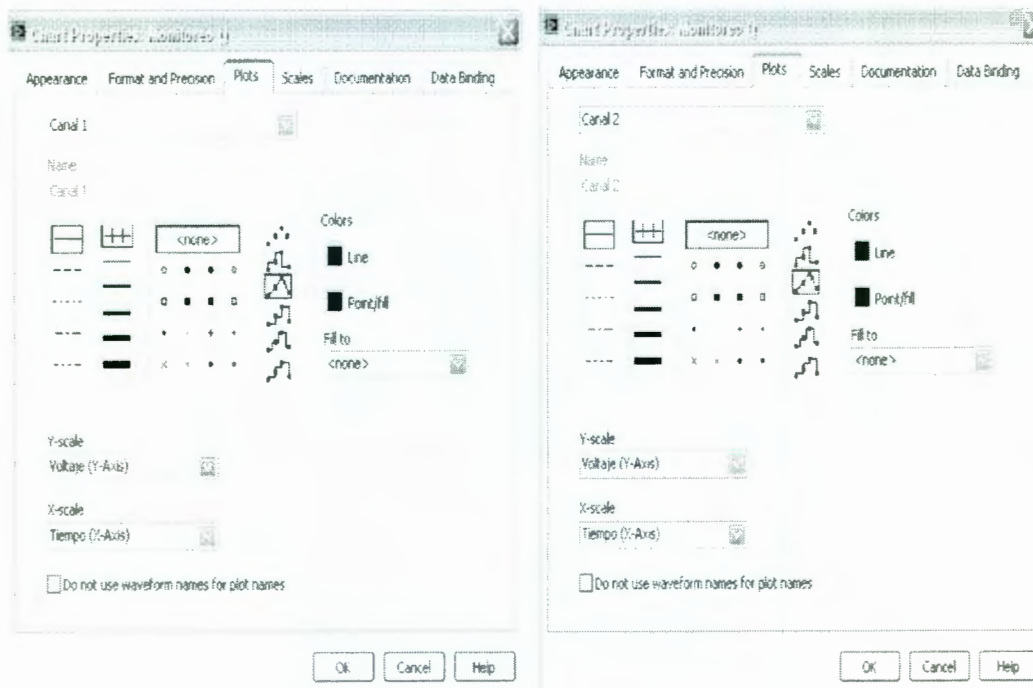
b) En la pestaña **Scales**:



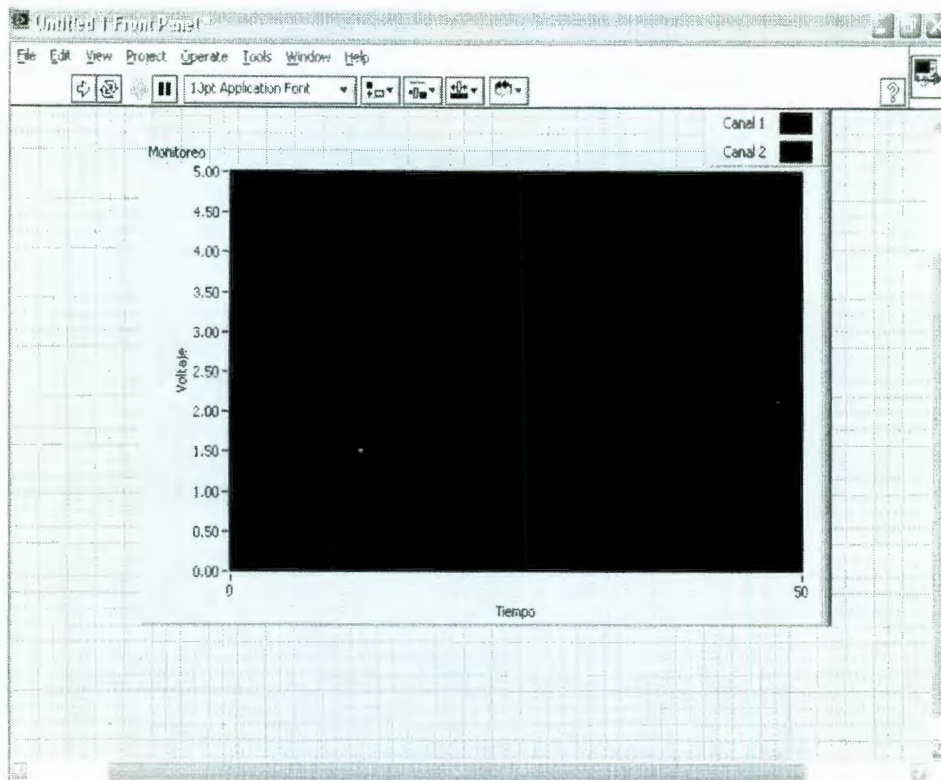
c) En la pestaña **Format and Precision** se configuró como muestra la imagen:



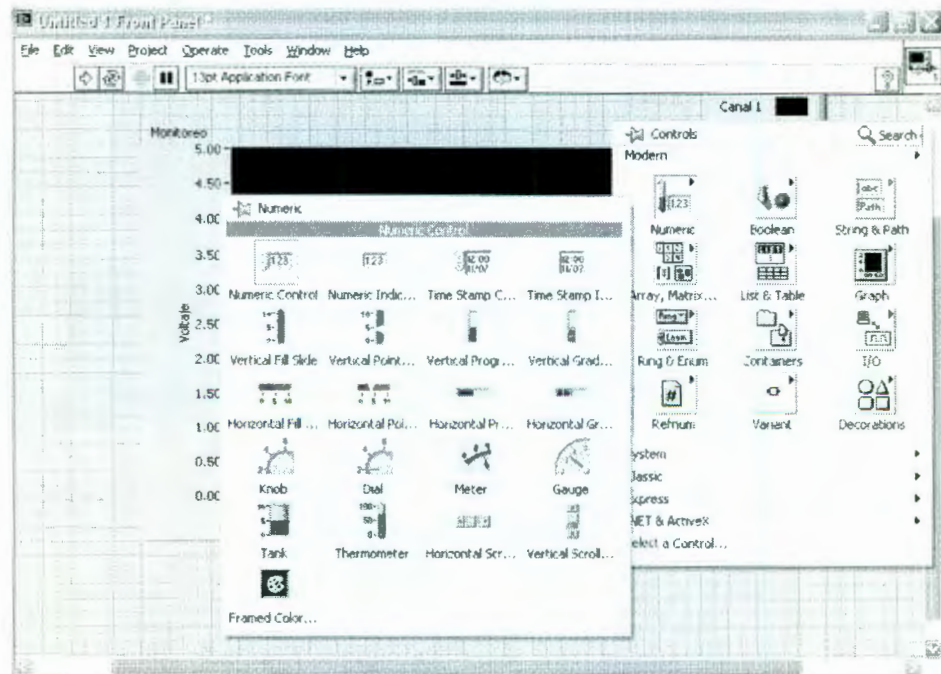
d) En la pestaña **Plots**:

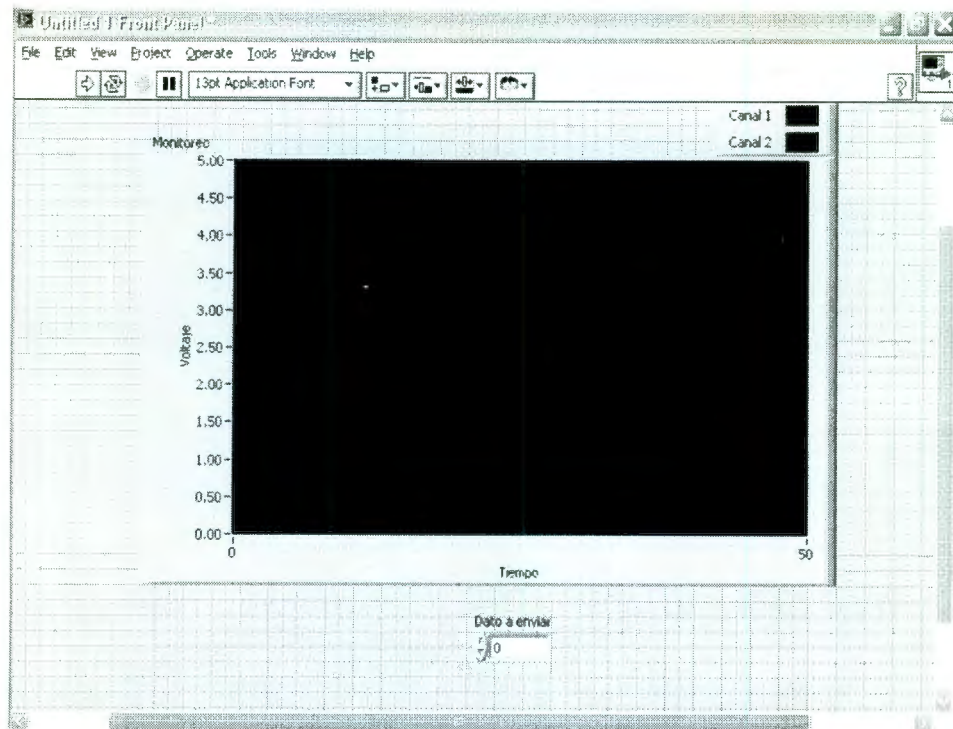


e) Se hizo click en **ok** y extendió la gráfica al gusto:

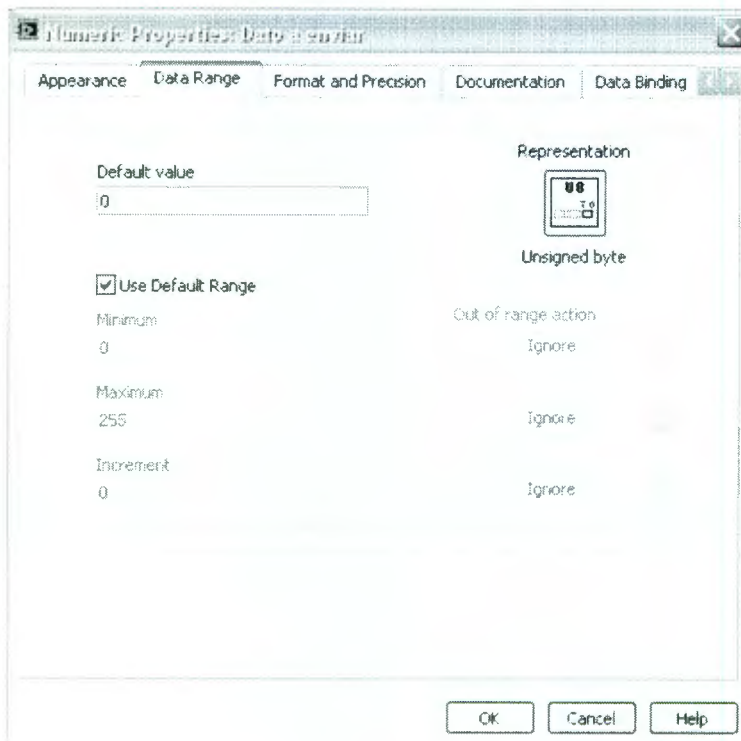


5.- En el panel frontal se hizo click derecho y se escogió un **Numeric control**:

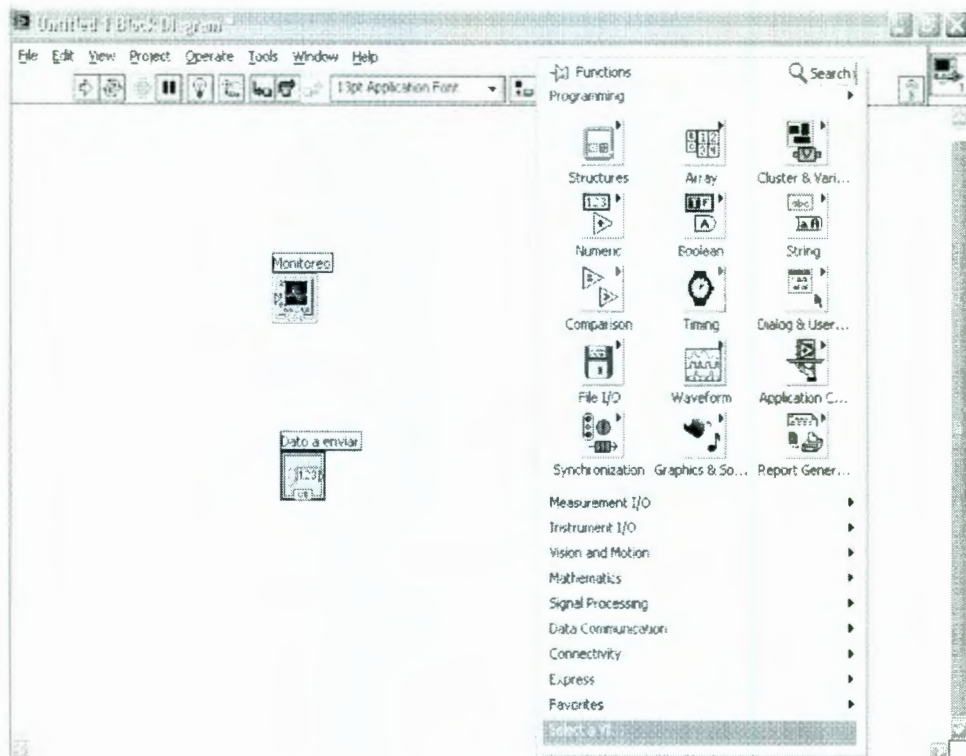




Se hizo click derecho sobre el control numérico y se escogió **properties**, para configurarse como lo muestra la imagen:



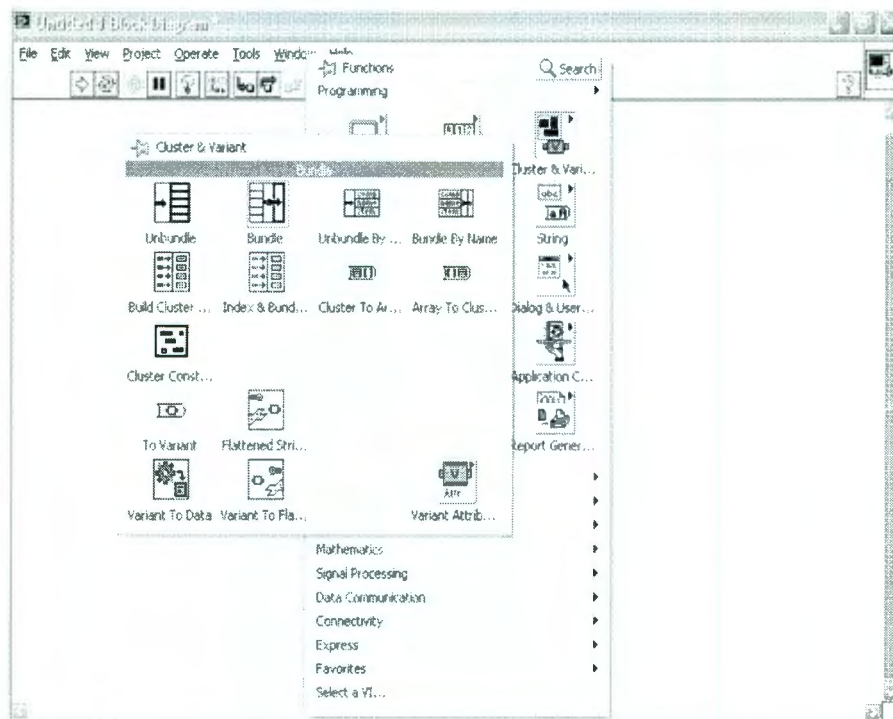
6.- En el **Diagrama de Bloques**, se hizo click derecho y se escogió un **select a VI...**



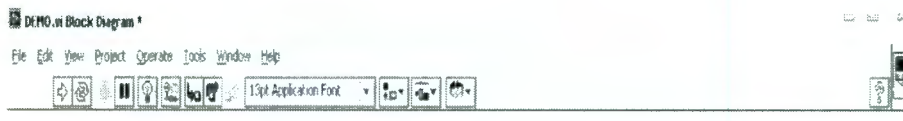
Después de ello se agrego el modulo **DUPG.vi** y se hizo click en ok.



7.- Nuevamente en el **Diagrama de Bloques**, se hizo click derecho y después click en **Cluster & Variant** para escoger un **Bundle**.



8.- Sé conectó como se muestra en la figura:

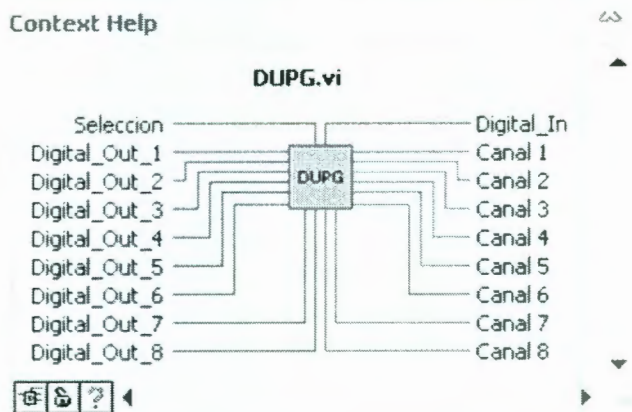


El bloque **DUPG** tiene 3 opciones para ser configurado a través de un valor constante llamado **selección**, cuyas opciones son las siguientes:

Valor 0 → Solamente puede **leer**

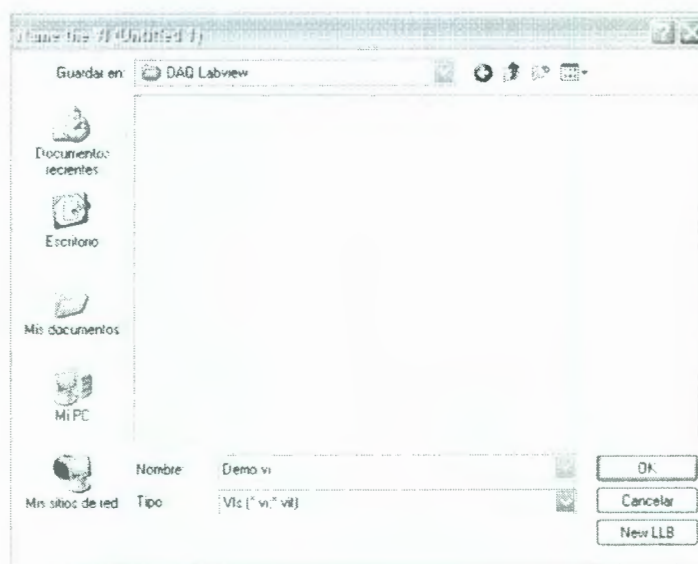
Valor 1 → Solamente puede **escribir**

Valor 2 → Puede **Leer y Escribir** al mismo tiempo



Para este caso se asignó **selección = 2**, ya que tanto se está leyendo como escribiendo.

9.- Se guardó el proyecto con extensión vi (ejemplo Demo.vi).



10.- Por último se fue al panel frontal y se corrió la aplicación.



El mismo procedimiento que se describió anteriormente para configurar la tarjeta de control y adquisición de datos USB se implementó para la programación de la interfaz gráfica ya con el algoritmo de control adaptable y la planta implementados en el microcontrolador de la tarjeta.

Para concluir este capítulo decir que las herramientas tanto en hardware como de software utilizadas durante este trabajo fueron seleccionadas bajo criterios actuales y de alta tecnología que permitieron llevar a buen puerto los alcances de los objetivos propuesto en el capítulo 1 y dan la pauta para resolver problemas de ingeniería donde involucren sistemas de control con interfases de comunicaciones rápidas, confiables y que pueden interactuar conjuntamente con sistemas embebidos con microcontroladores.

CAPÍTULO V

DESEMPEÑO DEL ALGORITMO DE CONTROL

DESEMPEÑO DEL ALGORITMO DE CONTROL

5.1 Modelo de la planta

Para la elaboración e implantación del algoritmo de control adaptable se utilizo el modelo de un sistema de control de temperatura de segundo orden como lo muestra la figura 5.1.

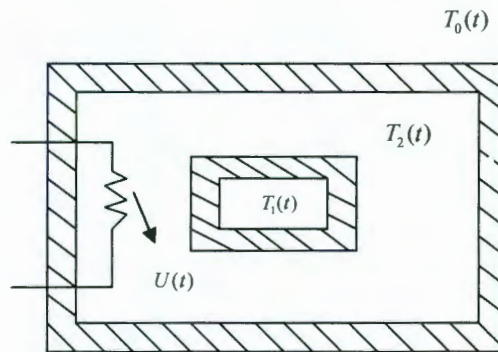


Figura 5.1
Sistema de temperatura

Las ecuaciones de la dinámica del sistema están dadas por:

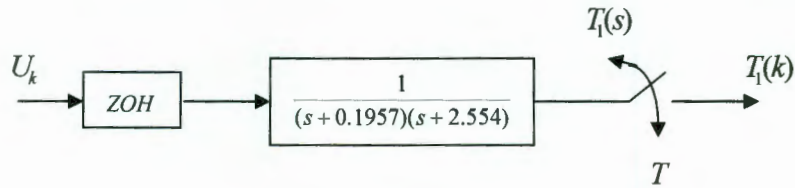
$$\frac{dT_1}{dt} = 2(T_2 - T_1) \quad (5.1.1)$$

$$\frac{dT_2}{dt} = -0.75T_2 + 0.5T_1 + kT_0 + 0.5u(t) \quad (5.1.2)$$

Resolviendo para T_1 y considerando $T_0 = 0$, nos queda la siguiente función de transferencia:

$$\frac{T_1(s)}{U(s)} = \frac{1}{s^2 + 2.75s + 0.5} \quad (5.1.3)$$

Ahora utilizando un muestreador de orden cero pasamos el sistema a la forma discreta con un periodo de muestreo de $T= 0.25$ segundos porque la constante de tiempo del polo dominante es $\frac{1}{0.1957} = 5.1$ seg.



$$\frac{G(s)}{s} = \frac{1}{s(s+0.1957)(s+2.554)} \quad (5.1.4)$$

$$\frac{G(s)}{s} = \frac{2}{s} - \frac{2.1667}{s+0.1957} + \frac{0.1667}{s+2.554} \quad (5.1.5)$$

$$Z\zeta^{-1} \left[\frac{G(s)}{s} \right] = \frac{2}{1-z^{-1}} - \frac{2.1667}{1-e^{-0.1957T}z^{-1}} + \frac{0.1667}{1-e^{-2.554T}z^{-1}} \quad (5.1.6)$$

$$G(z) = (1-z^{-1}) \left[\frac{\alpha z^{-1} + \beta z^{-2}}{(1-z^{-1})(1-e^{-0.1957T}z^{-1})(1-e^{-2.554T}z^{-1})} \right] \quad (5.1.7)$$

Donde:

$$\alpha = -2(e^{-0.1957T} + e^{-2.554T}) + 2.1667(1+e^{-2.554T}) - 0.1667(1+e^{-0.1957T})$$

$$\beta = 2e^{-0.1957T}e^{-2.554T} - 2.1667e^{-2.554T} + 0.1667e^{-0.1957T}$$

Obteniendo la siguiente función de transferencia en z:

$$G(z) = \frac{T_1(z)}{U(z)} = \frac{0.025z^{-1}(1+0.816z^{-1})}{(1-0.952z^{-1})(1-0.528z^{-1})} \quad (5.1.8)$$

5.2 Aplicación del controlador lineal de seguimiento y regulación

Desarrollamos la ecuación 5.1.8 a la forma del modelo de la ecuación 3.1.1 quedando de la siguiente forma:

$$\frac{Y(k)}{U(k)} = \frac{q^{-1}(0.025 + 0.0204q^{-1})}{(1 - 1.48q^{-1} + 0.502q^{-2})} \quad (5.2.1)$$

Donde:

$$b_0 = 0.025, \quad b_1 = 0.0204, \quad a_1 = -1.48, \quad a_2 = 0.502$$

Se propone el modelo de referencia:

$$\frac{Y^M(k)}{U^M(k)} = \frac{q^{-1}(0.8)}{1 - 0.2q^{-1}} \quad (5.2.2)$$

Porque tiene un polo en $z = 0.2$, lo que da una respuesta relativamente rápida y se propone el filtro estable C_r :

$$C_r(q^{-1}) = 1 - 0.5q^{-1} \quad (5.2.3)$$

Por tener un cero en $z=0.5$ lo que da un seguimiento amortiguado.

De donde se calculan las siguientes constantes:

$$d = 1, \quad n_A = 2, \quad n_B = 1, \quad n_{C_2} = 1$$

$$n_S = d - 1 = 0, \quad n_R = \max(n_A - 1, n_{C_2} - 1) = 1$$

Resolviendo la ecuación 3.1.9 obtenemos r_0 y r_1 :

$$r_0 = 0.98, \quad r_1 = -0.502$$

$$R(q^{-1}) = 0.98 - 0.502q^{-1} \quad (5.2.4)$$

A partir de las ecuaciones 3.1.21 y 3.1.22 podemos calcular la ecuación de control, quedando de la siguiente forma:

$$U(k) = 40Y^M(k+1) - 20Y^M(k) - 39.2Y(k) + 20.08Y(k-1) - 0.816U(k-1) \quad (5.2.5)$$

Desarrollando las ecuaciones 5.2.1 y 5.2.2:

$$Y(k) = -0.502Y(k-2) + 1.48Y(k-1) + 0.0204U(k-2) + 0.025U(k-1) \quad (5.2.6)$$

$$Y^M(k) = 0.2Y^M(k-1) + 0.8U^M(k-1) \quad (5.2.7)$$

El esquema completo para el control de la planta se puede apreciar en la figura 5.2.

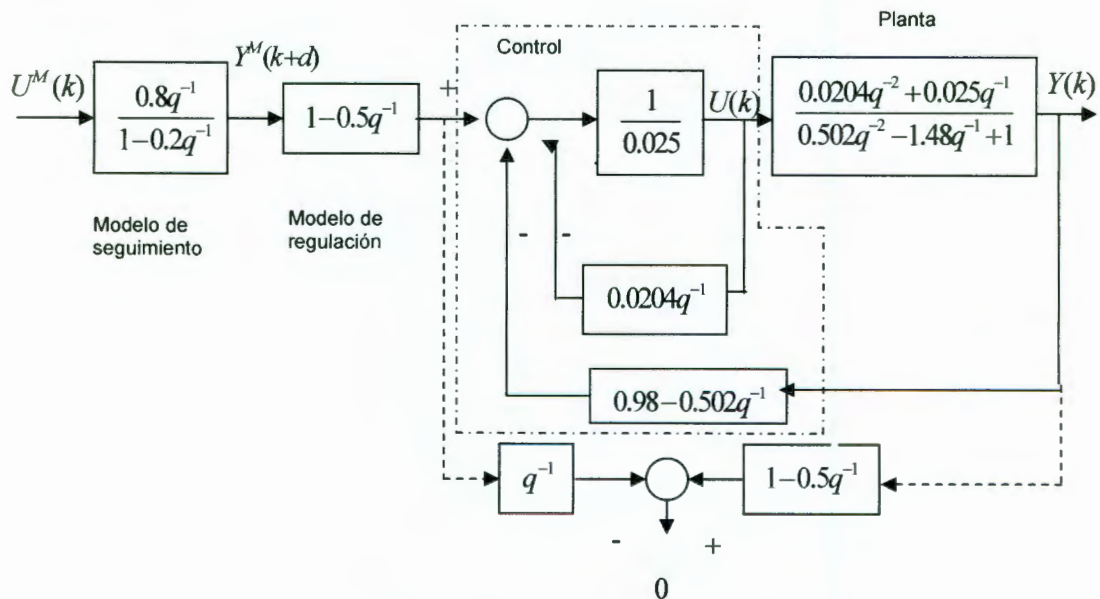
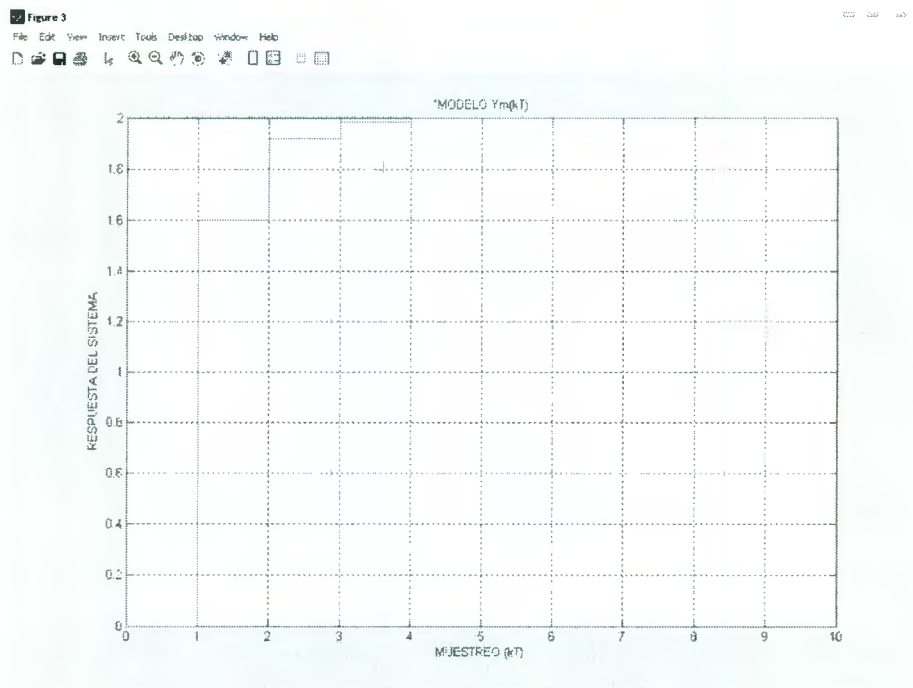


Figura 5.2
Aplicación del control de seguimiento y regulación

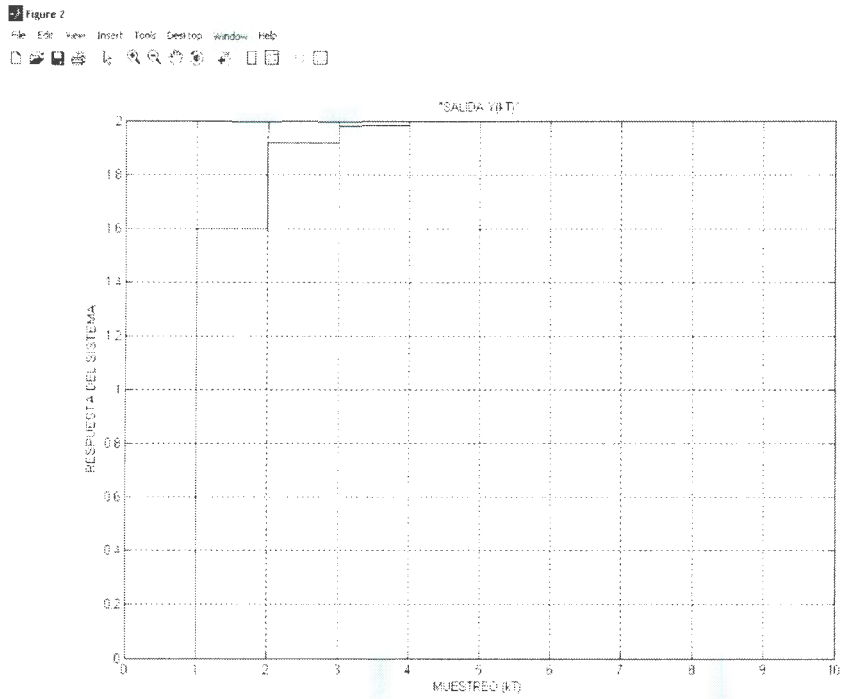
Con las ecuaciones 5.2.5, 5.2.6 y 5.2.7 se realizó un programa en matlab con la finalidad de poder simular la respuesta de la planta y la señal de control con el objetivo de conocer como respondían antes de la implementación en el microcontrolador. Cabe destacar que se utilizo como referencia del modelo de seguimiento un valor $U^M(k)=2$.

Se puede observar en la gráfica 5.2 que sigue perfectamente el modelo de referencia de la gráfica. 5.1 para cada instante de muestreo; por otro lado en la gráfica 5.3 se aprecia como la señal de control se atenúa con los instantes de muestreo hasta llegar a un valor constante de 0.97.

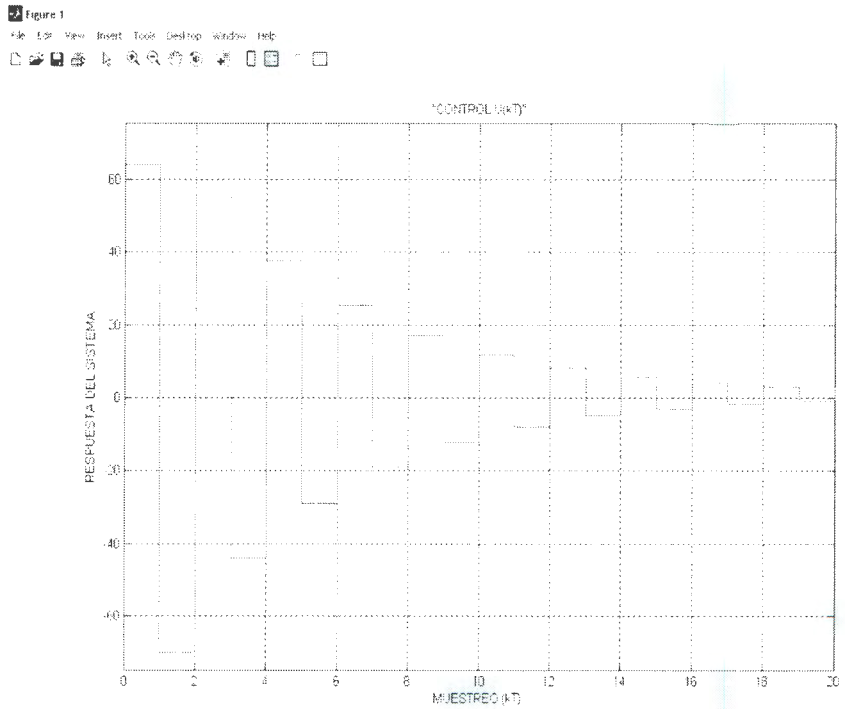


Gráfica 5.1

Salida del Modelo de Referencia para la planta ec. 5.2.7 con $U^M(k)=2$

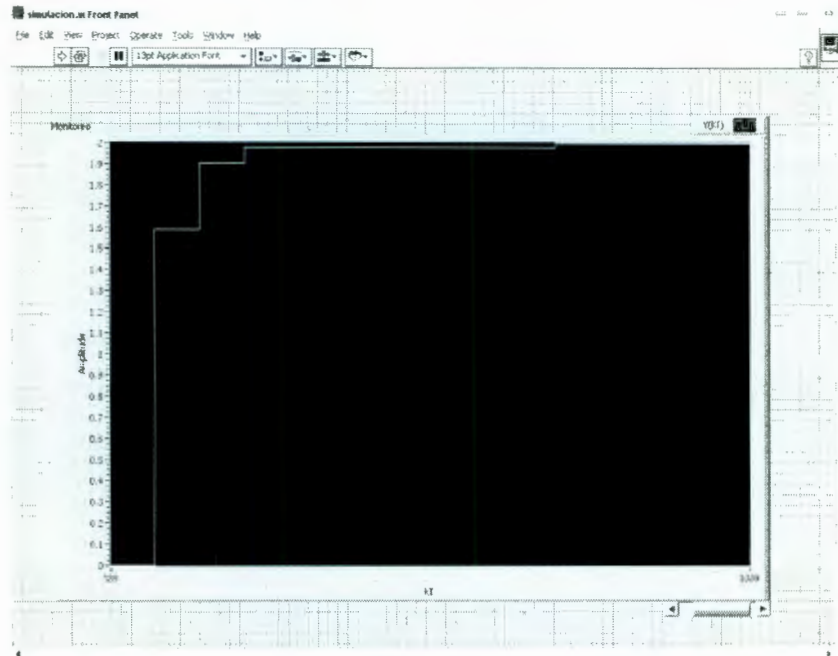


Gráfica 5.2
 Respuesta de la planta ec. 5.2.1

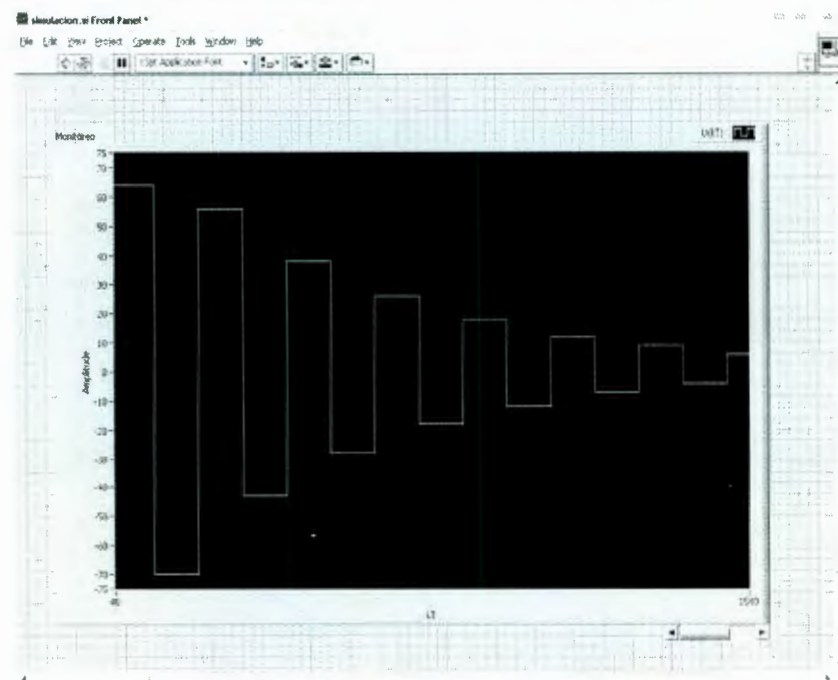


Gráfica 5.3
 Señal de control ec. 5.2.5

Las siguientes gráficas 5.4 y 5.5 son de la planta y del control respectivamente ya implementados en el microcontrolador. Como se puede apreciar son exactamente igual que las simuladas. Con esto se concluye que la implementación es correcta y con un excelente grado de precisión.



Gráfica 5.4
Respuesta de la planta implementada en tiempo real



Gráfica 5.5
Señal de control implementado en tiempo real

Desempeño del control ante una dinámica variante de la planta

Cuando se conoce el modelo de la planta y por consecuencia sus parámetros el control de seguimiento y regulación lineal (figura 5.2) visto en el capítulo 3.1 mantiene controlada a la planta de acuerdo al modelo de referencia establecido, esto se da siempre y cuando dichos parámetros no varíen en el tiempo.

Ahora se demostrará que si hay algún cambio en los parámetros de la planta el control no será lo suficientemente robusto para mantener al modelo de referencia sobre los cambios generados, como lo muestran las gráficas 5.6 y 5.7, es por esta razón que se presenta la necesidad de implementar algoritmos adaptables que estimen esos cambios en los parámetros de la planta a través de métodos recursivos de identificación.

Cambiando los parámetros de la planta:

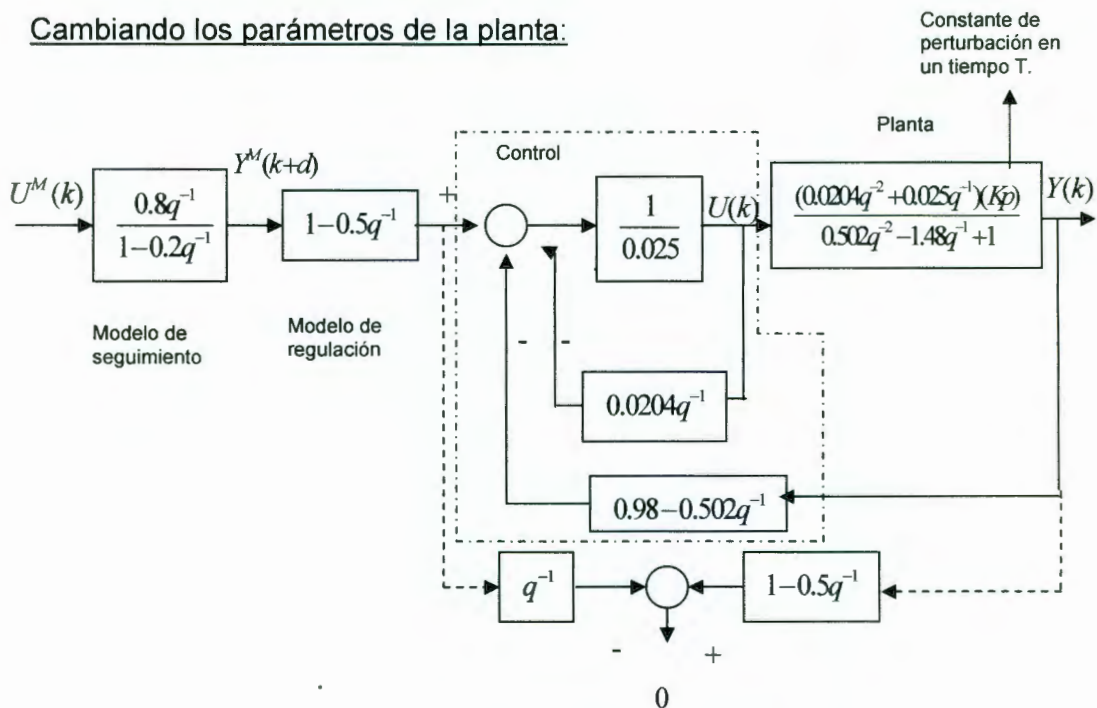


Figura 5.3
Cambiando los parámetros de la planta

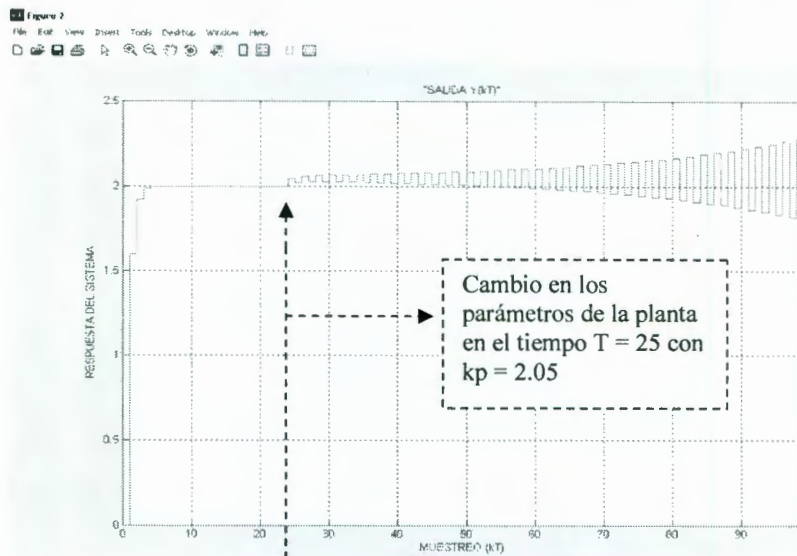
Calculando los valores donde Kp hace inestable el sistema de control y a partir de la ecuación característica de la planta de la figura 5.3:

$$q^2 - (1.48 - 0.98kp)q + (0.502 - 0.502kp) = 0$$

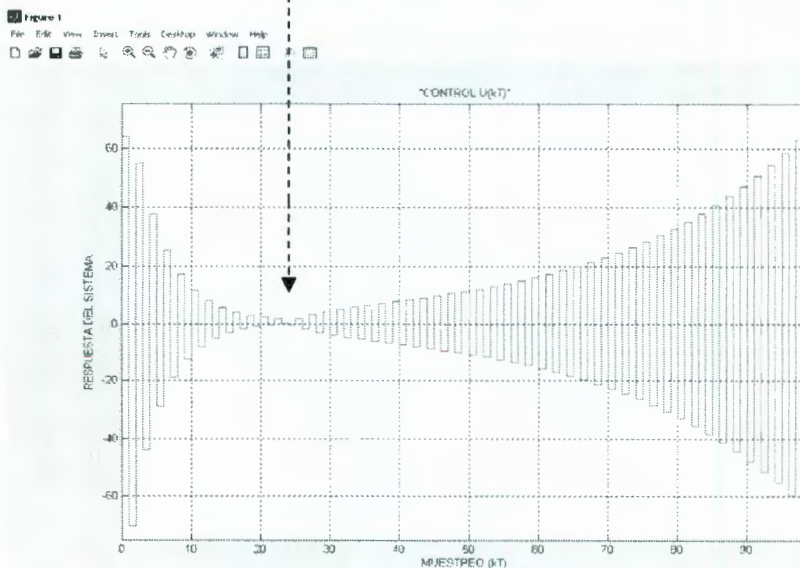
Aplicando el criterio de estabilidad de Routh para sistemas discretos se tiene que el sistema es estable para:

$$0 < k_p < 2.02$$

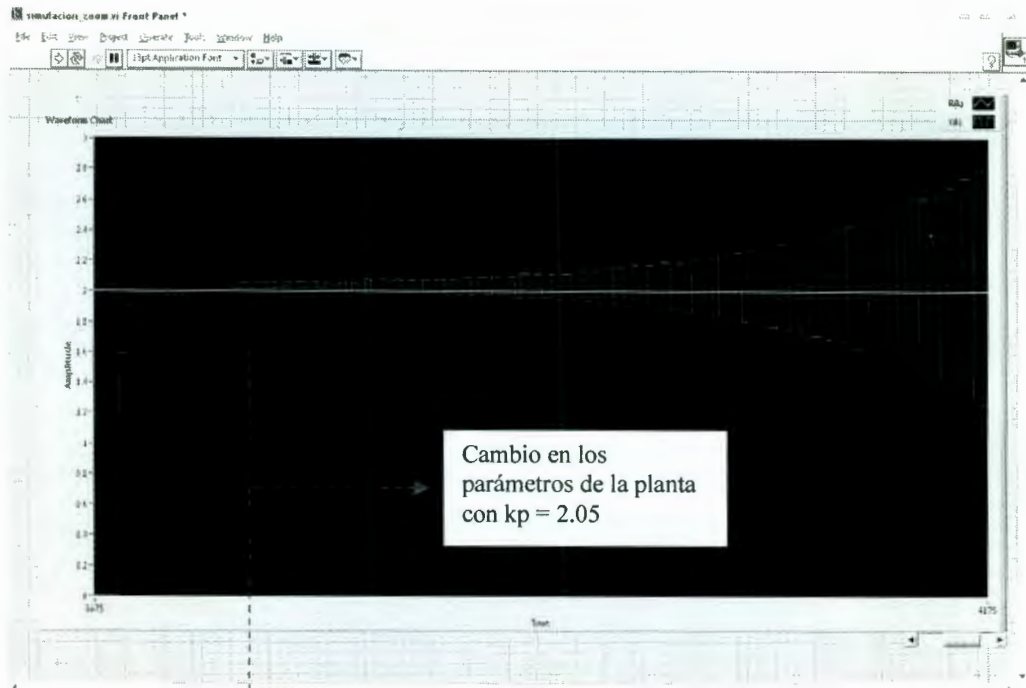
Proponiendo una $k_p = 2.05$ se tienen los siguientes resultados véase gráfica 5.6 y 5.7, donde se aprecia claramente la inestabilidad en la respuesta de la planta. En la gráfica 5.8 y 5.9 se muestra la respuesta de la planta así como la señal del control, ambas implementadas en el microcontrolador.



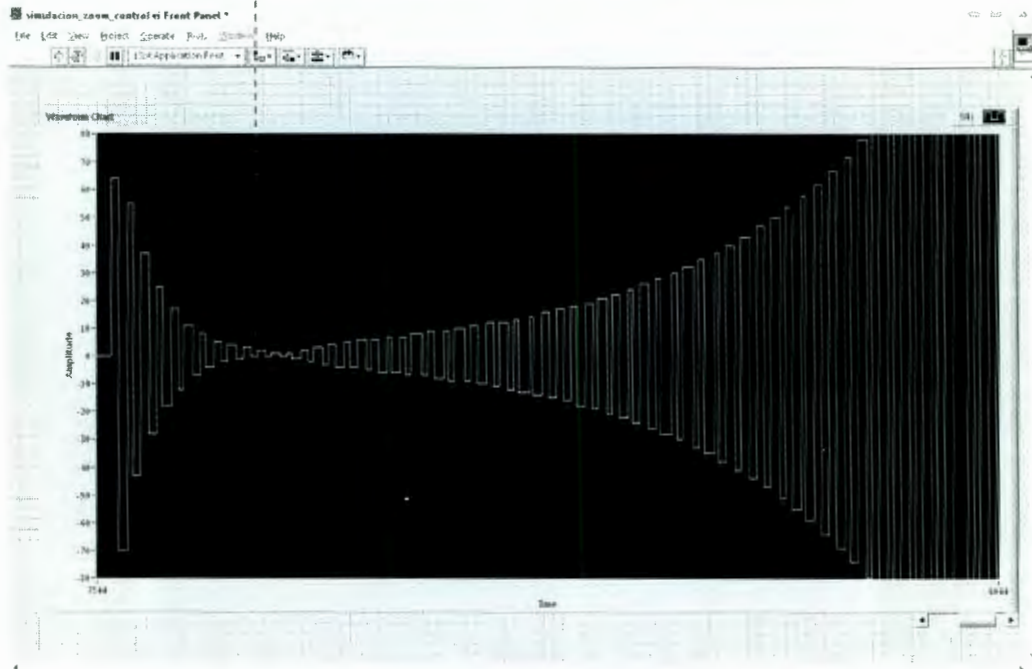
Gráfica 5.6
Dinámica difícil de la planta ec. 5.2.1 con $k_p = 2.05$



Gráfica 5.7
Señal de control ec. 5.2.5 a la dinámica difícil de la planta con $k_p = 2.05$

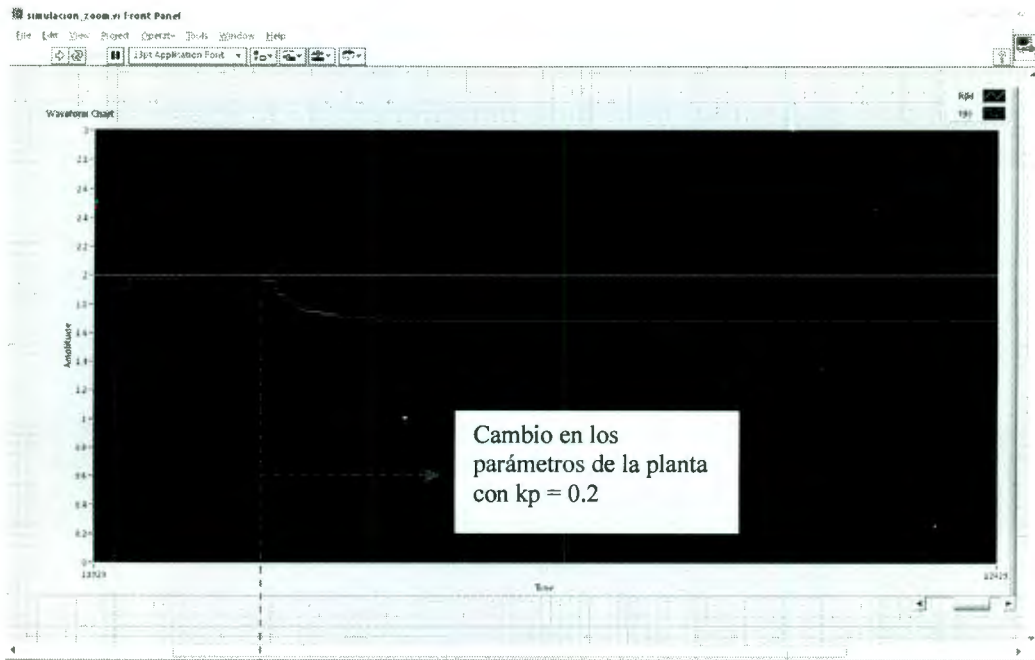


Gráfica 5.8
Dinámica difícil de la planta ec. 5.2.1 con $k_p=2.05$ en tiempo real

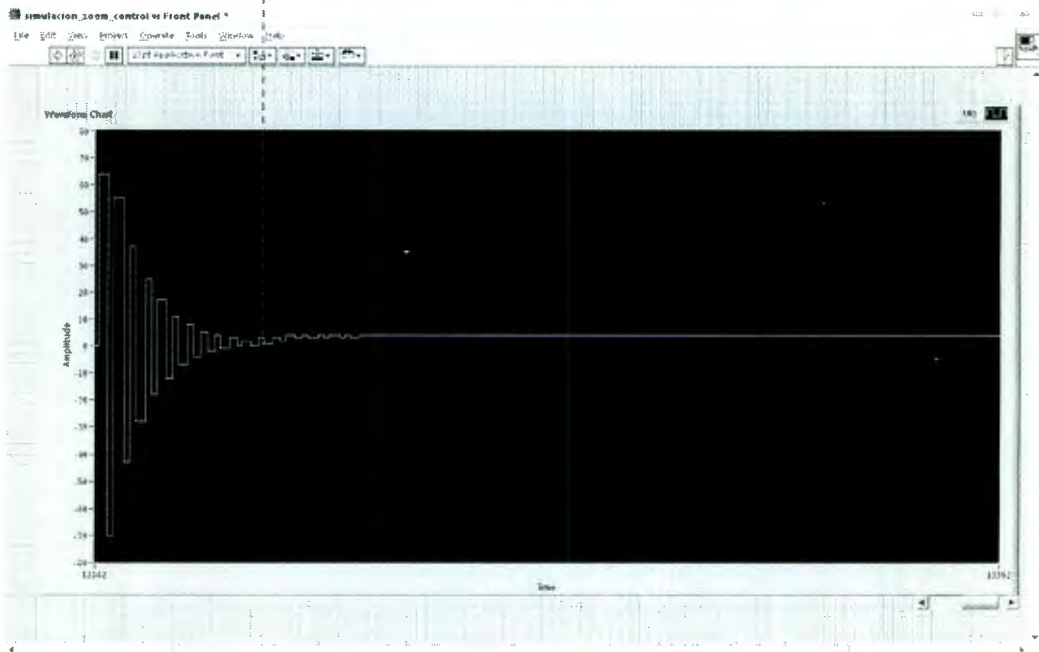


Gráfica 5.9
Señal de control ec. 5.2.5 a la dinámica difícil de la planta con $k_p=2.05$ en tiempo real

Ahora estableciendo $k_p=0.2$ se demostrará que el sistema es estable pero el control no es capaz de mantener el valor de referencia $U^M(k)=2$, véase gráficas 5.10 y 5.11.



Gráfica 5.10
Dinámica difícil de la planta ec. 5.2.1 con $k_p=0.2$ en tiempo real



Gráfica 5.11
Señal de control ec. 5.2.5 a la dinámica difícil de la planta con $k_p=0.2$ en tiempo real

Es por tal motivo que es necesaria la implementación del algoritmo adaptable propuesto en el capítulo 3 y que como veremos en el siguiente apartado resuelve favorablemente la dinámica difícil de la planta así como también perturbaciones no previstas.

5.3 Aplicación del controlador adaptable

En el tema anterior se demostró que el algoritmo de control de seguimiento y regulación funcionaba adecuadamente siempre y cuando conociéramos los coeficientes de la planta y que los mismos no cambiaran en el tiempo.

Pero ahora se parte de que no conocemos los coeficientes de la planta y que la misma cambia en un instante de tiempo t , se observará como el control compensa dichos cambios a través de los algoritmos de control vistos en el capítulo 3.

A partir de la ecuación 3.5.5 se calculan los parámetros estimados:

$$\hat{\theta}(t) = \begin{bmatrix} \hat{b}_0 \\ \hat{b}_1 \\ \hat{r}_0 \\ \hat{r}_1 \end{bmatrix}$$

Para obtener la ecuación de control para el esquema de la figura 3.3.

$$U(k) = \frac{1}{\hat{b}_0} \left[Y^M(k+1) - 0.5Y^M(k) - \hat{r}_0 Y(k) - \hat{r}_1 Y(k-1) - \hat{b}_1 U(k-1) \right] \quad (5.3.1)$$

Estimando los coeficientes de la planta

Para estimar los coeficientes de la planta se parte de la ecuación 3.5.5 que especifica todos los parámetros estimados de la misma, es importante notar que dicha ecuación necesita valores iniciales en $\hat{\theta}(t-1)$, estos se asignan de acuerdo a las características del modelo de la planta que se estudia, es decir que entre mejor se conozca a la planta obtendremos valores de parámetros mas cercanos a los reales y esto conduce a tener un mejor control en nuestro sistema, para este caso vamos a considerar que los valores iniciales de $\hat{\theta}(t-1)$ son iguales a 1, ya posteriormente se vera que entre mas cercanos se encuentren a los reales mejor será el control.

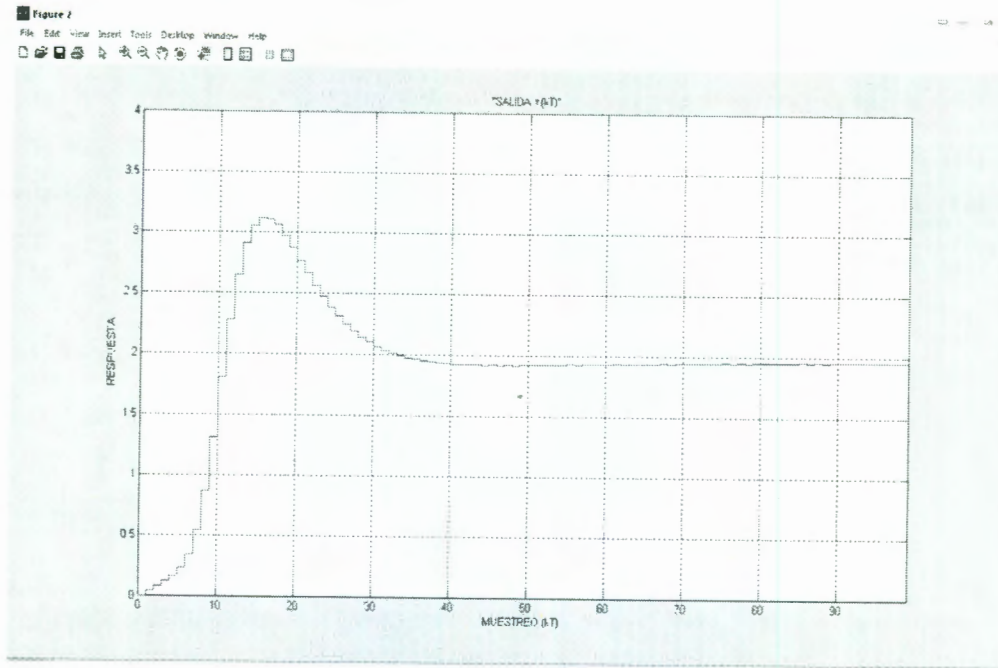
Así mismo también hay que asignarle valores iniciales a la matriz de inversión F_t de la ecuación 3.5.6, cabe mencionar que por medio de la experimentación un resultado adecuado del control se da cuando se inicializa solamente dando valores en la última columna como se aprecia en lo siguiente.

$$F_t = \begin{bmatrix} 0 & 0 & 0 & 50 \\ 0 & 0 & 0 & 50 \\ 0 & 0 & 0 & 50 \\ 0 & 0 & 0 & 50 \end{bmatrix} \quad \lambda=1.1 \quad \hat{\theta}(t-1) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

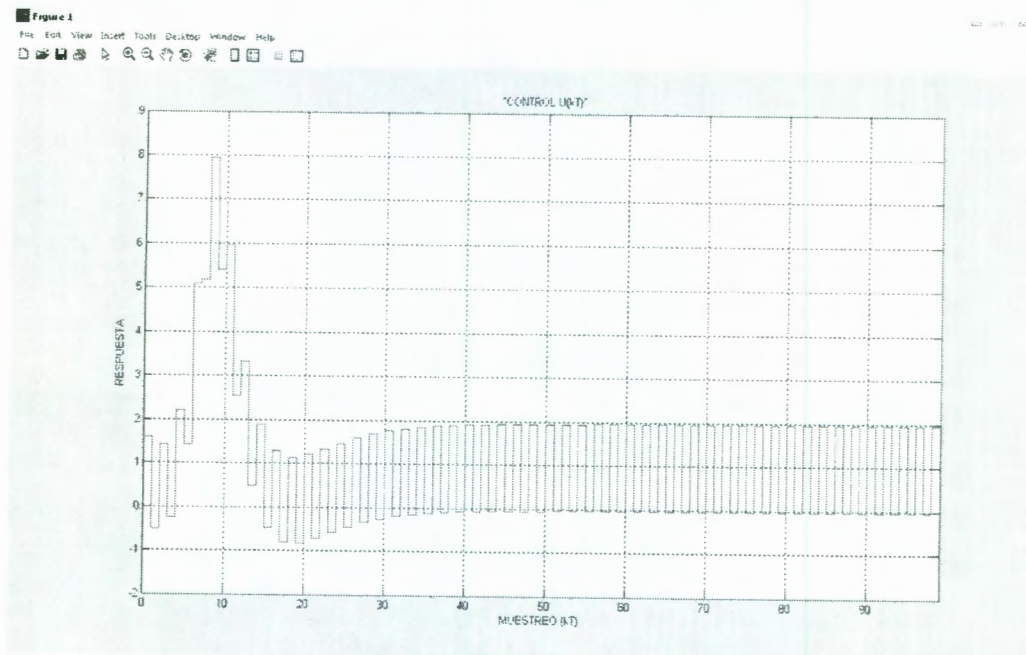
El modelo de referencia asignado se puede apreciar en la gráfica 5.1 y cuya referencia es igual a $U^M(k)=2$.

Con estas consideraciones se implementan las ecuaciones 3.5.5 y 3.5.6 en Matlab con motivo de saber con anterioridad los resultados y así poder portar las ecuaciones al microcontrolador.

Partiendo de las condiciones iniciales anteriores, se puede observar en la gráfica 5.12 que la respuesta de la planta posee un sobrepaso del 55 % en los primeros 15 muestreos, ahora bien para que la salida de la planta llegue a la referencia establecida en 2, se necesitaran cerca de 100 muestreos. Por otra parte en la gráfica 5.13 observamos que la señal de control llega hasta 8 unidades y después de ello decrementa hasta mantenerse en un periodo de oscilación constante de 0 a 2 unidades. En la gráfica 5.14 se puede apreciar la evolución de los valores estimados de $\hat{\theta}$ para la planta de la ecuación 5.2.1, es decir, los parámetros estimados b_0 , b_1 , r_0 y r_1 .

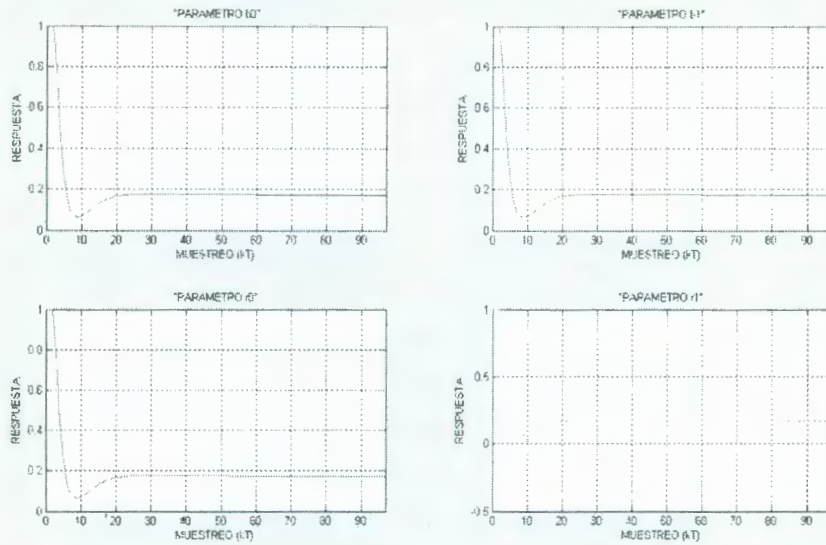


Gráfica 5.12
 Respuesta de la planta ec. 5.2.1 con controlador adaptable



Gráfica 5.13
 Señal de control adaptable ec. 5.3.1

Figure 7
 File Edit View Insert Tools Database Window Help
 [Icons]



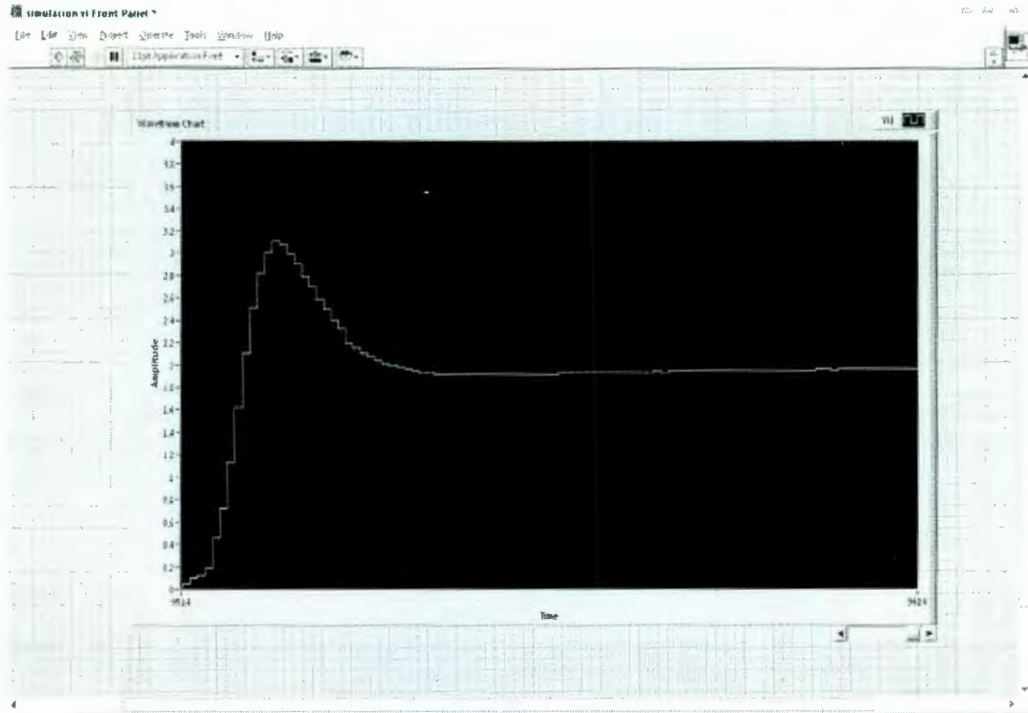
Gráfica 5.14
 Valores estimados de $\hat{\theta}$

Las siguientes gráficas 5.15 y 5.16 son de la planta y del control adaptable respectivamente implementados en el microcontrolador. Como se observa son casi iguales a las simuladas anteriormente. Con esto se puede concluir que la implementación es correcta y con un excelente grado de precisión.

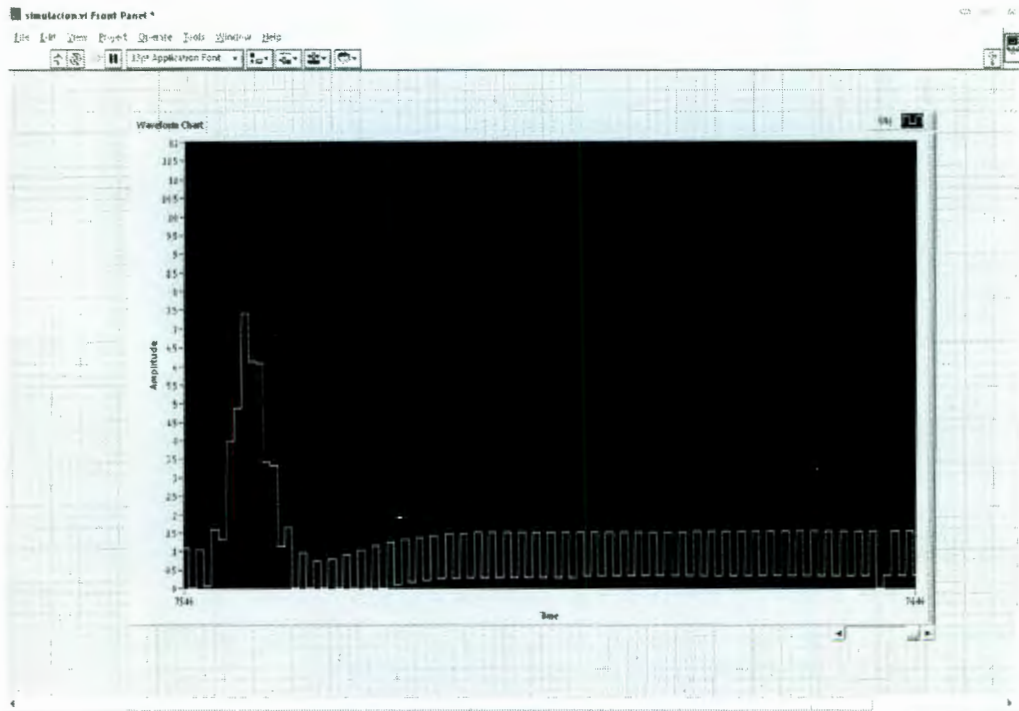
Un aspecto a considerar para la implementación elaborada en el microcontrolador y en general en las simulaciones es la inicialización tanto de la matriz de inversión F_t , la constante λ y los parámetros estimados $\hat{\theta}(t-1)$, es fundamental que estos valores se propongan en base a las siguientes 2 consideraciones:

- 1.- El sobrepaso de la respuesta de la planta
- 2.- El tiempo de establecimiento donde se llega a la referencia

Sobre ambos puntos se puede obtener un buen desempeño del sistema de control, es por ello que las simulaciones ayudan en este aspecto, porque de lo contrario al implementarlas en el microcontrolador se aprecian situaciones no deseadas como inestabilidades, sobrepasos demasiados grandes o retardos en las respuestas.



Gráfica 5.15
 Respuesta de la planta ec. 5.2.1 en tiempo real



Gráfica 5.16
 Señal de control adaptable ec. 5.3.1 en tiempo real

Desempeño del control adaptable ante una dinámica variante de la planta

Como se vio anteriormente el control lineal de seguimiento y regulación figura 5.3 no era lo suficientemente robusto para mantener el control de los cambios de la planta para $k_p > 2.02$ en un instante de tiempo t . Ahora se demostrará como el algoritmo de control adaptable figura 5.4 compensa y controla esos cambios que la planta hace en un instante de tiempo t . Tomando en base el modelo de referencia de la gráfica 5.1, se muestra la respuesta de la planta ante una variación de sus parámetros en el muestreo 25 con $k_p = 2.05$ véase gráfica 5.17, así mismo en la gráfica 5.18 se muestra la señal de control ante esos cambios generados por la planta y por último se observa la evolución de los parámetros estimados b_0, b_1, r_0 y r_1 en la gráfica 5.19.

Para esta prueba se utilizaron las siguientes condiciones iniciales en la matriz de inversión, el factor de olvido y parámetros estimados.

$$F_t = \begin{bmatrix} 50 & 0 & 0 & 50 \\ 0 & 50 & 0 & 50 \\ 0 & 0 & 50 & 50 \\ 0 & 0 & 0 & 50 \end{bmatrix} \quad \lambda = 1 \quad \hat{\theta}(t-1) = \begin{bmatrix} 0.025 \\ 0.0204 \\ 0.98 \\ -0.502 \end{bmatrix}$$

↗ Reales

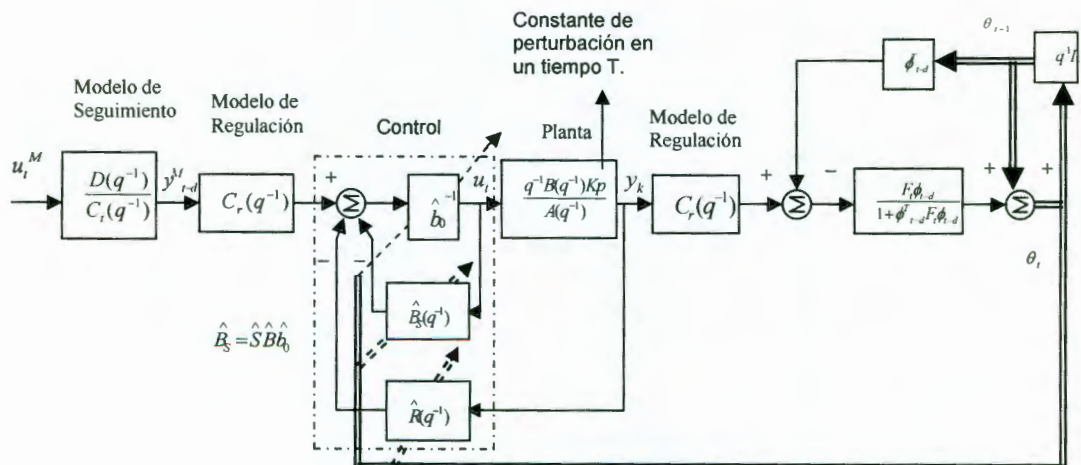
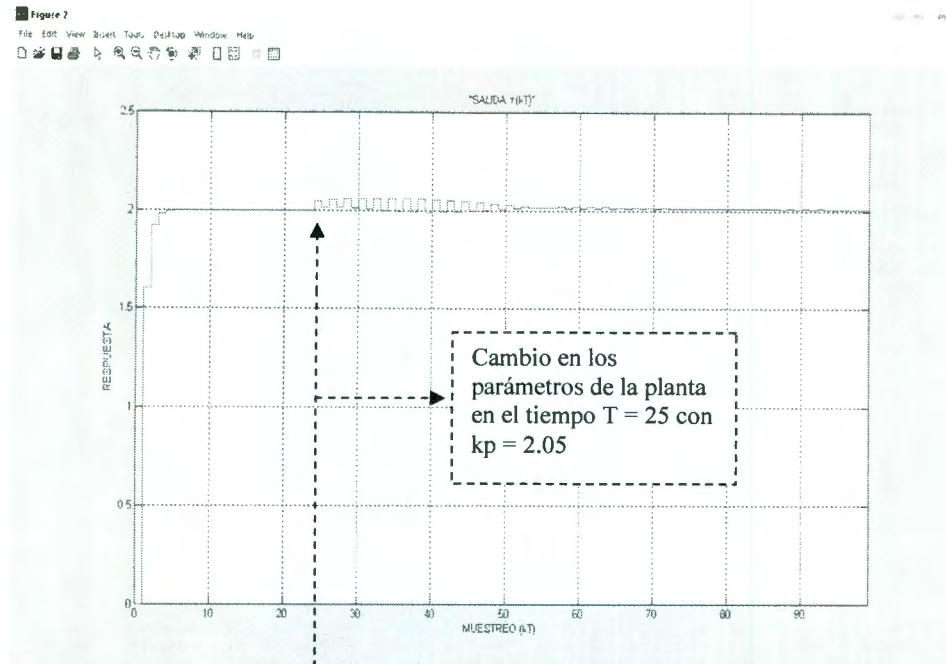
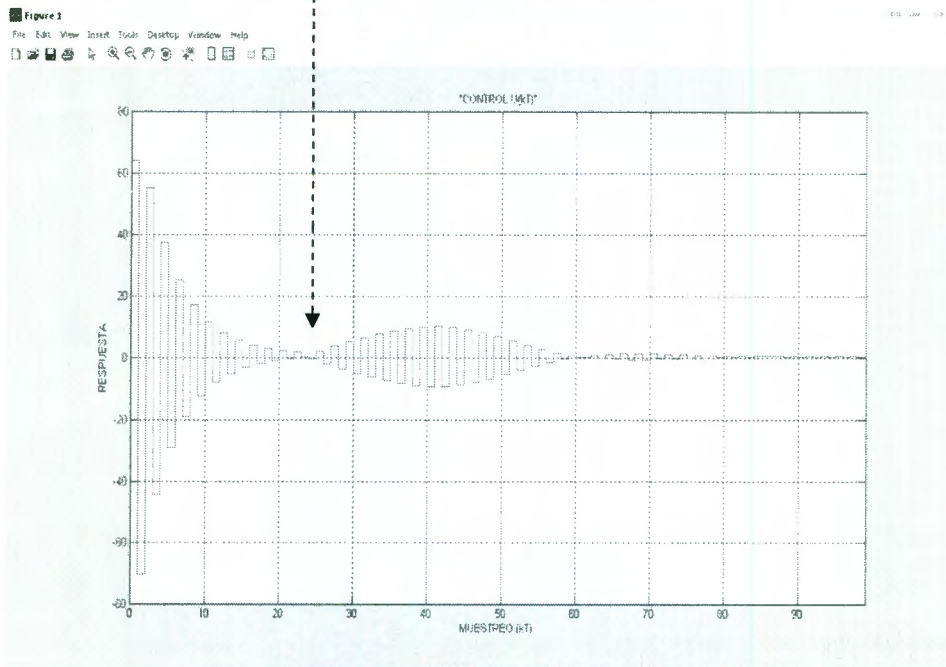


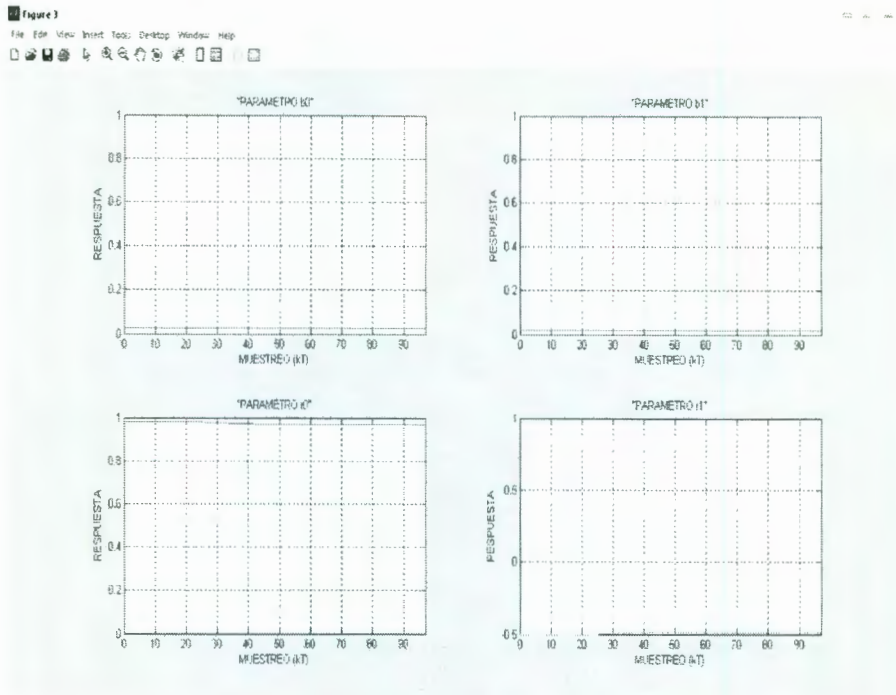
Figura 5.4
Cambiando los parámetros de la planta en el control adaptable directo



Gráfica 5.17
 Respuesta de la planta con dinámica difícil con $k_p=2.05$

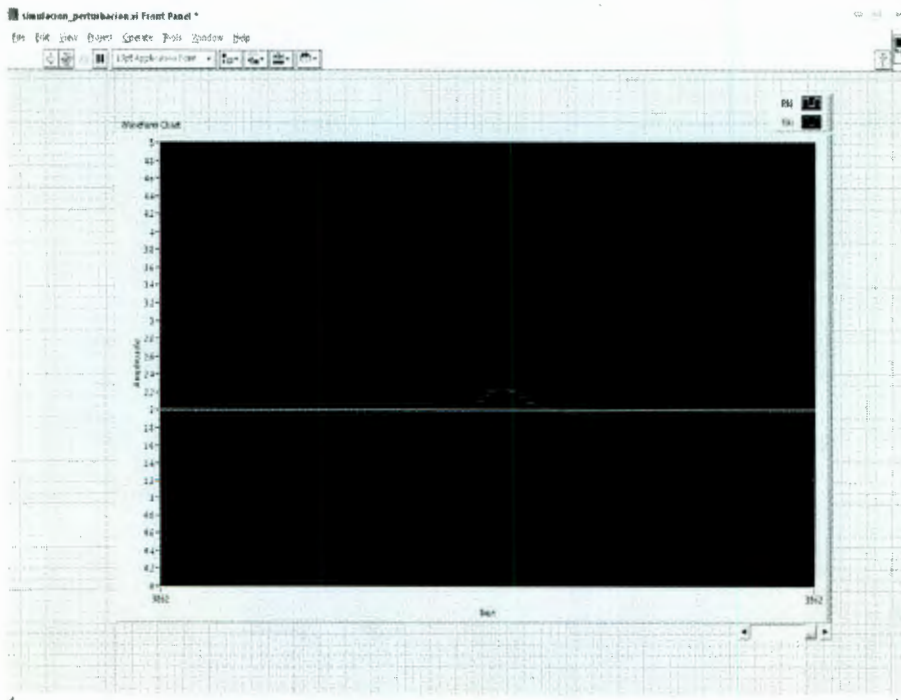


Gráfica 5.18
 Señal de control a la dinámica difícil de la planta con $k_p=2.05$

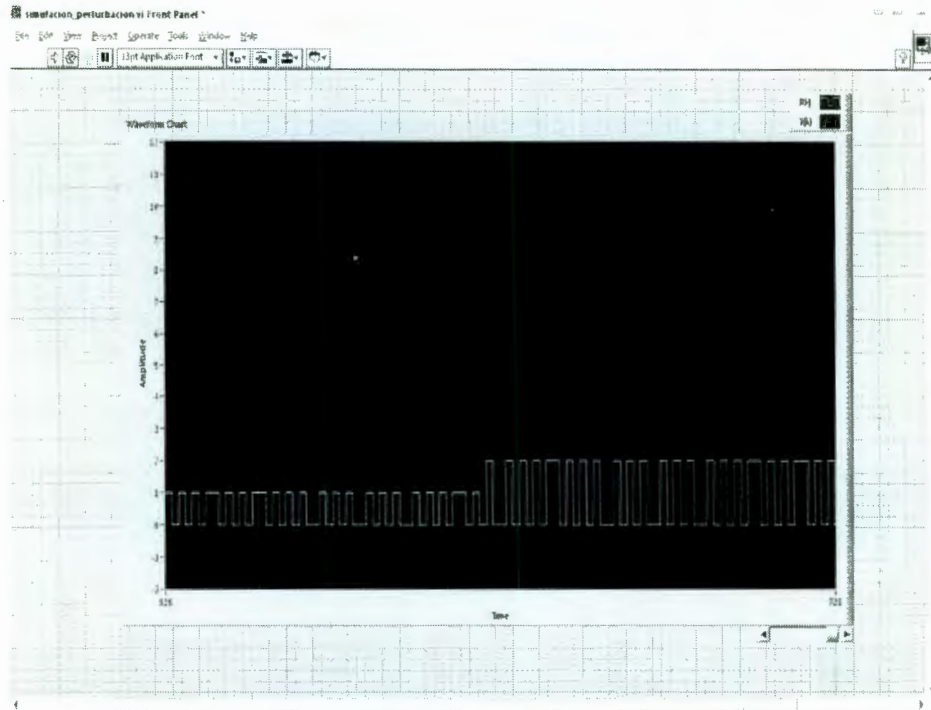


Gráfica 5.19
Valores estimados de $\hat{\theta}$ a la dinámica difícil de la planta con $k_p=2.05$

En las gráficas 5.20 y 5.21 se muestra la adaptación en tiempo real a los cambios que sufre la planta en un tiempo t .



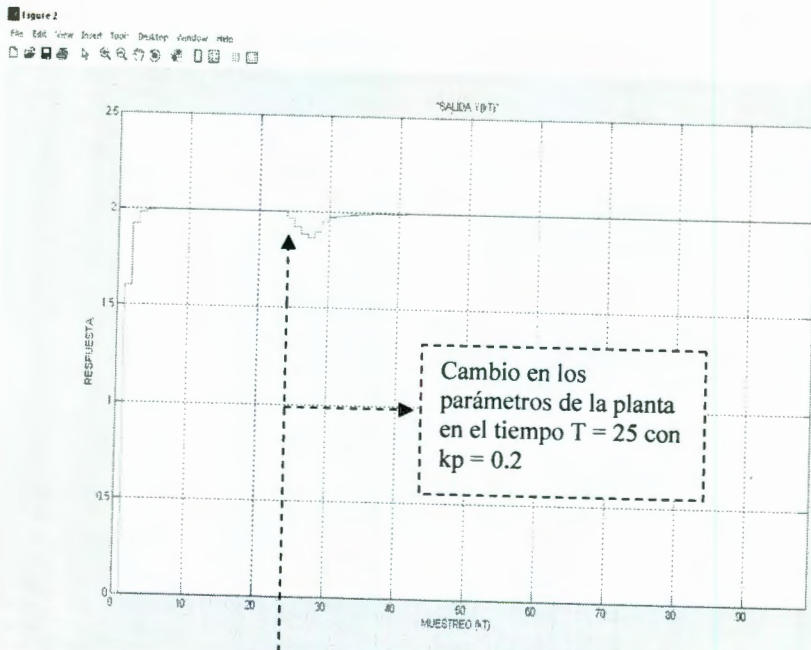
Gráfica 5.20
Respuesta de la planta con dinámica difícil con $k_p=2.05$ en tiempo real



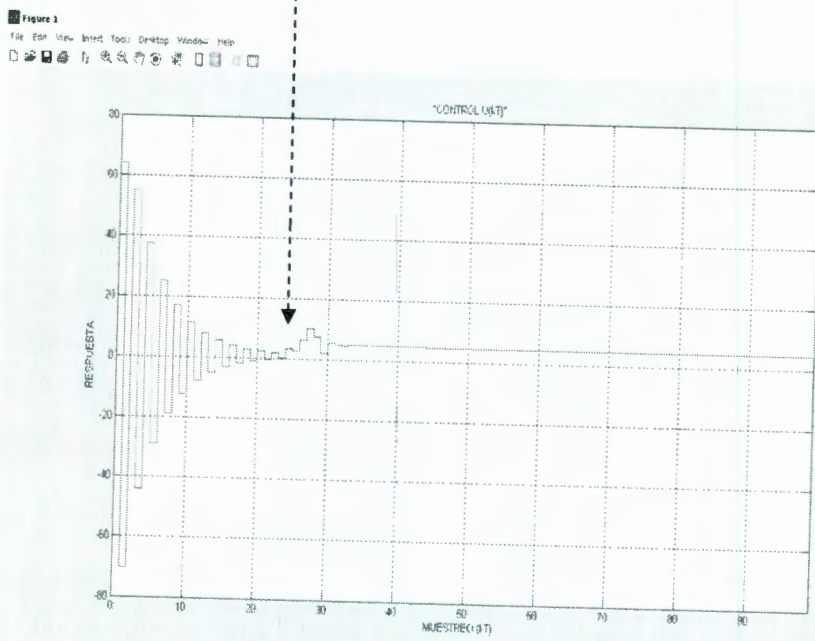
Gráfica 5.21
Señal de control a la dinámica difícil de la planta con $k_p=2.05$ en tiempo real

Ahora estableciendo $k_p=0.2$ se demostrará que el algoritmo de control adaptable compensa el cambio de la planta en el tiempo $T=25$ con el valor de referencia $U^M(k)=2$, véase gráficas 5.22 y 5.23. Y la apreciación de la evolución de los parámetros estimados b_0 , b_1 , r_0 y r_1 en la gráfica 5.26.

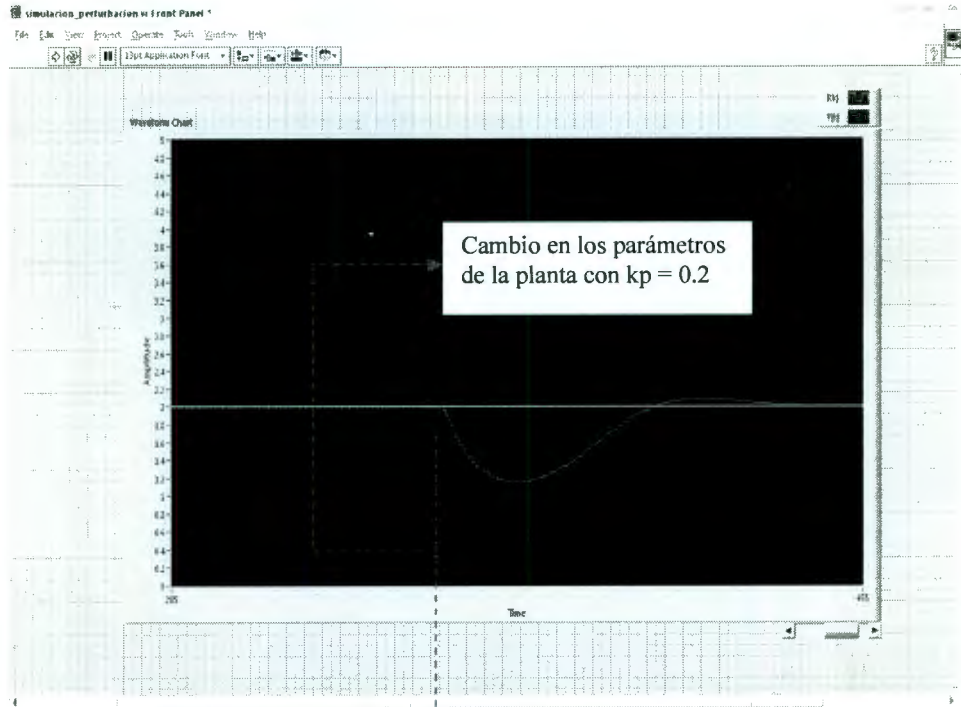
Una vez que se analizaron los resultados de las gráficas anteriores, se implemento el algoritmo adaptable al microcontrolador con la interrupción de la perturbación k_p . De esta manera los resultados fueron satisfactorios y demuestra que se hace la adaptación ante el cambio de la planta véase gráficas 5.24 y 5.25.



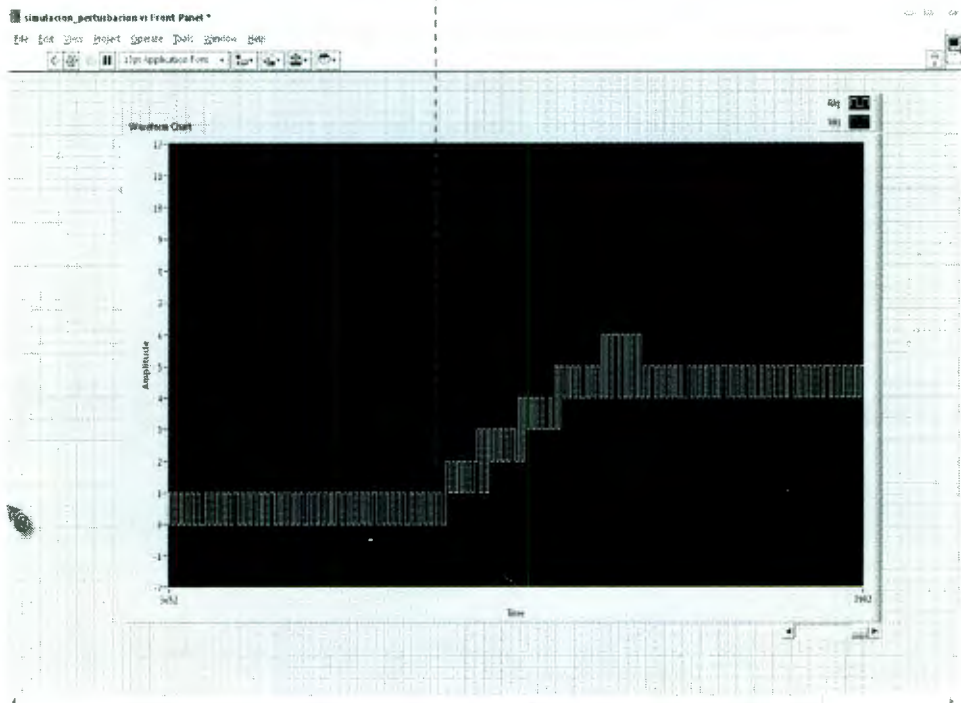
Gráfica 5.22
 Respuesta de la planta con dinámica difícil con $k_p=0.2$



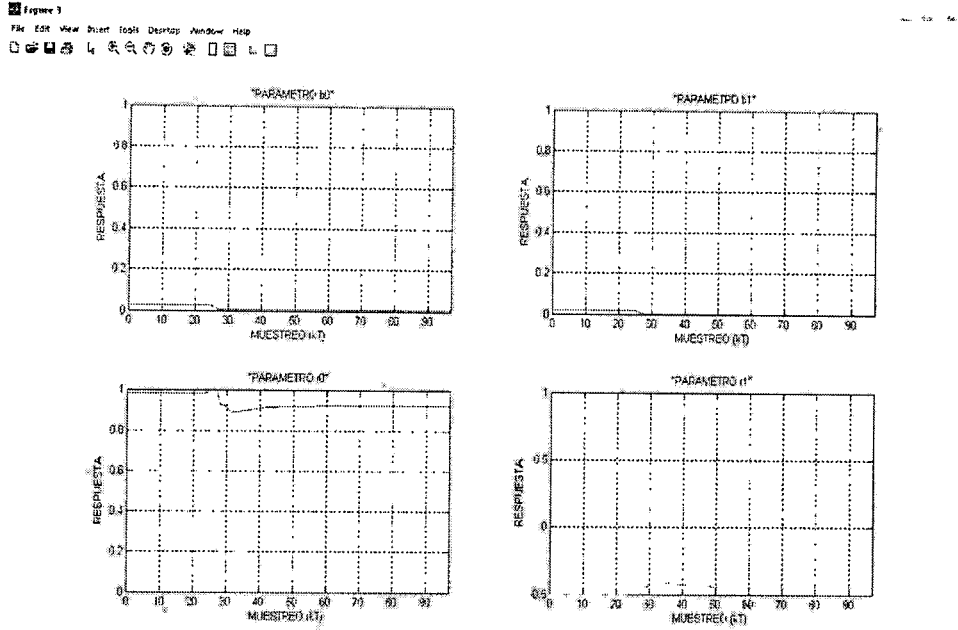
Gráfica 5.23
 Señal de control a la dinámica difícil de la planta con $k_p=0.2$



Gráfica 5.24
 Respuesta del la planta con dinámica difícil con $k_p=0.2$ en tiempo real



Gráfica 5.25
 Señal de control a la dinámica difícil de la planta con $k_p=0.2$ en tiempo real



Gráfica 5.26
 Valores estimados de $\hat{\theta}$ a la dinámica difícil de la planta con $k_p=0.2$

Para concluir este capítulo del desempeño del algoritmo de control adaptable, decir que los resultados fueron exitosos en todas las pruebas tanto de las simulaciones que se hicieron así como de las reales que se llevaron a cabo a través del microcontrolador. Por otra parte comentar también que el diseño del algoritmo de control adaptable basado en la identificación recursiva de mínimos cuadrados y llevado a la práctica tiene muchas ventajas frente a los clásicos algoritmos PID, como lo muestran los resultados anteriores donde se hace un cambio radical a la planta de tal manera que el control se adapte a los cambios y estime nuevos parámetros del controlador para cada instante de muestreo (cosa que los PID no poseen) y es ahí donde el campo de aplicación es muy tangible para el control adaptable, evitándose lo que se hace en la actualidad con los PID que se tienen que ajustar para llegar a los valores que garantizan un control satisfactorio, además que son muy susceptibles a desajustarse ante cualquier condición no considerada o perturbación.

VI. CONCLUSIONES

CONCLUSIONES

En el presente trabajo se implemento un algoritmo de control adaptable basado en la identificación recursiva de mínimos cuadrados en un microcontrolador de propósito general con un sistema de comunicación USB. La investigación de este tipo de algoritmo para el área de control permite contribuir de manera más eficiente a la solución de problemas en las áreas de control de ingeniería y de la ciencia exacta.

En la actualidad, el avance tecnológico y los métodos modernos de control hacen posible que los ingenieros diseñen sistemas de control confiables, sin embargo conforme las plantas con muchas entradas y muchas salidas se vuelven más y más complejas y variantes, el análisis y la implementación de este tipo de controladores facilita enormemente esta tarea de tal forma que se reduce el tiempo de diseño, los costos de implementación y hacen que el control sea mas robusto para el sistema.

En este proyecto de tesis, se analizaron las propiedades que ofrece el algoritmo de control adaptable y además el diseño de estos mediante el uso de dos enfoques diferentes. Una conclusión casi obvia con respecto a los algoritmos de control clásicos PID es la habilidad de adaptarse a las dinámicas complicadas de las plantas y mantener el control en seguimiento y regulación siendo estos dos últimos asignados de manera independiente por el modelo y el filtro respectivamente.

Por otra parte, se considero que en este trabajo de investigación se alcanzaron los objetivos que eran esencialmente:

- Crear nuevos enfoques para la construcción e implementación de controladores adaptables en microcontroladores de propósito general.
- El diseño, implementación y análisis de esto a un sistema con el fin de comprobar su fiabilidad y efectividad.

En la etapa de desempeño del algoritmo de control de este proyecto, se mostró que el control adaptable ofrece buenos resultados al identificar sistemas no lineales. A partir de estos experimentos se concluye que el uso de este algoritmo evita en gran medida el tratar con los algoritmos convencionales. Aunque los ejemplos presentados en este trabajo fueron para sistemas de una entrada-una salida, las ideas pueden ser extendidas para sistemas de control con muchas entradas y muchas salidas.

Al implementar el controlador adaptable en tiempo real, se comparó su respuesta con las simuladas en la plataforma de Matlab, observando que las señales de control correspondientes presentan un mínimo de error en el objetivo del control y logran la estabilidad de la planta en poco tiempo. Por tanto, de esta forma se concluye que el algoritmo de control adaptable analizado e implementado en este trabajo sugiere ser empleado en cualquier dominio de aplicación donde el objetivo principal del ingeniero de diseño sea el de resolver problemas de control sin la necesidad de emplear algoritmos complejos y por supuesto con un mínimo de recursos computacionales para su implementación.

VII. BIBLIOGRAFÍA

BIBLIOGRAFÍA

- ▶ Alberto Aguado Behar, Temas de identificación y control adaptable, PALCIEN, mayo 2000.

- ▶ Don Anderson, USB System Architecture (USB 2.0), ADDISON-WESLEY, 2001.

- ▶ Gang Fend and Rogelio Lozano, Adaptive Control System, Newnes First published 1999.

- ▶ Gang Tao, Adaptive Control Design and Analysis, a John Wiley & Sons, inc., publication, 2003.

- ▶ Jan Axelson, USB Complete Everything You Need to Develop Custom USB Peripherals Third edition, published by Lakeview Research LLC, 2005.

- ▶ Jeffrey T. Spooner, Manfredi Maggiore, Raúl Ordóñez, Kevin M. Passino, Stable Adaptive Control and Estimation for Nonlinear Systems, a John Wiley & Sons, inc., publication, 2002.

- ▶ John Hyde, Design by Example a Practical Guide to Building I/O Device, Intel University Press.

- ▶ Karl J Aström and Björn Wittenmark, Computer-Controlled System Theory and Design Third edition, Prentice Hall, 1997.

- ▶ katsuhiko Ogata, Sistemas de Control en Tiempo Discreto, segunda edición Pearson Educación, 1996.
- ▶ Raymond G. Jacquot, Modern Digital Control System, Marcel Dekker, 1981.
 - <http://www.microchip.com>
 - <http://www.ccsinfo.com>
 - <http://www.usb.org/home>

VIII. APÉNDICE

APÉNDICE

Apéndice A.

Programa del controlador lineal de seguimiento y regulación

```
close all
clc
clear
k=1;a=0;b=0;c=0;s=1;Y=0;Y_1=0;Y_2=0;U=0;U_1=0;
U_2=0;Ym_1=0;Ym=0;Ym1=0;R_1=0;R=2.0;
Ym=(0.2*Ym_1)+(0.8*R_1);

t=0:1:100;
Yg=[0:1:100];
Ug=[0:1:100];
Ymg=[0:1:100];

while(k<100)
    Ym1=(0.2*Ym)+(0.8*R);
    Y=(-0.502*Y_2)+(1.48*Y_1)+(0.0204*U_2)+(0.025*U_1);
    U=(40*Ym1)-(20*Ym)-(39.2*Y)+(20.08*Y_1)-(0.816*U_1);
    Yg(k)=Y;
    Ug(k)=U;
    Ymg(k)=Ym;
    if(s==1)
        U_2=U_1;
        U_1=U;
        Y_2=Y_1;
        Y_1=Y;
    end
    if(s==2)
        U_2=U_1;
        U_1=U;
        Y_2=Y_1;
        Y_1=Y;
        s=0;
    end
    end
    s=s+1;
    fprintf('R[%i] = %1.2f   ',k,2.0);
    fprintf('Y[%i] = %1.2f   ',k,Y);
    fprintf('Ym[%i] = %1.2f   ',k,Ym);
    fprintf('U[%i] = %1.2f\n',k,U);
    Ym=Ym1;
    k=k+1;
end

figure(1)
stairs(t,Ug,'b')
axis([0,20,-75,75])
grid on
title('CONTROL U(kT)')
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA DEL CONTROL')
```

```
figure(2)
stairs(t,Yg,'k')
axis([0,10,0,2])
grid on
title("SALIDA Y(kT)")
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA DE LA PLANTA')
```

```
figure(3)
stairs(t,Ymg,'r')
axis([0,10,0,2])
grid on
title("MODELO Ym(kT)")
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA DEL MODELO DE REF')
```

Apéndice B.

Desempeño del control ante una dinámica variante de la planta

```
close all
clc
clear
k=1;a=0;b=0;c=0;s=1;Y=0;Y_1=0;Y_2=0;U=0;U_1=0;U_2=0;
Ym_1=0;Ym=0;Ym1=0;R_1=0;R=2.0;Perturbacion=2.05;
Ym=(0.2*Ym_1)+(0.8*R_1);

t=0:1:100;
Yg=[0:1:100];
Ug=[0:1:100];
Ymg=[0:1:100];

while(k<100)
    Ym1=(0.2*Ym)+(0.8*R);
    if(k<25)
        Y=(-0.502*Y_2)+(1.48*Y_1)+(0.0204*U_2)+(0.025*U_1);
    else
        Y=(-0.502*Y_2)+(1.48*Y_1)+((Perturbacion*0.0204)*U_2)+((Perturbacion*0.025)*U_1);
    end
    U=(40*Ym1)-(20*Ym)-(39.2*Y)+(20.08*Y_1)-(0.816*U_1);
    Yg(k)=Y;
    Ug(k)=U;
    Ymg(k)=Ym;
    if(s==1)
        U_2=U_1;
        U_1=U;
        Y_2=Y_1;
        Y_1=Y;
    end
    if(s==2)
        U_2=U_1;
        U_1=U;
        Y_2=Y_1;
        Y_1=Y;
        s=0;
    end
    s=s+1;
    fprintf('R[%i] = %1.2f   ',k,2.0);
    fprintf('Y[%i] = %1.2f   ',k,Y);
    fprintf('Ym[%i] = %1.2f   ',k,Ym);
    fprintf('U[%i] = %1.2f\n',k,U);
    Ym=Ym1;
    k=k+1;
end

figure(1)
stairs(t,Ug,'b')
axis([0,99,-75,75])
grid on
title('CONTROL U(kT)')
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA DEL CONTROL')
```



```
figure(2)
stairs(t,Yg,'r')
axis([0,99,0,2.5])
grid on
title("SALIDA Y(kT)")
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA DE LA PLANTA')
```

```
figure(3)
stairs(t,Ymg,'r')
axis([0,99,0,2.5])
grid on
title("MODELO Ym(kT)")
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA DEL MODELO DE REF')
```

Apéndice C.

Aplicación del controlador adaptable

```
clc
close all
clear
k=1;a=0;b=0;c=0;s=1;Y=0;Y_1=0;Y_2=0;U=0;U_1=0;U_2=0;Ym_1=0;
Ym=0;Ym1=0;R_1=0;R=2.0;muestras=100;lamda=1.1;
t=0:1:muestras;

Yg=[0:1:muestras];
Ug=[0:1:muestras];
b0=[0:1:muestras];
b1=[0:1:muestras];
r0=[0:1:muestras];
r1=[0:1:muestras];
Ymg=[0:1:muestras];
Ym=(0.2*Ym_1)+(0.8*R_1);

for n = 1:4
    for m = 1:4
        Ft(n,m)=0;
        Fi(n,m)=0;
        Ft1(n,m)=0;
        Te(n,m)=0;
        Te_1(n,m)=0;
    end
end

Ft(1,1)=0; Ft(1,2)=0; Ft(1,3)=0; Ft(1,4)=50;
Ft(2,1)=0; Ft(2,2)=0; Ft(2,3)=0; Ft(2,4)=50;
Ft(3,1)=0; Ft(3,2)=0; Ft(3,3)=0; Ft(3,4)=50;
Ft(4,1)=0; Ft(4,2)=0; Ft(4,3)=0; Ft(4,4)=50;

Te_1(1,1)=1;
Te_1(2,1)=1;
Te_1(3,1)=1;
Te_1(4,1)=1;

while(k<muestras)
    Ym1=(0.2*Ym)+(0.8*R);
    Y=(-0.502*Y_2)+(1.48*Y_1)+(0.0204*U_2)+(0.025*U_1);
    Ft1 = (operacion1(Ft,Fi))/lamda;
    Te = operacion2(Ft,Fi,Te_1,Y,Y_1);
    U=(1/Te(1,1))*(Ym1-(0.5*Ym)-(Te(3,1)*Y)-(Te(4,1)*Y_1)-(Te(2,1)*U_1));
    Fi(1,1)=U;
    Fi(2,1)=U_1;
    Fi(3,1)=Y;
    Fi(4,1)=Y_1;
    Ft=Ft1;
    Te_1=Te;
    Yg(k)=Y;
    Ug(k)=U;
    Ymg(k)=Ym;
    b0(k)=Te_1(1,1);
    b1(k)=Te_1(2,1);
    r0(k)=Te_1(3,1);
    r1(k)=Te_1(4,1);
end
```

```

    if(s==1)
        U_2=U_1;
        U_1=U;
        Y_2=Y_1;
        Y_1=Y;
    end
    if(s==2)
        U_2=U_1;
        U_1=U;
        Y_2=Y_1;
        Y_1=Y;
        s=0;
    end
    s=s+1;
    fprintf('R[%i] = %1.2f    ',k,R);
    fprintf('Y[%i] = %1.2f    ',k,Y);
    fprintf('Ym[%i] = %1.2f    ',k,Ym);
    fprintf('U[%i] = %1.2fn',k,U);
    Ym=Ym1;
    k=k+1;
end

```

```

figure(1)
stairs(t,Ug,'r')
axis([0,99,-2,12])
grid on
title('CONTROL U(kT)')
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA')

```

```

figure(2)
stairs(t,Yg,'b')
axis([0,99,0,4])
grid on
title('SALIDA Y(kT)')
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA')

```

```

figure(3)
subplot(2, 2, 1)
plot(t,b0,'r')
axis([0,99,0,1])
grid on
title('PARAMETRO b0')
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA')

```

```

subplot(2, 2, 2)
plot(t,b1,'g')
axis([0,99,0,1])
grid on
title('PARAMETRO b1')
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA')

```

```

subplot(2, 2, 3)
plot(t,r0,'b')
axis([0,99,0,1])
grid on
title("PARAMETRO r0")
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA')

subplot(2, 2, 4)
plot(t,r1,'m')
axis([0,99,0,-.5,1])
grid on
title("PARAMETRO r1")
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA')

figure(4)
stairs(t,Ymg,'r')
axis([0,60,0,6])
grid on
title("MODELO DE REFERENCIA Ym(kT)")
xlabel('MUESTREO (kT)')
ylabel('RESPUESTA')

function Ft1=operacion1(Ft,Fi)
X=Fi*Ft*Fi;
Y=X(1,1)+1;
Ft1=Ft-((Ft*Fi*Fi*Ft)/(Y));

function Te=operacion2(Ft,Fi,Te_1,Y,Y_1)
for n = 1:4
    for m = 1:4
        W(n,m)=0.0;
    end
end
W(1,1)=Y-(0.5*Y_1);
X=Fi*Ft*Fi;
Y=X(1,1)+1;
A=W-(Te_1*Fi);
Te=Te_1+(((Ft*Fi)*A)/Y);

```