

**SISTEMA DE PROTOTIPADO RÁPIDO DE CONTROL PARA EL MÓDULO DE  
PRACTICAS MIC955 DE LA EMPRESA FEEDBACK**



**Oscar Fernando Jaramillo Chamorro  
José Ricardo Galíndez López**

**Director  
Mg. Juan Fernando Flórez Marulanda**

**Universidad del Cauca  
Facultad de Ingeniería Electrónica y Telecomunicaciones  
Departamento de Electrónica Instrumentación y Control  
Ingeniería en Automática Industrial  
Popayán, Abril de 2011**

**SISTEMA DE PROTOTIPADO RÁPIDO DE CONTROL PARA EL MÓDULO DE  
PRACTICAS MIC955 DE LA EMPRESA FEEDBACK**



**Oscar Fernando Jaramillo Chamorro  
José Ricardo Galíndez López**

**Monografía presentada como requisito parcial para optar por el título de  
Ingenieros en Automática Industrial**

**Director  
Mg. Juan Fernando Flórez Marulanda**

**Universidad del Cauca  
Facultad de Ingeniería Electrónica y Telecomunicaciones  
Departamento de Electrónica Instrumentación y Control  
Ingeniería en Automática Industrial  
Popayán, Abril de 2011**

Nota de aceptación: \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

\_\_\_\_\_  
Firma del presidente del jurado

\_\_\_\_\_  
Firma del jurado

\_\_\_\_\_  
Firma del jurado

Popayán, Abril de 2011

## **AGRADECIMIENTOS**

Los autores del presente trabajo manifiestan sus agradecimientos al Magíster Juan Fernando Flórez Marulanda, por brindarles la oportunidad de trabajar en este proyecto y haber demostrado un gran interés en el desarrollo del mismo, al Ingeniero Carlos Felipe Rengifo, MSc, cuyos conocimientos son fuente de inspiración, a los amigos y compañeros quienes contribuyeron en la culminación de este trabajo.

*“Lo que un hombre piensa de sí mismo, esto es lo que determina, o más bien indica,  
su destino”*

*Henry David Thoreau*

*A Dios, por permitirme estar aquí.*

*A mis padres, Alfonso y Stella, por su amor que no conoce límites.*

*A mis hermanos, Cristian y Daniela, por su apoyo incondicional.*

*A Sami, por estar a mi lado*

*A mi familia, amigos y profesores*

*A la Universidad del Cauca.*

*Gracias Totales*

*Oscar Fernando Jaramillo Chamorro*

*“Lo que importa en la vida no son los objetivos que nos marcamos, sino los caminos que seguimos para lograrlo”*

*Peter Bamm*

*A cada miembro de mi familia, por el amor y comprensión brindados en cada momento de mi vida.*

*A mis maestros, por la luz que me infunden con su sabiduría.*

*A mis hermanos del alma, por su gran apoyo.*

*A el amor de mi vida.*

*A la Universidad del Cauca.*

*Simplemente, Gracias*

*José Ricardo Galíndez López*

## CONTENIDO

INTRODUCCIÓN.....	1
1 SISTEMAS DE PROTOTIPADO RÁPIDO DE CONTROL.....	2
1.1 SISTEMAS DE TIEMPO REAL .....	2
1.2 SISTEMAS OPERATIVOS DE TIEMPO REAL .....	3
1.3 PROTOTIPADO RÁPIDO DE CONTROL .....	5
1.3.1 Ventajas y desventajas del RCP .....	7
1.3.2 Ejemplos de RCP en la actualidad .....	8
1.4 SISTEMA OPERATIVO GNU LINUX EN TIEMPO REAL Y RTAI-LAB .....	9
1.4.1 Arquitectura de RTAI-Lab.....	10
1.4.2 Paleta de RTAI-Lib.....	11
1.5 RESUMEN .....	13
2 INTERFAZ DE COMUNICACIÓN ENTRE EL MÓDULO DE TEMPERATURA MIC955 DE LA FEEDBACK Y RTAI-LAB. ....	14
2.1 CARACTERÍSTICAS DEL MÓDULO MIC 955 DE LA FEEDBACK .....	14
2.1.1 Descripción física del Módulo MIC 955 .....	14
2.1.2 Configuración y comunicación entre el Módulo MIC 955 y el PC. ....	16
2.2 EL PUERTO PARALELO .....	18
2.2.1 Descripción General del Puerto Paralelo .....	19
2.2.2 Acceso al Puerto Paralelo .....	20
2.3 CONEXIÓN ENTRE EL MIC 955 Y EL PUERTO PARALELO .....	21
2.4 COMUNICACIÓN ENTRE EL MIC 955 Y EL PUERTO PARALELO .....	24

2.4.1 Sensor de Temperatura MIC 955.....	26
2.4.2 Bloque MIC 955 Actuador .....	27
2.5 ACONDICIONAMIENTO DE SEÑALES.....	29
2.5.1 Transmisor de Temperatura.....	29
2.5.2 Actuador.....	31
2.6 PRUEBAS DE IMPLEMENTACIÓN .....	34
2.7 RESUMEN .....	35
3 CONTROLADORES PID .....	36
3.1 CONCEPTOS BÁSICOS DE CONTROL REALIMENTADO .....	36
3.2 ALGORITMOS DE CONTROL .....	37
3.2.1 Controlador ON/OFF .....	37
3.2.2 Controlador Proporcional .....	38
3.2.3 Controlador Proporcional Integral .....	40
3.2.4 Controlador Proporcional Derivativo.....	41
3.2.5 Controlador PID .....	43
3.3 IMPLEMENTACIONES DEL CONTROLADOR PID.....	44
3.3.1 Controladores PID de estructura Paralela .....	45
3.3.2 Controladores PID de estructura Serie.....	47
3.4 MODIFICACIONES ADICIONALES DE UN CONTROLADOR PID.....	49
3.4.1 Efecto <i>Windup</i> del Integrador.....	49
3.4.2 <i>Bump Transfer</i> o Salto de transferencia entre modos de control.....	50
3.4.3 Técnicas Antiwindup Bumpless Transfer .....	51
3.5 CONTROLADORES PID EN <i>Scilab/Scicos</i> .....	54
3.5.1 Controlador PID con Filtro Derivativo .....	54
3.5.2 Controlador PID Clásico.....	56

3.6 RESUMEN .....	58
4 SINTONIZACIÓN DE CONTROLADORES PID .....	59
4.1 ESPECIFICACIONES DE DISEÑO.....	59
4.2 CRITERIOS DE DESEMPEÑO .....	60
4.2.1 Criterios de desempeño temporales.....	60
4.2.2 Criterios Integrales de Desempeño .....	62
4.3 SINTONIZACIÓN DE CONTROLADORES PID .....	63
4.3.1 Identificación de Procesos .....	64
4.3.2 Métodos de sintonización para controladores PID .....	66
4.4 RESUMEN .....	69
5 PROTOTIPADO RÁPIDO DE CONTROL DE TEMPERATURA PARA EL MÓDULO MIC955 DE LA EMPRESA FEEDBACK.....	70
5.1 PROCEDIMIENTO PARA REALIZAR RCP DE TEMPERATURA EN EL MÓDULO MIC955.....	70
5.1.1 Definición de los objetivos y/o requerimientos del sistema de control y selección de la estructura del controlador PID .....	71
5.1.2 Identificación y validación de un modelo POMTM del proceso térmico que ocurre en el módulo MIC955.....	71
5.1.3 Sintonización de los parámetros del controlador PID mediante métodos de sintonización válidos para el modelo POMTM .....	71
5.1.4 Diseño y simulación de una versión prototipo del sistema de control mediante diagramas de bloques.....	72
5.1.5 Diseño y generación del código ejecutable (tarea en tiempo real) de la versión prototipo del sistema de control en el sistema de RCP .....	74
5.1.6 Pruebas operacionales de la versión prototipo del sistema de control en el módulo MIC955 (ejecución de la tarea en tiempo real) .....	74
5.1.7 Evaluación mediante criterios de desempeño integrales y temporales del comportamiento de la versión prototipo del sistema de control, en lazos de control tanto regulatorios como servocontrolados .....	74

5.2 REQUERIMIENTOS Y OBJETIVOS QUE DEBE CUMPLIR EL SISTEMA DE CONTROL DE TEMPERATURA EN EL MÓDULO MIC955 .....	74
5.3 IDENTIFICACIÓN Y VALIDACIÓN DE UN MODELO POMTM DEL PROCESO TÉRMICO QUE OCURRE EN EL MÓDULO MIC955 .....	75
5.4 SELECCIÓN DE LOS MÉTODOS DE SINTONIZACIÓN PARA EL CONTROL DE TEMPERATURA EN EL SISTEMA RCP .....	77
5.4.1 Métodos de sintonización para el controlador PID con filtro derivativo.....	77
5.4.2 Métodos de sintonización para el controlador PID clásico .....	78
5.5 IMPLEMENTACIÓN DE SISTEMAS DE CONTROL EN EL SISTEMA DE RCP ..	79
5.5.1 Paleta para realizar RCP de temperatura en el Módulo MIC955 de la empresa Feedback.....	81
5.5.2 Diagrama de bloques para la identificación del modelo del proceso .....	83
5.5.3 Diagramas de bloques de la versión prototipo del sistema de control para el lazo de control regulatorio.....	84
5.5.4 Diagramas de bloques de la versión prototipo del sistema de control para el lazo servocontrolado.....	86
5.6 RESULTADOS DE LAS PRUEBAS OPERACIONALES DE LOS PROTOTIPOS DE CONTROL .....	87
5.6.1 Pruebas operacionales en el lazo de control regulatorio .....	87
5.6.2 Pruebas operacionales en el lazo de servocontrol .....	89
5.7 EVALUACIÓN MEDIANTE CRITERIOS DE DESEMPEÑO INTEGRALES Y TEMPORALES DEL COMPORTAMIENTO DEL SISTEMA DE CONTROL .....	91
5.7.1 Evaluación de la versión prototipo del sistema de control utilizando el control PID con filtro derivativo .....	92
5.7.2 Evaluación de la versión prototipo del sistema de control utilizando el control PID Clásico.....	94
5.8 RESUMEN .....	97
6 CONCLUSIONES.....	98
7 BIBLIOGRAFIA .....	100

## LISTA DE ILUSTRACIONES

Ilustración 1.1: Ejecución de una tarea en tiempo real.....	4
Ilustración 1.2: Ejecución de una tarea en tiempo real.....	6
Ilustración 1.3: Paleta de RTAI-Lib .....	12
Ilustración 2.1: Módulo MIC 955 .....	15
Ilustración 2.2: Modo de funcionamiento del conversor ADC.....	17
Ilustración 2.3: El Puerto Paralelo.....	20
Ilustración 2.40: Bloque MIC 955 Actuador.....	28
Ilustración 2.51: Bloque Planta MIC 955.....	29
Ilustración 2.64: Actuador MIC 955.....	32
Ilustración 2.74: Relación EC y PWM vs Potencia.....	32
Ilustración 2.8: Funcionamiento del PWM MIC 955 .....	33
Ilustración 2.9: Respuesta de la Planta ante un escalón del 1% desde T-amb.....	34
Ilustración 3.1: Diagrama de bloques de un sistema de control realimentado.....	36
Ilustración 3.2: Respuesta de un controlador On-Off .....	38
Ilustración 3.3: Control Proporcional de temperatura para un intercambiador de calor.....	40
Ilustración 3.4: Respuesta de un controlador PI.....	41

Ilustración 3.5. Respuesta de un controlador PD en lazo abierto.....	43
Ilustración 3.6: Diagrama de bloques de las estructuras serie y paralelo en un controlador PID .....	44
Ilustración 3.7: Diagrama de bloques controlador PID ideal.....	45
Ilustración 3.8: Controlador PID Paralelo .....	46
Ilustración 3.9. Efecto <i>windup</i> en un lazo de control de temperatura para un intercambiador de calor. ....	50
Ilustración 3.10: Diagrama de bloques controlador PI operado en modo manual ...	51
Ilustración 3.11: <i>Antireset windup</i> .....	52
Ilustración 3.12: Estrategia <i>Conventional Anti-windup</i> .....	52
Ilustración 3.13: Estructura de Control por Modelo Interno .....	53
Ilustración 3.14: Implementación AntiWindup para IMC.....	54
Ilustración 3.15: Implementación en diagrama de bloques del Controlador PID con Filtro Derivativo.....	55
Ilustración 3.16: Implementación mediante diagrama de bloques de la técnica AWBT basada en re-cálculo y seguimiento del modo integral .....	56
Ilustración 3.17: Implementación componente D para el controlador PID con filtro derivativo. ....	56
Ilustración 3.18: Implementación en diagrama de bloques del controlador PID clásico.....	57
Ilustración 3.19: Implementación del componente D para el controlador PID clásico.....	58
Ilustración 4.1: Respuesta de un proceso a cambios en el SP .....	61
Ilustración 4.2: Curva de respuesta de un proceso .....	62

Ilustración 4.3: Oscilaciones sostenidas de la variable controlada.....	67
Ilustración 5.1: Curva de reacción del proceso térmico que ocurre en el módulo MIC955 .....	76
Ilustración 5.2: Comparación de los modelos POMTM y Temperatura real .....	77
Ilustración 5.3: Acceso directo a la consola de GNU Linux .....	80
Ilustración 5.4: Acceso directo al software Scilab/Scicos .....	80
Ilustración 5.5: Ventana de Aplicación del Software Scilab .....	80
Ilustración 5.6: Área de trabajo de <i>Scicos</i> .....	81
Ilustración 5.7: Paleta RCP-MIC955 .....	81
Ilustración 5.8: Bloque Normalización .....	82
Ilustración 5.9: Bloque Módulo de Control Manual (MCM) .....	82
Ilustración 5.10: Bloque tr_mode .....	83
Ilustración 5.11: Ventana de configuración controlador PID.....	83
Ilustración 5.12: Diagrama de bloques para la identificación del proceso .....	84
Ilustración 5.13: Diagrama de bloques para la simulación de la versión prototipo del sistema de control regulatorio .....	84
Ilustración 5.14: Diagrama de bloques para la simulación del disturbio. ....	85
Ilustración 5.15: Diagrama de bloques para generar la tarea en tiempo real asociada a la versión prototipo del sistema de control regulatorio .....	85
Ilustración 5.16: Diagrama de bloques para la simulación del lazo servocontrolado... ..	86
Ilustración 5.17: Diagrama de bloques para generar la tarea en tiempo real asociada al sistema servocontrolado .....	87

Ilustración 5.18: Respuesta del controlador PID Clásico sintonizado mediante el método <i>mínimum</i> ITAE de Huang – Chaos en el lazo de control regulatorio .....	88
Ilustración 5.19: Respuesta del controlador PID Clásico sintonizado mediante el método <i>mínimum</i> ITAE de Huang – Chaos en el lazo servocontrolado .....	90
Ilustración 5.20: Respuesta del controlador PID Clásico sintonizado mediante el método <i>mínimum</i> ITAE de Huang – Chaos en el lazo servocontrolado .....	91
Ilustración 5.21: Respuesta del Controlador PID con Filtro Derivativo sintonizado con el método <i>minimun</i> IAE de Alfaro Ruiz en el lazo de control regulatorio .....	92
Ilustración 5.22: Respuesta del Controlador PID con Filtro Derivativo sintonizado con el método <i>minimun</i> IAE de Alfaro Ruiz en el lazo de servocontrol .....	94
Ilustración 5.23: Respuesta del Controlador PID Clásico sintonizado con el método <i>minimun</i> ITAE de Huang – Chaos, 1982, en el lazo de control regulatorio .....	95
Ilustración 5.24: Respuesta del Controlador PID Clásico sintonizado con el método <i>minimun</i> IAE de Huang – Chaos, en el lazo de servocontrol .....	96

## LISTA DE TABLAS

Tabla 2.1: Configuración de puertos del módulo MIC 955. Modificado de (Feedback, 1983).....	16
Tabla 2.2: Configuración de puertos del módulo MIC 955. Fuente propia .....	17
Tabla 2.3: Tipos de Puerto Paralelo. Modificado de (ETI, 2011) .....	19
Tabla 2.4: Mapeo de señales Puerto Paralelo – Módulo MIC 955. Fuente propia. ....	23
Tabla 4.1: Formulas de ajuste de controladores P, PI y PID para el método de Ziegler - Nichols en lazo abierto.....	66
Tabla 4.2: Formulas de ajuste de controladores P, PI y PID para el método de Ziegler - Nichols en lazo cerrado .....	67
Tabla 4.3: Formulas de ajuste del método CHR .....	68
Tabla 4.4: Formulas de ajuste del método Cohen - Coon.....	68
Tabla 5.1: Evaluación del desempeño del controlador PID con filtro derivativo sintonizado con el método <i>minimun</i> IAE de Alfaro – Ruiz en el lazo de control regulatorio.....	93
Tabla 5.2: Evaluación del desempeño del controlador PID con filtro derivativo, sintonizado con el método <i>minimun</i> IAE de Alfaro – Ruiz para el lazo de servocontrol .....	94
Tabla 5.3: Evaluación del desempeño del controlador PID clásico, sintonizado con el método <i>minimun</i> ITAE de Huang – Chaos para el lazo de control regulatorio. ....	95
Tabla 5.4: Evaluación del desempeño del controlador PID Clásico, sintonizado con el método <i>minimun</i> ITAE de Huang – Chaos para el lazo de servocontrol .....	97

## LISTA DE CUADROS

Cuadro 5.1: Método de sintonización *Minimum IAE* Alfaro – Ruiz (2005a) para lazos de control regulatorio. Fuente: (O’Dwyer, 2009) ..... 78

Cuadro 5.2: Método de sintonización *Minimum ITAE* Huang-Chao (1982) para lazos de control regulatorio y servocontrolados. Fuente: (O’Dwyer, 2009) ..... 79

## **LISTA DE ANEXOS**

**ANEXO A:** GUIA DE INSTALACIÓN DEL SISTEMA OPERATIVO GNU LINUX FEDORA 7

**ANEXO B:** GUIA DE INSTALACIÓN DE RTAI-LAB

**ANEXO C:** CREACIÓN DE BLOQUES EN RTAI-LAB CON SCILAB/SCICOS 4.1.2

**ANEXO D:** LINEALIZACIÓN, FILTRADO Y NORMALIZACIÓN DEL TRANSMISOR DE TEMPERATURA Y EXPERIMENTOS ADJUNTOS

**ANEXO E:** IMPLEMENTACIÓN DEL COMPONENTE DERIVATIVO

**ANEXO F:** MÉTODOS DE IDENTIFICACIÓN TEORÍA Y PRACTICA

**ANEXO G:** RESULTADOS PRUEBAS OPERACIONALES

**ANEXO H:** GUIAS DE PRÁCTICA PARA CONTROL DE TEMPERATURA EN EL MÓDULO MIC955

## RESUMEN

Este documento describe el desarrollo de un Sistema de Prototipado Rápido de Control (RCP) para el módulo MIC955 de la empresa Feedback, el cual pertenece al laboratorio de Instrumentación Industrial del Programa de Ingeniería en Automática Industrial (PIAI) de la Universidad del Cauca. Este sistema de RCP permite que los estudiantes puedan desarrollar diversas prácticas de control de temperatura, mediante el uso de diferentes tipos de controladores, con lo cual se espera que los estudiantes del PIAI mejoren sus conocimientos en control de procesos.

En el capítulo 1 se presentan conceptos relativos a los sistemas de RCP, los sistemas operativos en tiempo real y la herramienta RTAI-Lab. En el capítulo 2 se describe el diseño e implementación de la interfaz de comunicación que permite la conexión del módulo MIC955 con el PC a través del puerto paralelo. En el capítulo 3 se expone los conceptos de los sistemas de control realimentado haciendo énfasis en los controladores tipo PID. En el capítulo 4 se presenta información acerca del proceso de sintonización de controladores PID y la evaluación del comportamiento de los sistemas de control mediante criterios de desempeño temporales e integrales. En el capítulo 5 se describe un procedimiento para realizar RCP de temperatura en el módulo MIC955 y se presentan los resultados de pruebas operacionales que se realizaron en el sistema de RCP.

## INTRODUCCIÓN

Los Sistemas de Prototipado Rápido de Control (RCP), constituyen una poderosa herramienta desarrollada con el fin de facilitar el proceso de diseño y puesta en marcha de diferentes tipos de sistemas de control. Los sistemas de RCP permiten una rápida transición entre la simulación del modelo en ambientes asistidos por computador y la implementación real de los controladores en el proceso. El RCP es ampliamente utilizado en empresas y centros educativos debido a su funcionalidad y el bajo costo que involucra la implementación de algunos sistemas; algunas universidades incluso desarrollan sus propias herramientas para casos particulares.

El laboratorio de Instrumentación Industrial del programa de Ingeniería en Automática Industrial, cuenta con el módulo de control de temperatura MIC955 de la empresa Feedback. Anteriormente los programas de control se desarrollaban mediante el lenguaje de programación BASIC y se ejecutaban en un PC de prestaciones limitadas, razón por la cual, no se explotaba todo el potencial didáctico que el módulo MIC955 ofrece, en cuanto a capacitación de estudiantes en el área de control de procesos.

Por este motivo se implementó un sistema de RCP para el módulo MIC955, mediante el reemplazo del PC anterior por un equipo de características más avanzadas y el desarrollo de interfaces de comunicación adecuadas, con el cual se pudo actualizar las prácticas de control de temperatura y explorar diferentes métodos de control a través del uso de herramientas de libre distribución, las cuales, gracias a su bajo costo, permiten seguir el camino propuesto en anteriores trabajos de grado, de utilizar sistemas operativos de tiempo real.

La principal contribución de este trabajo, es que el sistema RCP permite realizar prácticas de control de temperatura utilizando controladores ON/OFF, P, PI y PID y permite poner en práctica los conocimientos teóricos obtenidos en cursos del programa como instrumentación industrial, control analógico e identificación de sistemas, además también permite entender ciertos fenómenos no lineales como el *bump transfer* y el *windup* del integrador.

# 1 SISTEMAS DE PROTOTIPADO RÁPIDO DE CONTROL

Las metodologías de diseño asistidas por computador tienen gran relevancia en todas las áreas de ingeniería; constituyen una importante ayuda a los ingenieros debido a la creciente complejidad de los procesos. En el campo de la regulación, del control y de las tecnologías de automatización, los Sistemas de Prototipado Rápido de Control o RCP (*Rapid Control Prototyping*, por sus siglas en Inglés), como método y herramienta de trabajo, permiten integrar diferentes fases de la implementación de un sistema de control, precisando el uso de técnicas de modelado y simulación de procesos, como un paso necesario para su implementación y total verificación (Dirk et al, 2008).

El RCP nació en el mercado automotriz de Estados Unidos en la década de 1990 como una solución para gestionar las complejas tareas de la ingeniería de control (PresicionMBA, 2005). Puede ser implementado rápida y directamente en diferentes tipos de dispositivos y procesos industriales, controlados por complejos sistemas de control distribuido. Tiene como mérito permitir la optimización dinámica de la implementación del sistema de control, desde fases muy tempranas de su desarrollo, haciéndole más flexible y permitiendo una mirada retrospectiva que disminuirá acciones correctivas. Es utilizado ampliamente en diversas industrias como la automotriz, en procesos tales como antibloqueo de frenos y estabilidad de vehículos, en la industria aeroespacial, en sistemas de control de vuelo, en automatización industrial, máquinas móviles, sistemas de navegación, servo-control y en el desarrollo de dispositivos médicos (Chen et al, 2010).

## 1.1 SISTEMAS DE TIEMPO REAL

Actualmente el funcionamiento apropiado de muchos sistemas y dispositivos de uso común, no dependen solamente del efecto de las acciones que estos realizan sino también del tiempo en el cual estas acciones son ejecutadas. Por ejemplo, cuando el conductor de un automóvil aplica el freno, el control de antibloqueo analiza las condiciones del sistema (la velocidad del automóvil, la superficie del camino y la dirección) y activa el freno con la temporización apropiada en centésimas de segundo, por lo que el resultado (la activación del freno) y el tiempo en el que se produce es importante, asegurando la integridad del sistema. Recientemente se ha incluido hardware y software de cómputo de tiempo real en la mayoría de los sistemas, con el fin de supervisar y controlar su funcionamiento; estos sistemas computacionales se denominan sistemas embebidos, sistemas de cómputo de tiempo real, o simplemente sistemas de tiempo real. Al contrario de los sistemas de cómputo convencionales,

estos se vinculan estrechamente con las condiciones del proceso a supervisar y controlar (Cheng, 2002).

Un sistema de tiempo real se puede definir como un sistema en el cual su funcionamiento apropiado, no depende solamente de los resultados lógicos de los cómputos sino también del momento físico en el cuál estos resultados se producen (Kopetz, 2002). En algunos sistemas, incluso es más importante éste último, escogiendo un método de cálculo aproximado más rápido, con el objetivo de cumplir las restricciones temporales; cuando el tiempo de respuesta es menor que el tiempo en que la variable o condición determinada tarda en ocasionar un cambio en el proceso, se dice que el sistema de control opera en tiempo real. Sin embargo es importante aclarar que el tiempo de respuesta no está dado solamente por la velocidad sino también por el determinismo (Rodríguez, 2007).

Un sistema de tiempo real debe reaccionar a los estímulos del objeto controlado o del operador, dentro de los intervalos de tiempo exigidos por el ambiente de trabajo. El instante en el cual un resultado debe producirse se llama plazo (*deadline*). Si un resultado es útil incluso después de exceder el tiempo de plazo, el sistema se clasifica como sistema de "tiempo real suave" (*soft real-time*), en el cual se acepta la pérdida de datos, reduciendo la calidad del servicio. Por ejemplo, la pérdida de marcos mientras se reproduce un video, en este caso la pérdida de algunos cuadros no constituye una falta grave en la aplicación. Por otro lado, si pudiera presentarse un evento adverso debido al no cumplimiento de un plazo, el sistema se clasifica como sistema de "tiempo real duro" (*hard real-time*); estos sistemas deben garantizar el tiempo de ejecución y no permiten limitaciones en la latencia. Por ejemplo, un semáforo en un cruce de ferrocarril, si el semáforo no cambia a "rojo" antes del cruce de un tren, podría ocurrir una catástrofe (Kopetz, 2002).

Un sistema de tiempo real debe tener las siguientes características (TimeSys, 2002):

- Respuesta rápida y predecible: El sistema debe responder rápida y predeciblemente a eventos urgentes.
- Alto grado de previsión: Deben satisfacerse los requisitos de tiempo del sistema incluso cuando se utilice gran parte de sus recursos.
- Estabilidad bajo carga excesiva transitoria: aún cuando el sistema se cargue excesivamente por distintos eventos y sea imposible cumplir todos los plazos, debe garantizar los plazos de las tareas críticas seleccionadas.

## 1.2 SISTEMAS OPERATIVOS DE TIEMPO REAL

El software de tiempo real es ampliamente utilizado en sistemas embebidos, la línea que separa el sistema operativo de las aplicaciones en las computadoras, es a menudo difusa. Los sistemas embebidos que pueden funcionar bajo limitaciones de tiempo, generalmente requieren que el propio sistema operativo tenga capacidades de tiempo real.

Un sistema operativo en tiempo real o RTOS (*Real Time Operating System* por sus siglas en inglés) es un sistema operativo multitarea que ha sido desarrollado para aplicaciones de tiempo real, siendo compacto y eficiente, excluyendo algunas funciones que los sistemas operativos convencionales proporcionan, con el fin de responder prioritariamente a eventos y tiempos de ejecución. En un RTOS las tareas escogidas por el usuario se ejecutan sobre el núcleo (*Kernel*) para una ejecución rápida, proporcionando también manejo efectivo de interrupciones, controladores y procesos, siendo un sistema determinista, sensible, fiable y tolerante a fallos (Kamal, 2008).

Las actividades que realiza un RTOS son llamadas tareas y aquellas actividades que necesitan requerimientos de tiempo crítico son llamadas tareas de tiempo real. La ilustración 1.1 muestra la ejecución de una tarea en tiempo real, se puede apreciar que la ejecución de la tarea toma un tiempo menor al plazo requerido. Normalmente es deseable encontrar actividades con y sin tiempo crítico, coexistiendo juntas en el mismo sistema.

### Ilustración 1.1: Ejecución de una tarea en tiempo real



Fuente: Modificado de (De la Puente, 2007)

Las tareas de tiempo real típicamente tienen los siguientes requerimientos y/o limitaciones (The RTAI Development Team, 2006):

- Limitaciones de tiempo: Los cuales se presentan de forma periódica o aperiódica. Una tarea aperiódica tiene un plazo en el cual debe empezar y/o terminar, una tarea periódica es repetitiva cada periodo de tiempo. La mayoría de procesamiento sensorial es periódico, mientras que los requisitos aperiódicos pueden surgir de eventos dinámicos.
- Requerimiento de recursos: Una tarea de tiempo real puede exigir el acceso a ciertos recursos, tales como dispositivos I/O, estructuras de datos, archivos y bases de datos.
- Requerimientos de comunicación: Debe permitirse la comunicación de mensajes entre tareas.
- Limitaciones de concurrencia: Se debe permitir el acceso simultáneo de diferentes tareas a recursos comunes, sin violar la coherencia de los mismos.

### 1.3 PROTOTIPADO RÁPIDO DE CONTROL

Los métodos tradicionales de control dependen de un modelo de la planta. Aunque existen diversos métodos con los cuales se puede aproximar modelos asociados al proceso, en muchas ocasiones los parámetros de estos modelos son desconocidos o difícilmente medibles en el proceso. Mediante el desarrollo de experimentos, es posible obtener información del comportamiento dinámico de la planta que ayude a la estimación de parámetros. Un ingeniero puede empezar el diseño de un sistema de control con un modelo inexacto o inexistente de la planta. En esta etapa, el control se desarrolla con el fin de estabilizar el sistema, obteniendo las condiciones adecuadas para la realización de experimentos, que permitan adquirir datos relacionados con las respuestas del sistema, bajo diferentes condiciones de operación. Los datos obtenidos pueden ser usados para mejorar el modelo existente de la planta y diseñar un sistema de control basado en un modelo más preciso. Un diseñador puede estudiar y optimizar el rendimiento del sistema de control antes de ser implementado, mediante la simulación combinada del sistema de control y el modelo de la planta. Si el sistema no alcanza el rendimiento obtenido en la simulación, puede ser refinado posteriormente (Aström y Wittenmark, 1984).

En diversas investigaciones se ha dedicado tiempo y esfuerzo al análisis, diseño y síntesis de controladores basados en modelos del proceso, sin embargo, una vez que estos se diseñan, aún es necesario realizar la conexión con la planta real. Gran parte del esfuerzo actual en ingeniería de control, está dedicado a llevar el modelo del controlador a la implementación real. El propósito del Prototipado Rápido de Control (RCP) es obtener un diseño confiable y establecer claramente los defectos y errores en un diseño parcial, antes de escoger el diseño final (Erkkinen et al, 2005).

El RCP es un mecanismo que permite probar rápidamente diferentes estrategias de control, en cada una de las fases de desarrollo (diseño, implementación, pruebas y verificación), en un equipo que funcione en tiempo real con entradas y salidas conectadas a un dispositivo o planta real. Esto permite que los ingenieros puedan ver los resultados e interactuar con ellos, para generar rápidamente nuevas soluciones, acelerar el desarrollo y reducir costos (Chen et al, 2010).

La idea básica del RCP es desarrollar y validar nuevas estrategias y algoritmos de control, con modelos o bloques simbólicos en ambientes simulados, usando software de modelamiento matemático, los cuales, tan pronto sean diseñados y tengan un resultado de simulación aceptable, pueden ser transformados a un prototipo de control en tiempo real, creado automáticamente por el sistema sin que el usuario se preocupe por programar en lenguaje C, el cual es usado para validar el algoritmo en condiciones reales de trabajo. El proceso de RCP consiste de las siguientes fases (Hölttä, Palmroth, y Eriksson, 2004):

1. Modelamiento de la planta en un ambiente simulado
2. Validación del modelo

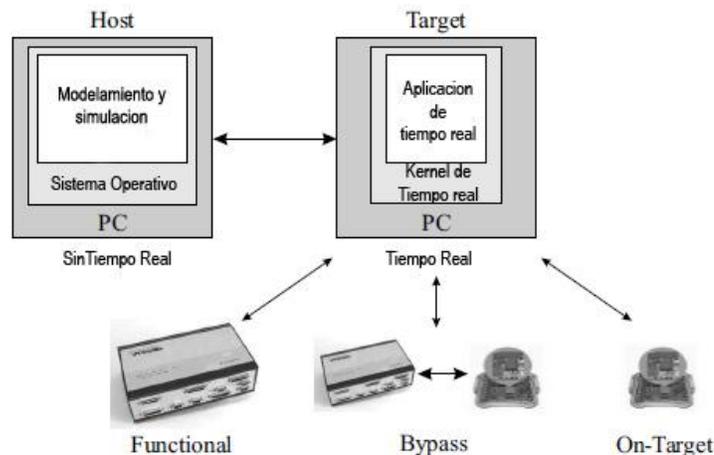
3. Diseño de una versión prototipo del sistema de control
4. Pruebas de la versión prototipo del sistema de control en el modelo simulado
5. Evaluación de la versión prototipo del sistema del control en la planta real.

Un sistema típico para realizar RCP está conformado por los siguientes elementos (Chen et al, 2010):

- Un Computador con un sistema operativo de tiempo real.
- Software de Diseño de Sistemas de Control Asistido por Computador (DSCAC), que permita el modelamiento matemático mediante funciones y/o bloques básicos.
- Un paquete de bloques simbólicos de entradas y salidas, que permitan importar la estrategia de control, probada en el modelo simulado de la planta, hacia la creación de la tarea en tiempo real, la cual se ejecutará en el hardware del RCP.
- Un generador de código que transfiera los diagramas de bloques implementados en el DSCAC, a su correspondiente algoritmo en código C, creando un archivo ejecutable para utilizar en el PC.
- Una interfaz de entradas y salidas para señales analógicas y/o digitales.
- Una aplicación GUI (*Graphical User Interface*) que monitoree el proceso y permita ajustes de parámetros en tiempo real.

La ilustración 1.2 muestra los tres enfoques básicos de RCP: *Functional*, *bypass* y *On-target rapid prototyping*. Estos enfoques se explican a continuación (Erkkinen, Kluza, y Shenoy, 2005).

### Ilustración 1.2: Ejecución de una tarea en tiempo real



Fuente Modificado de (De la Puente, 2007)

*Functional Rapid Prototyping*: Es usado en la prueba de nuevos proyectos e investigaciones donde hay ausencia de controladores o éstos no soportan las nuevas estrategias de control. En éste caso, el sistema de control original es reemplazado completamente por uno nuevo, dándole el control total al sistema de RCP. El

hardware utilizado para las funciones del RCP es a menudo tipo PC y no está destinado a aplicaciones de control de producción. Es importante la flexibilidad en cuanto a la adición de dispositivos de entradas y salidas, en situaciones donde no puedan anticiparse futuras implicaciones del proceso.

*Bypass rapid prototyping:* El controlador prototipo se implementa parcialmente y permite la adición de nuevas funciones o el reemplazo y/o modificación de funciones ya existentes. Este enfoque es útil cuando el sistema de control demanda altas capacidades de corriente, las cuales típicamente no están disponibles en los RCP o si solo ciertas funciones del controlador deben ser reemplazadas. También puede ser usado para probar nuevos sensores y actuadores sin modificar el control actual de la planta.

*On-target rapid prototyping:* Es usado directamente sobre los dispositivos de producción y captura todas las dependencias y limitaciones de sus entradas y salidas. Permite a los ingenieros evaluar la habilidad de los algoritmos de control bajo condiciones críticas de trabajo, que no pueden ser fácilmente simuladas.

Los sistemas de RCP también se pueden clasificar en las siguientes categorías:

- *DSP based rapid prototyping:* Su principal característica es la rapidez de implementación y es ampliamente utilizada en las industrias a pesar de su alto costo.
- *PC - based rapid prototyping:* En comparación con la anterior tiene una mejor relación costo - eficiencia y la habilidad de extenderse a otros sistemas (Hölttä, Palmroth, y Eriksson, 2004).

### **1.3.1 Ventajas y desventajas del RCP**

Las principales ventajas de los sistemas de RCP son: (Hölttä, Palmroth, y Eriksson, 2004):

1. El uso de modelos de simulación ayuda al diseñador del controlador a profundizar en el entendimiento de la planta, siendo éste obligado a obtener un modelo adecuado, el cual puede ser utilizado con múltiples propósitos.
2. En la simulación es posible verificar el estado de variables que son difícilmente medibles en el proceso real o en casos extremos imposibles de medir. También permite experimentar con características no disponibles en el hardware, para realizar futuras modificaciones con mejores equipos o software.
3. El software y el hardware utilizado en el RCP puede ser utilizado como herramienta de alto rendimiento para adquisición de datos, además de proporcionar un sistema de control en tiempo real para nuevos algoritmos.
4. Se ahorra tiempo y dinero en el diseño, realizando el cableado de las entradas y salidas de la planta una sola vez, ya que el mismo equipo es usado tanto en la adquisición de datos experimentales en cada una de las fases del modelamiento

- de la planta, como en las pruebas operacionales del nuevo prototipo de control en el ambiente de trabajo real.
5. No se requiere hacer ninguna modificación a los componentes de control existentes en la planta real.
  6. La rápida generación de un algoritmo en forma automática, aumenta significativamente la interacción en el diseño del control, ya que el tiempo consumido codificando manualmente el algoritmo es inexistente, adicionalmente evita la existencia de errores que puedan aparecer al crear manualmente el código.
  7. En síntesis el RCP genera una reducción en el tiempo de diseño, implementación y prueba de las estrategias de control, lo cual permite probar diferentes métodos en la planta real en menos tiempo, reduciendo significativamente los costos.

Entre las posibles desventajas de los RCP [Hölttä, Palmroth, y Eriksson, 2004]:

1. Para realizar un buen diseño del algoritmo de control, es necesario obtener un buen modelo de la planta a controlar. La presencia de no linealidades, ruidos y disturbios inesperados durante las pruebas con la planta real, pueden ocasionar un desempeño pobre del algoritmo de control, aunque este algoritmo se comporte adecuadamente en las simulaciones. En caso de presentarse esta situación, cabe aclarar que los nuevos datos obtenidos en la planta podrían utilizarse para el diseño de un mejor algoritmo de control.
2. El uso de RCP podría requerir de la compra de software y hardware usados durante los procesos de modelamiento y pruebas, estos pueden variar alrededor de \$30 - 45 mil dólares dependiendo del vendedor (PresicionMBA 2005), sin embargo estos costes son relativamente bajos, comparados con lo que podrían costar *test* similares realizados mediante otros métodos, además que el mismo software y hardware pueden ser utilizados en otros proyectos.

### 1.3.2 Ejemplos de RCP en la actualidad

Algunos ejemplos de RCP comerciales son: dSPACE (dSPACE, 2011), ConCurrent (ConCurrent, 2009), RT - LAB (Opal-RT, 2009), xPCTarget (MathWorks, 2010), y CompactRIO (National Instruments, 2011). Otros RCP de bajo costo son empleados también como herramientas de aprendizaje en diferentes universidades en cursos de teoría clásica y moderna de control (Chen et al, 2010).

A pesar de la existencia de estos RCP comerciales, muchos diseñadores han optado por desarrollar sus propios RCP para diferentes aplicaciones; por ejemplo, Hong, Gan, Chong, Chew, Lee, y Koh, implementaron un algoritmo de procesamiento para una señal digital usando MATLAB y el módulo de evaluación Texas Instrument TMS320C30 (Hong et al, 2000). Hercog y Jezernik presentaron un RPC basado en DSP utilizando MATLAB/Simulink, este RCP provee una rápida transición desde la simulación en Simulink, a la operación de tiempo real del control de un motor (Hercog y Jezernik, 2005). Wootaik, Minsuk, y Myoungcho propusieron una plataforma "*target identical*" de RCP para un controlador MPC555 (Wootaik et al, 2004). Duma, Dobra,

Abrudean, y Dobra desarrollaron un RCP usando un sistema embebido para un TI C2000 DSP para control de un motor DC (Duma et al, 2007).

#### 1.4 SISTEMA OPERATIVO GNU LINUX EN TIEMPO REAL Y RTAI-LAB

GNU Linux es un sistema operativo multitarea de distribución libre, fue creado en 1991 por Linus Torvalds con la asistencia de programadores de todo el mundo. GNU Linux permite gran variedad de aplicaciones, sin embargo no es un sistema operativo de tiempo real, lo cual es una desventaja al trabajar en áreas como control de dispositivos robóticos, aplicaciones militares, de salud y otros sistemas integrados que utilizan diferentes tipos de dispositivos. Existe varios RTOS comerciales disponibles, entre los cuales se encuentran QNX, AMX, RTKernel, estos RTOS son pequeños y convienen ser usados en sistemas integrados, sin embargo también es útil disponer de controladores de dispositivos, protocolos de redes y otras características disponibles en GNU Linux; por esta razón se han presentado varias implementaciones de tiempo real para GNU Linux, entre las cuales se destaca RTLinux. Este RTOS inserta un pequeño *kernel* de tiempo real debajo del *kernel* estándar de GNU Linux, tratándolo como un proceso de tiempo real; este sistema operativo tiene como desventaja que un error en la tarea de tiempo real podría causar daños en el sistema ya que operan en el espacio del *kernel* (Acosta, 2009).

Una solución basada originalmente en RTLinux y actualmente en ADEOS es “*Real Time Application Interface*” (RTAI), desarrollado en el departamento de Ingeniería Aeroespacial del Instituto Politécnico de Milán. Fue concebido por Paolo Mantegazza como un proyecto colaborativo de código abierto. Sus características más importantes incluyen varios métodos de comunicación flexible entre procesos, y un API simétrico que permite la creación de tareas de tiempo real en el espacio de usuario mediante el módulo LXRT, evitando la desventaja de operación en el espacio de *kernel*. Mediante esta implementación, se añade un *kernel* de tiempo real bajo el *kernel* estándar de GNU Linux, el cual pasa a ser una tarea de menor prioridad, además proporciona una amplia selección de mecanismos de comunicación entre procesos y otros servicios de tiempo real (Acosta, 2009).

RTAI-Lab es un conjunto de herramientas software incluida en la distribución de RTAI. RTAI-Lab proporciona un conjunto de herramientas para el diseño, construcción, ejecución y monitoreo de sistemas de control en tiempo real, en un ambiente DSCAC. RTAI-Lab se integra completamente en el software DSCAC para permitir el diseño de sistemas de control mediante diagramas de bloques y su posterior generación de código para un objetivo específico, mediante la adición de algunos bloques y la determinación de ciertas opciones de construcción. Estos diagramas de bloques pueden ser implementados mediante la herramienta comercial Matlab/Simulink/Real-Time workshop (RTW) (MathWorks, 2010) y/o el software de código abierto Scilab/Scicos/CodeGen CACSD (Scilab, 2011). El archivo generado es ejecutado posteriormente en tiempo real duro y puede ser controlado por una interfaz gráfica de usuario (Bucher y Dozio, 2003).

RTAI-Lab permite (Bucher et al, 2010):

- Desarrollar y ejecutar software en tiempo real en un sistema local, remoto o distribuido.
- Monitorear la ejecución de controladores locales, remotos o distribuidos.
- Cambiar los parámetros de control sobre la marcha.

El conjunto de herramientas de RTAI-Lab están basadas en (Bucher et al, 2010):

Scilab/Scicos (Scilab, 2011): Scilab es un software de código abierto para cómputo numérico que ha sido desarrollado desde 1990 por científicos del INRIA (*Institut National de Recherche on Informatique et on Automatique*) y del ENPC (*École Nationale des ponts et chaussée*). Scilab ofrece un conjunto de funciones para aplicaciones científicas y de ingeniería (Skiba et al 2006). Scicos es una herramienta incluida en el software Scilab, esta herramienta permite el diseño gráfico y la simulación de esquemas de control por medio de diagramas de bloques, los cuales también son utilizados para la generación y compilación automática de código (Netter, 2009).

Comedi: Provee una colección de controladores para una gran variedad de tarjetas de adquisición de datos comunes, los controladores se implementan como módulos en el *kernel* de GNU Linux, proporcionando una funcionalidad común e individual a bajo nivel (Comedi, 2011).

RTAI: mencionado anteriormente, prioriza el desarrollo de las tareas en tiempo real duro, puede correr FIFOs y puede transferir datos del proceso entre el espacio de usuario y el de tiempo real.

RTAI-Lib: es una paleta de bloques para Scicos que permite el diseño de diagramas de bloques con sensores y actuadores, provee una interfaz de comunicación entre RTAI y el hardware de adquisición de datos. Los diagramas de bloques que usa RTAI-Lib pueden ser compilados en un software de RTAI ejecutable.

Xrtailab: Es un osciloscopio virtual que puede ser conectado con el ejecutable de tiempo real. Xrtailab permite visualizar y monitorear tanto señales como eventos en tiempo real, usando indicadores, *scopes*, *leds*, *meters*, etc. También permite el ajuste de parámetros del sistema, mientras la tarea de tiempo real se está ejecutando.

#### **1.4.1 Arquitectura de RTAI-Lab**

El concepto básico de RTAI-Lab es permitir que dos sistemas separados, el *host* y el *target*, puedan comunicarse. En una implementación remota, el sistema *host* es el PC donde la interfaz grafica de usuario (GUI) del RTAI-Lab se ejecuta en tiempo real suave y el sistema *target* es el PC donde el código generado corre en tiempo real duro. En una configuración local, los dos procesos corren en el mismo hardware comunicándose por medio de mensajes locales. La interfaz remota se implementa

explotando las características de la pequeña y efectiva capa *middleware* (un software de conectividad de aplicaciones distribuidas) de tiempo real de RTAI, llamado *net\_rpc*. El *host* envía y recibe mensajes a través de *net\_rpc* solicitando al *target* aceptar los cambios en los parámetros y al tiempo enviar los datos obtenidos, para mostrarlos gráficamente o guardarlos en un archivo de registro.

En RTAI-Lab la capa que comunica el sistema de simulación con el código generado en tiempo real, está integrada totalmente dentro del mismo sistema, por tal razón es posible correr al mismo tiempo varias sesiones de GUI, con el objetivo de monitorear un conjunto de *targets* simultáneamente, permitiendo además la comunicación con otros PCs en una red de control distribuido. La interoperabilidad entre el uso local, remoto y distribuido, es posible a través del establecimiento de identificadores que permiten distinguir y separar los correspondientes entornos de las secciones de RTAI-Lab que se estén ejecutando en un mismo *host*.

La creación del sistema de control y generación del código asociado al sistema de control se realiza en tres pasos (Bucher y Dozio, 2003):

1. El diseñador crea un diagrama de bloques en Simulink o Scicos que represente la estrategia de control elegida (prototipo de control). Este diagrama está compuesto por bloques del software de modelamiento matemático, bloques específicos de RTAI y bloques de DAQ que proporcionan la interfaz de E/S para el control de RTAI-Lab.
2. En la generación del código C para la tarea en tiempo real, el usuario sólo tiene que elegir el lenguaje de compilación para que el código se genere automáticamente con un *click*.
3. El ejecutable permite elegir opciones del modo de tiempo real, el periodo de temporización y el tiempo de ejecución final.

El controlador generado para la aplicación de tiempo real típicamente corre en el espacio de usuario de un computador estándar, hoy en día es típico usar RCP basados en Scilab/Scicos y RTAI el cual soporta las siguientes arquitecturas de Hardware: x86, x86\_64, PowerPC y ARM (Skiba et al, 2006).

#### **1.4.2 Paleta de RTAI-Lib**

El desarrollo de software en tiempo real se puede realizar mediante la paleta RTAI-Lib que se instala en el editor de diagramas de bloques Scicos. Algunas pruebas de los ejecutables en tiempo real se pueden hacer con Xrtailab y nuevos bloques de Scicos pueden ser programados utilizando el lenguaje Scilab (actualmente con Scicoslab), mientras su componente ejecutable es programado en lenguaje C.

La ilustración 1.3 muestra la paleta de RTAI-Lib y sus principales componentes, entre los cuales se encuentran bloques de entradas, salidas, buzones de mensajes, interfaces de hardware y semáforos.

Sus bloques corresponden a (Bucher et al, 2010):

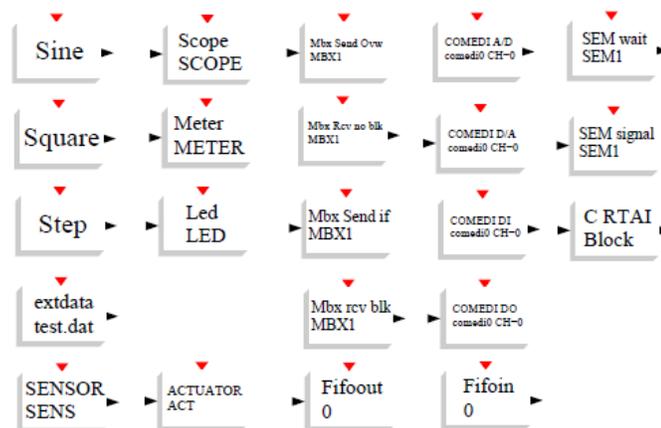
Entradas:

- *Sine*: Un generador de onda seno en tiempo real.
- *Square*: Un generador de señal cuadrada en tiempo real.
- *Step*: Un generador de señal escalón en tiempo real.
- *Extdata*: Carga datos desde un archivo con extensión .dat.
- *SENSOR*: Una entrada de sensor genérica, donde puede ajustarse el número de entradas y se configurado a través de código C.

Salidas:

- *Scope*: Un osciloscopio multicanal del Xrtailab.
- *Meter*: Un medidor simple del Xrtailab.
- *LED*: Un bloque multicanal de *leds*.
- *Actuator*: Una salida de actuador genérica similar al bloque sensor.

### Ilustración 1.3: Paleta de RTAI-Lib



Fuente Propia

Buzón de mensajes:

- *Mbx Send Ovw*: Envía un mensaje al *mailbox* sobre escribiendo sus datos.
- *Mbx Rcv no blk*: Recibe un mensaje solo si el mensaje completo puede pasar sin ser bloqueado por la tarea llamada.
- *Mbx Rcv blk*: Recibe un mensaje. La tarea llamada será bloqueada hasta que todos los bytes del mensaje sean transmitidos u ocurra un error.
- *Mbx Send if*: Envía un mensaje solo si la totalidad de éste puede pasar sin ser bloqueado por la tarea llamada.
- *Fifoout* y *Fifoin*: *Fifo* multicanal, donde puede ser ajustadas el número de entradas.

Controladores hardware de Comedi:

- *Comedi A/D*: Soporte para entradas analógicas.
- *Comedi D/A*: Soporte para salidas analógicas.
- *Comedi DI*: Soporte para entradas digitales.
- *Comedi DO*: Soporte para salidas digitales.

Semáforos y otros:

- *SEM wait*: Decrece el valor de un semáforo y espera por un evento en una señal, la llamada a la tarea es bloqueada mientras el valor del semáforo permanece en negativo.
- *SEM signal*: Incremente el valor de un semáforo, si el valor resultante no es positivo entonces la primera tarea en el semáforo espera para ser ejecutada.
- *C RTAI*: Este es un bloque genérico que permite editar en código C.

## 1.5 RESUMEN

En este capítulo se explicaron conceptos fundamentales relacionados con los Sistemas de Prototipado Rápido de Control, se mencionaron las principales características de estos sistemas y la influencia que tienen en diferentes áreas o campos del control industrial. Se mencionó un procedimiento general para diseñar e implementar sistemas de control utilizando este concepto y se describieron los diferentes enfoques de RCP existentes, su estructura y las ventajas y desventajas de estos sistemas.

En este capítulo también se mencionaron los sistemas operativos de tiempo real (RTOS), como una herramienta que permite aplicar el concepto de RCP, enfatizando el uso del conjunto de herramientas que se incluyen en RTAI-Lab, del cual se hace un análisis detallado, explicando su concepto, sus principales componentes y la forma como este puede utilizarse.

## 2 INTERFAZ DE COMUNICACIÓN ENTRE EL MÓDULO DE TEMPERATURA MIC 955 DE LA FEEDBACK Y RTAI-LAB.

En este capítulo se describen los componentes hardware y software que permiten la comunicación entre el módulo MIC 955 y el conjunto de herramientas incluidas en RTAI-Lab, el cual fue instalado sobre el sistema operativo GNU Linux Fedora 7. En el anexo A y el anexo B se detalla el procedimiento de instalación del sistema operativo GNU Linux Fedora 7 y RTAI-Lab respectivamente.

### 2.1 CARACTERÍSTICAS DEL MÓDULO MIC 955 DE LA FEEDBACK

El módulo MIC 955 pertenece a los módulos de la serie MICA comercializados por la empresa inglesa *Feedback Instruments Ltda*, lanzados al mercado en 1981 y diseñados para la enseñanza en aplicaciones de control informático por medio de PC's de diferentes marcas (Feedback, 2010). Este módulo didáctico portable, es una planta térmica elemental que fue diseñada originalmente para prácticas como: medición digital de temperatura, manipulación de potencia eléctrica y diseño de lazos de control retroalimentado sencillos.

#### 2.1.1 Descripción física del Módulo MIC 955

Básicamente el módulo MIC 955 está compuesto por dos unidades, tal como se muestra en la Ilustración 2.1:

- Un panel compuesto por una tira de aluminio negro anodizado (*Workpiece*), dos resistencias que sirven de calefactores (*Heater*), las cuales pueden ser activadas desde un PC y una resistencia de banda de platino al aguafuerte que es utilizada como sensor de temperatura.
- Un ventilador refrigerador (*Blower*) que tiene tres modos de configuración (programado, desactivado y manual), los cuales pueden activarse mediante un interruptor (*switch*). En modo manual la velocidad del ventilador varía con el movimiento de un potenciómetro (*Blower Speed*). En modo programado el ventilador se activado desde el PC, a través de un bit conectado a un amplificador, entregando mayor potencia que en el modo manual (Feedback, 1983).

El sensor de temperatura está conectado a un circuito puente de Wheatstone DC. La función de este sensor es convertir las variaciones de resistencia a variaciones de voltaje, las cuales que serán amplificadas y posteriormente equilibradas a 0°C, obteniendo una sensibilidad de 20mV/°C. Ésta sensibilidad puede ser monitoreada



El módulo MIC 955 cuenta con dos puertos de E/S:

- El puerto A envía el valor de la temperatura entregado por el conversor ADC en una palabra de 8 bits.
- El puerto B recibe los comandos de activación de las señales para los calentadores, el ventilador en modo programado y del conversor ADC.

La Tabla 2.1 muestra las asignaciones de puerto y de bits para realizar las comunicaciones entre el PC y el conector FC-26P macho del módulo.

**Tabla 2.1: Configuración de puertos del módulo MIC 955. Modificado de (Feedback, 1983)**

Nº de bit	Puerto A (Entrada al PC)	Pin	Puerto B (Salida del PC)	Pin
0 (LSB)	ADC salida del conversor analógico-digital, bit 0	21		
1	ADC salida del conversor analógico-digital, bit 1	20		
2	ADC salida del conversor analógico-digital, bit 2	19		
3	ADC salida del conversor analógico-digital, bit 3	18		
4	ADC salida del conversor analógico-digital, bit 4	17		
5	ADC salida del conversor analógico-digital, bit 5	16	Señal de ventilador (activado = 1)	3
6	ADC salida del conversor analógico-digital, bit 6	15	Señal de calentador (activado = 1)	2
7(MSB)	ADC salida del conversor analógico-digital, bit 7	14	Inicio de conversión de ADC	1

Adicionalmente el módulo MIC 955 tiene dos salidas a tierra en los pines 13, 26 y una señal de fin de conversión generada por el ADC en el pin 23 del conector FC26-P.

### 2.1.2 Configuración y comunicación entre el Módulo MIC 955 y el PC.

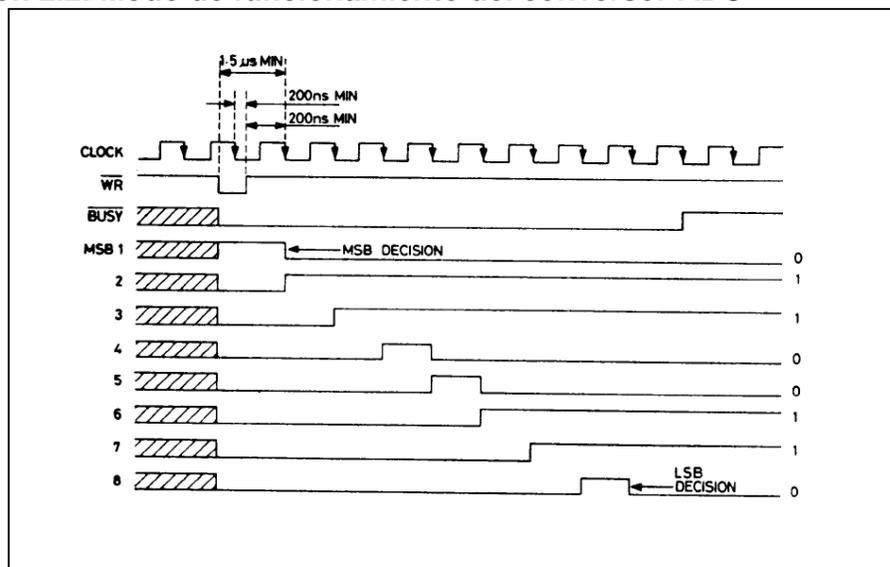
Para realizar la conexión con la tarjeta Feedback 2-926-12050A, el bus ISA implementa una aplicación para tarjetas prototipos (*Prototype Adapter*) cuyo rango de direcciones varía entre 300h y 31Fh. La tarjeta Feedback 2-926-12050A tiene asignadas las direcciones dadas en la Tabla 2.2, estas deben ser configuradas por el usuario desde una plataforma de programación.

**Tabla 2.2: Configuración de puertos del módulo MIC 955. Fuente propia**

Nombre	Dirección	Descripción
Puerto A	300h	Registro de lectura que recibe los 8 bits de conversión del ADC
Puerto B	300h +1	Registro de escritura encargado de enviar las señales de activación del calentador, ventilador y de inicio de conversión.
Registro de Control	300h +3	Encargado de configurar la tarjeta de interfaz en modo bidireccional, esto se realiza enviando el valor hexadecimal 99h

Para iniciar la conversión AD, el bit 7 del Puerto B (Pin WR del convertor) debe recibir un impulso positivo generado por una secuencia binaria 1 – 0 – 1. Cuando se recibe este impulso, el pin BUSY del convertor cambia a bajo durante la conversión AD, la cual tiene una duración aproximada de 10 ciclos de reloj. Como se muestra en la Ilustración 2.2, en la conversión AD se le da prioridad al bit de mayor peso (MSB). Una vez la conversión termina, el pin BUSY del convertor cambia a su estado alto. En ese momento puede realizarse la lectura del Puerto A para una toma de datos exitosa. Es importante mencionar que las salidas y las entradas del convertor son compatibles con el estándar TTL (GEC PLESSEY, 1994).

**Ilustración 2.2: Modo de funcionamiento del convertor ADC**



Fuente (GEC PLESSEY, 1994)

Los algoritmos de control originales del módulo MIC 955 se basan en programación estructurada y están escritos en lenguaje Basic o en código de máquina (dependiendo del PC en uso). Estos programas se describen en el manual de usuario del módulo MIC 955 y tienen como fin la realización de tres prácticas de laboratorio:

1. Práctica 1: “Termómetro digital”. Esta práctica consiste en leer y visualizar la temperatura leída por el sensor.
2. Práctica 2: “ciclo límite”. Esta práctica comprende un control de temperatura ON/OFF con histéresis. Los límites superior e inferior de la banda de histéresis se establecen por medio del teclado.
3. Práctica 3: “controlador proporcional”. Esta práctica se realiza a través de modulación por ancho de pulso PWM (Feedback, 1983).

Estos programas presentan una interfaz de usuario deficiente, no permiten la toma de datos precisa ni el almacenamiento adecuado de estos, además el manejo de parámetros y la visualización de variables es limitada. Otra desventaja es que para modificar estos algoritmos se debe tener un conocimiento avanzado del lenguaje de programación e incluso es complejo realizar la modificación de los mismos, pues se pueden presentar errores en la ejecución.

## 2.2 EL PUERTO PARALELO

Con la introducción de la computadora personal de IBM en 1981, se agregó el puerto paralelo como una alternativa al puerto serie utilizándolo como controlador de impresoras de matriz de punto de alto rendimiento. Éste contaba con la capacidad de realizar la transferencia simultánea de todos los bits que constituyen el byte o palabra de datos (manteniendo el último valor establecido en su salida) mientras que el puerto serie lo realizaba de bit en bit. En un principio, el puerto paralelo se tenía como una opción adecuada para el manejo de dispositivos de alta velocidad, sin embargo, con paso del tiempo, la velocidad de transferencia del puerto paralelo se vió limitada a una transferencia promedio de 150 kbytes/seg, pues era dependiente del software con el cual era utilizado. Otras desventajas fueron la inexistencia de un estándar para la interfaz eléctrica, que garantizara la operación en múltiples plataformas y la distancia de cables externos de transmisión que estaba limitada a un máximo de 1,8 metros. Tiempo después, el puerto paralelo se modificó para hacerle más veloz, creando el puerto bidireccional compatible con PS/2, el cual permite una comunicación bidireccional. Posteriormente fueron lanzados al mercado el *Enhanced Parallel Port* (EPP) y el *Extended Capability Port* (ECP) (ETI, 2011).

Actualmente el puerto paralelo se basa en el estándar IEEE 1284, liberado en 1994, y normalmente es utilizado para la conexión con impresoras. También se ha utilizado en programadores EPROM, escáneres, interfaces de red Ethernet a 10 MB, unidades ZIP, Súper disk y para comunicación entre dos PCs (Fundación Rutten 2010).

En la actualidad, el puerto paralelo soporta 4 modos de operación (Herrera, 2009):

- Puerto Paralelo Estándar (SPP).
- Puerto Paralelo PS/2 (Bidireccional).
- Puerto Paralelo Mejorado (EPP).
- Puerto Paralelo con Capacidades Extendidas (ECP).

En la Tabla 2.3 se aprecian las características de cada configuración del puerto:

**Tabla 2.3: Tipos de Puerto Paralelo. Modificado de (ETI, 2011)**

Características Principales	Tipos de Puerto Paralelo			
	SPP	PS/2	EPP	ECP
Fecha de Introducción	1981	1987	1994	1994
Fabricante	IBM	IBM	Intel, Xircom y Zenith Data Systems	Hewlett Packard y Microsoft
Bidireccional	No	Si	Si	Si
Velocidad	150 Kbyte/seg	150 Kytes/seg	2 Mbytes/seg	2 Mbytes /seg

### 2.2.1 Descripción General del Puerto Paralelo

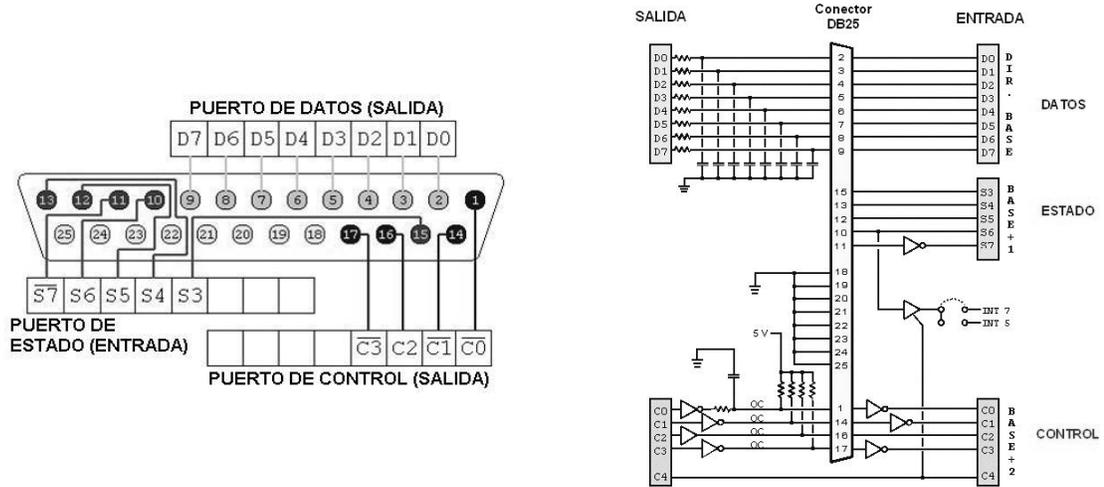
El computador posee un conector hembra DB25, el cual está conformado por 17 líneas de señales, de las cuales 5 son entradas correspondientes a las líneas de estado, 12 son salidas tipo *latch* (con memoria), que corresponden a 8 líneas de datos y 4 señales de control. Además cuenta con 8 líneas a tierra, de las cuales 7 sirven como masa proporcionando apantallamiento y protección a interferencias.

El puerto paralelo maneja el estándar LS-TTL, donde la tensión de salida en alto es igual a 5v y en bajo igual a 0 v. La corriente de salida máxima es de 2.6mA y la de entrada es de 24mA, aunque hay algunas que funcionan en colector abierto (Gallego, 1998).

En la Ilustración 2.3a se aprecia la distribución física de los pines en el conector DB-25. Por ejemplo, en el caso del registro de estado, se tienen disponible las señales a partir del bit 4 y hasta el bit 8 (S7, S6, S5, S4, S3), los demás bits están comprometidos o reservados para otros propósitos. La ilustración 2.3b muestra que el bit más significativo del registro de estado (S7) trabaja con lógica negativa y está físicamente ubicado en el pin 11 del conector. Si se requiere leer una palabra de entrada a través de este registro, es importante considerar con la lógica con la cual funciona cada pin (Herrera, 2009).

En cuanto al software, para fines de análisis, se considera que los tres registros de configuración del puerto: registro de datos, registro de estado y registro de control tiene 8 bits, por lo que se tiene un orden significativo el cual es necesario respetar cuando se forma una palabra de configuración. Estos ocupan tres direcciones de I/O consecutivas.

## Ilustración 2.3: El Puerto Paralelo



a) Conector DB25 hembra

b) Diagrama interno del LPT

Fuente (Herrera, 2009)

### 2.2.2 Acceso al Puerto Paralelo

Para acceder al puerto, la dirección base del registro llamada comúnmente BASE, puede tomar el valor hexadecimal: 0x378, 0x278, 0x3bc dependiendo del puerto a utilizar (/dev/lp0, /dev/lp1, /dev/lp2)(Capó, 2002). La definición de los demás registros se hace con relación a la dirección de I/O BASE:

- BASE+0: registro de datos.
- BASE+1: registro de estado.
- BASE+2: registro de control.

**Registro de datos:** Controla los bits de Dato 0 a Dato 7, correspondientes en ese orden a los pines del 2 al 9 del conector DB25. Trabajan con lógica positiva. Normalmente se utilizan para escritura de datos, sin embargo pueden ser usados como puertos de lectura configurando el puerto de manera bidireccional, en caso no ser así la lectura del puerto devolverá el último valor escrito.

**Registro de estado:** Se utiliza sólo para lectura (las escrituras son ignoradas). Su lectura da el estado de los 5 pines de entrada en el momento en que la orden es ejecutada. Se identifican por su nombre en inglés de la siguiente manera: S7 *Busy* (pin 11), S6 *Ack* (pin 10), S5 *No paper* (pin 12), S4 *Selected* (pin 13), S3 *Error* (pin 15). Trabajan con lógica positiva a excepción del bit *Busy*, el cual trabaja con lógica negativa y tiene una resistencia *pull-up* interna.

**Registro de Control:** Se utiliza para lectura y escritura. Los cuatro bits inferiores, C0 *Select* (pin 1), C1 *Auto Feed* (pin 14), C2 *Initialize* (pin 16) y C3 *Select* (pin 17) son salidas TTL a colector abierto con resistencias de *pull-up* de 4700 ohms. Un dispositivo externo puede forzar el estado de los pines sin dañar el puerto, esto

permite utilizarlos como entradas. Para hacer esto, se coloca en alto las cuatro salidas (escribiendo 0000100b en BASE+2), lo cual hace que las salidas “floten”. Los bits C3, C1 y C0 trabajan con lógica negativa.

**Bit de puerto bidireccional (compatible PS/2):** El bit C5 de registro de control, está disponible sólo si se trata de un puerto bidireccional. Si C5 es igual a 1, el buffer de los datos de salida se pone en alta impedancia, desconectándolo de los pines 2 a 9 del registro de datos. Al escribir en el registro de datos se modifica el valor en el buffer pero no en la salida. Esto permite que al leer el puerto, se lea el estado de las entradas y no lo que hay en buffer. Para habilitar el puerto paralelo en computadoras IBM PS/2, además de hacer lo anterior, se debe poner el bit 7 del registro del puerto 102h en 1.

**Bit de interrupción:** Normalmente un PC dispone de 16 interrupciones hardware, que se conocen comúnmente como IRQ. El puerto paralelo utiliza las líneas IRQ 5 y 7 en el manejo de impresoras y otros dispositivos. Para habilitar la interrupción es necesario colocar el bit C4 del registro de control en 1. La interrupción se produce cuando exista un flanco descendente en el pin 10, correspondiente al bit S6 del registro de estado. Para que un programa maneje una interrupción debe seguir los siguientes pasos:

1. Al iniciar:

- Modificar vectores de la interrupción 0Fh (IRQ7).
- Permitir la interrupción IRQ7, estableciendo el bit 7 del puerto 021h en 0.

2. Durante la ejecución:

- Habilitar la interrupción en el registro de control, estableciendo el bit 4 en 1 siempre que sea necesario.
- Procesar la interrupción.
- Escribir el valor 020h en el puerto 020h.
- Inhibir la interrupción, estableciendo el bit 4 del puerto de control en 0.
- Repetir.

3. Al terminar:

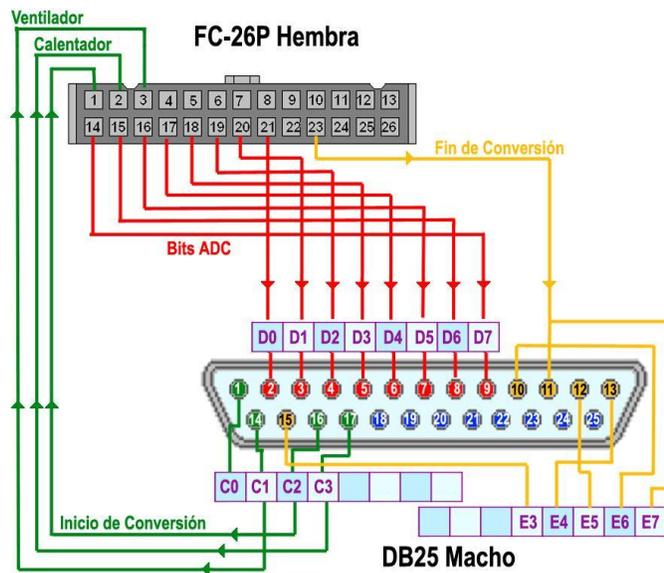
- Inhabilitar la IRQ7, estableciendo el bit 7 del puerto 021h en 1.
- Recuperar los vectores de interrupción anteriores.

### **2.3 CONEXIÓN ENTRE EL MIC 955 Y EL PUERTO PARALELO**

El puerto paralelo y la tarjeta Feedback 2-926-12050A del módulo MIC 955 trabajan con el estándar TTL, por lo tanto potencialmente existe la posibilidad de realizar un cableado directo entre el conector hembra DB25 del PC y el conector macho FC-26P

del módulo MIC955, sin la necesidad de diseñar algún tipo de dispositivo de adaptación de corrientes o voltajes. Sin embargo, el orden y funciones de las señales IN/OUT del módulo MIC955 presentes en el conector FC-26P no coinciden con las señales y funciones de los puertos del puerto paralelo del conector DB25, por lo que la conexión debe realizarse teniendo en cuenta algún tipo de mapeo entre las señales. En la ilustración 2.4 se muestra el mapeo de señales entre el puerto paralelo y el módulo MIC 955.

**Ilustración 2.4: Mapeo de señales Puerto Paralelo – Módulo MIC 955**



Fuente: Propia

**Salidas del MIC955:** La salida del ADC, asociada a la señal temperatura, tiene 8 bits y el registro de estado del puerto paralelo (normalmente usado para la lectura) solamente tiene 5 bits, por lo tanto no es posible hacer una conexión directa entre ellos. Para superar este inconveniente, se eligió utilizar el registro de datos del puerto paralelo para leer la salida del conversor, configurándolo previamente en modo bidireccional, sin embargo, la prioridad de los bits de salida del módulo MIC955 tiene orden descendente en relación con su número de pin, siendo el pin 21 el de menor peso y el pin 14 el de mayor peso, por esta razón la conexión tenía que implementarse de forma inversa, haciendo coincidir los pines del 14 al 21 del conector FC-26P correspondientemente con los pines del 9 al 2 del conector DB25.

**Entradas del MIC955:** El módulo MIC 955 requiere de 3 entradas que corresponden a las señales de activación del calentador, del ventilador y de inicio de conversión. Estas señales fueron acopladas a 3 de los 4 bits del registro de control de la siguiente manera:

- Ventilador : Control 1 (*Auto Feed*).
- Calentador: Control 3 (*Select*).
- Inicio de conversión: Control 2 (*Initialize*).

Finalmente la señal de fin de conversión del módulo MIC955 (pin 23 del conector FC-26P) correspondiente a *Busy* en el conversor ADC, se conectó al pin 11 correspondiente a la señal de *Busy* en el registro de estado del puerto paralelo.

La Tabla 2.4 muestra el orden en el cual el conector macho del puerto paralelo, se conecta al conector hembra FC-26P. Además de las conexiones mencionadas anteriormente se escogieron dos pines a tierras del puerto para ser conectados en los pines 13 y 26 de Módulo MIC 955 también correspondientes a tierra. Aunque no se utilizan todos los pines, estos se conectaron para hacer el cable más resistente y mejorar su presentación.

**Tabla 2.4: Mapeo de señales Puerto Paralelo – Módulo MIC 955. Fuente propia.**

DB25			FC-26P		
Pin	Descripción		Color cable	Pin	Descripción
1	<i>Strobe</i>	C0	Negro	4	
2	Dato 0	D0	Negro - Blanco	21	ADC Bit 0 LSB
3	Dato 1	D1	Café	20	ADC Bit 1
4	Dato 2	D2	Café - Blanco	19	ADC Bit 2
5	Dato 3	D3	Rojo	18	ADC Bit 3
6	Dato 4	D4	Rojo - Negro	17	ADC Bit 4
7	Dato 5	D5	Rojo - Blanco.	16	ADC Bit 5
8	Dato 6	D6	Naranja	15	ADC Bit 6
9	Dato 7	D7	Naranja - Negro	14	ADC Bit 7 MSB
10	<i>Ack</i>	S6	Naranja - Blanco	6	
11	<i>Busy</i>	S7	Amarillo	23	Fin de conversión
12	<i>No paper</i>	S5	Amarillo - Negro	7	
13	<i>Selected</i>	S4	Verde	8	
14	<i>Auto Feed</i>	C1	Verde - Negro	3	Ventilador
15	<i>Error</i>	S3	Verde - Blanco	5	
16	<i>Initialize</i>	C2	Azul	1	Inicio de conversión
17	<i>Select</i>	C3	Azul - Blanco	2	Calentador
18	Tierra		Violeta	22	
19			Violeta - Blanco	9	
20			Gris	10	
21			Gris - Negro	11	
22			Rosa	24	
23			Rosa - Negro	12	
24			Verde claro	13	Tierra
25			Rojo - Rosa	26	
Masa				Blanco	25

En la Ilustración 2.5a se muestra los conectores FC-26P y DB-25 utilizados y en la Ilustración 2.5b se muestra el montaje final del nuevo cable de conexión

#### Ilustración 2.5: Cable de conexión MIC955 - LPT



a) Conectores FC 26P y DB25



b) Cable completo

Fuente: Propia

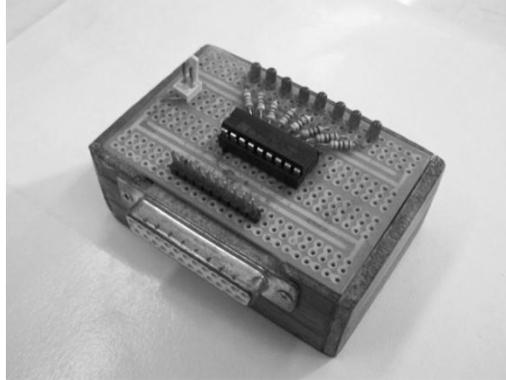
## 2.4 COMUNICACIÓN ENTRE EL MIC 955 Y EL PUERTO PARALELO

El nuevo cable de conexión permite la comunicación directa de la señales entre el módulo MIC 955 y el puerto paralelo del PC, por lo que la tarjeta de interfaz Feedback 2-926-12050A deja de ser necesaria en la nueva implementación. Sin embargo, es necesario realizar la configuración y el manejo de las señales que permitan generar la lectura exitosa de la salida del ADC y el control de los diferentes componentes programables del módulo. Esta función se delega al conjunto de herramientas de RTAI-Lab.

Durante el proceso de instalación del *kernel* en tiempo real es obligatorio desinstalar los controladores estándar de Linux que manejan las funciones del puerto paralelo, con el fin de garantizar su funcionamiento en las condiciones temporales requeridas por la nueva implementación sin que genere conflictos en su ejecución. Los controladores de tiempo real necesarios son proporcionados por las librerías de Comedi y son llamados *comedi\_parport*, estos se cargan y se configuran con la dirección base 0x378 al inicio de sesión del PC, junto con los módulos de tiempo real necesarios para el funcionamiento de RTAI-Lab (Bucher et al 2010).

Sin embargo antes de realizar la primera conexión entre el MIC 955 y el puerto paralelo fue necesario probar la funcionalidad total de éste último. Para ello se ensayó la configuración del puerto de salida mediante la conexión del registro de datos a un conjunto de 8 *leds*, a través de un IC ULN 2803A tal como se muestra en la Ilustración 2.6. Se utilizó el código *comedi\_dioin*, el cual se incluye en la librería de Comedi-lib. Una vez que la prueba fue exitosa, se procedió a configurar el puerto para entrada de datos. Inicialmente se utilizó el mismo circuito y se ejecutó el código *comedi\_dioout* incluido también en la librería Comedi-lib, esperando que los bits del registro de datos se establecieran en alto para comprobar el estado de alta impedancia típico de esta configuración.

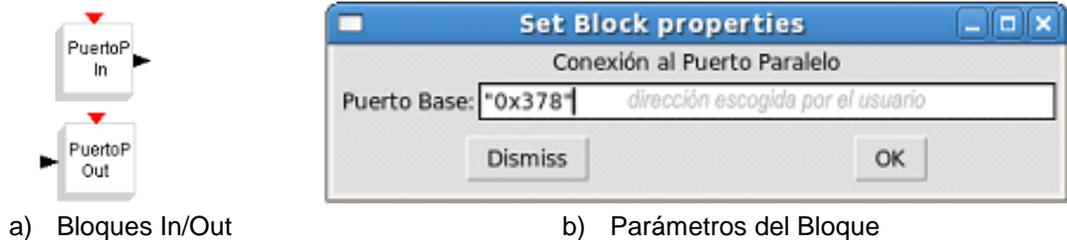
### Ilustración 2.6: Circuito de prueba de la palabra de datos del LPT



Fuente: Propia

Los controladores *comedi\_parport* proveen las herramientas básicas de comunicación con el puerto paralelo, sin embargo, su simplicidad hace que la comunicación con el MIC 955 a través de Scicos sea difícil de implementar con los bloques de entrada y salida hardware de Comedi incluidos en la paleta de RTAI-Lib. Por esta razón se propuso la creación de nuevos bloques, que permitieran conectar las tareas de tiempo real compiladas en Scilab con el puerto paralelo. Estos bloques prototipo fueron llamados *PuertoPOut* y *PuertoPIn*. La principal función de estos bloques es escribir y leer un dato a una dirección de registro del puerto paralelo, siendo 0x378 el registro de datos, 0x379 el registro de estado y 0x37A el registro de control como se muestra en la Ilustración 2.7.

### Ilustración 2.7: Bloques PuertoPIn y PuertoPOut

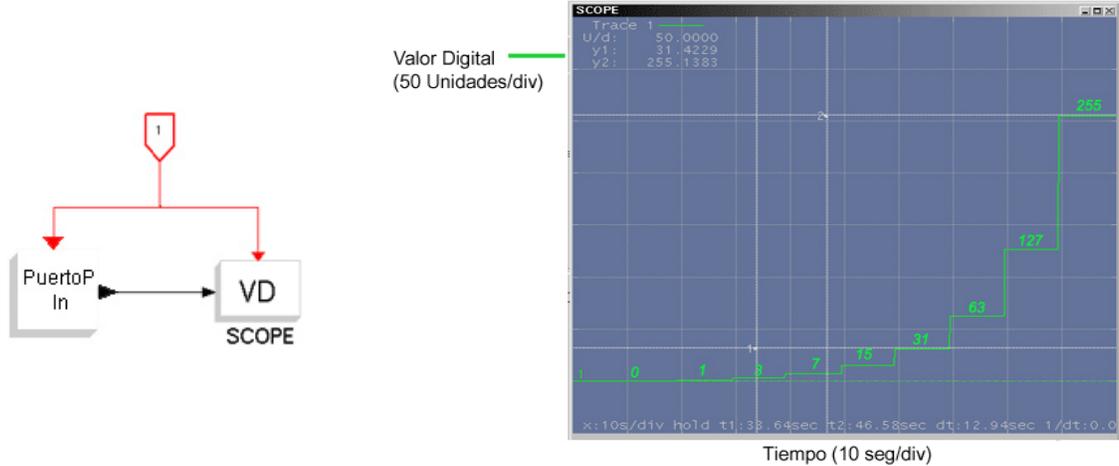


Fuente: Propia

En el anexo C se muestra el procedimiento detallado para crear estos bloques de tiempo real en Scilab/Scicos, la descripción funcional básica, el modo de uso, limitaciones, montaje y otras características.

Con la implementación de estos bloques básicos prototipo se realizó nuevamente la prueba de escritura obteniendo resultados satisfactorios y se procedió a realizar una prueba de lectura ingresando valores utilizando resistencias pull-up conectadas al registro de datos. La ilustración 2.8 muestra el funcionamiento de esta configuración y el diagrama de bloques utilizado, como se puede observar la implementación toma valores enteros de forma exitosa y los muestra en el osciloscopio virtual de Xrtailab.

## Ilustración 2.8: Conexión Puerto Paralelo- Xrtailab



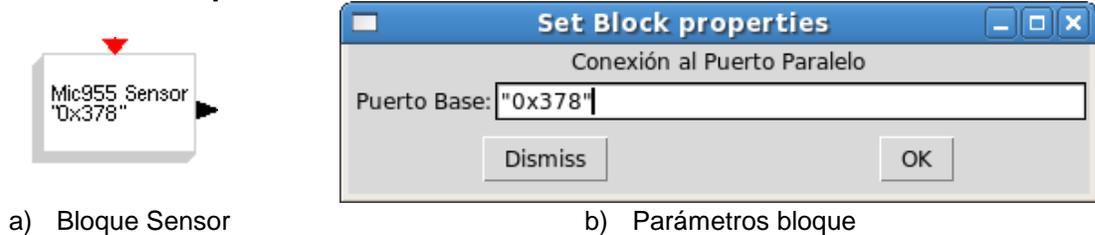
Fuente: Propia

Finalmente, para dejar el puerto paralelo listo para la conexión con el módulo MIC 955, debe tenerse en cuenta el estado normal de los registros utilizados, poniendo especial cuidado en los bits C1 y C3 del registro de control, ya que se encuentran habitualmente en estado alto por su lógica negativa, lo que ocasionaría que el ventilador y el calentador estuvieran encendidos mientras no se ejecute una tarea de control. Para solucionar este problema, se envía el valor binario 00001010b por medio de un ejecutable nombrado *Load*, el cual se carga al inicio de sesión del PC junto a los módulos de RTAI-Lab y los controladores del paralelo. Por esta razón no es conveniente encender el regulador de voltaje del MIC955 antes de haber iniciado la sesión de trabajo en RTAI-Lab. Nótese que el registro de datos aún se mantiene configurado en modo de escritura, con el fin de proteger el puerto en caso de cualquier eventualidad hasta que la tarea de tiempo real requerida sea ejecutada.

### 2.4.1 Sensor de Temperatura MIC 955.

La Ilustración 2.9 muestra el bloque MIC 955 Sensor, el cual cumple la función de entregar el valor entero generado por el ADC del módulo MIC 955, al registro de datos de la dirección base del puerto paralelo del PC, la cual es suministrada por el usuario a través de la ventana de configuración "Conexión al Puerto Paralelo"

## Ilustración 2.9: Bloque MIC 955 Sensor



Fuente: Propia

Para realizar la tarea de lectura del conversor, no basta con leer el valor escrito en el registro de datos, sino que también se debe enviar la señal de inicio de conversión en un periodo definido por cada pulso de reloj (*Event Clock* de Scicos), programado por el usuario. Además se debe configurar el registro de datos en alta impedancia para lectura, para ello es necesario el uso del registro de control: estableciendo el *bit* C5 en alto (1) y enviando al bit C2 *Initialize* (pin 16) una secuencia binaria 1-0-1, sin modificar cualquier otro *bit* de la palabra de control, con el fin de no intervenir en las acciones de control de los demás dispositivos conectados al mismo registro. El funcionamiento del sensor puede resumirse en tres etapas Inicialización, Lectura y Finalización las cuales se explican a continuación:

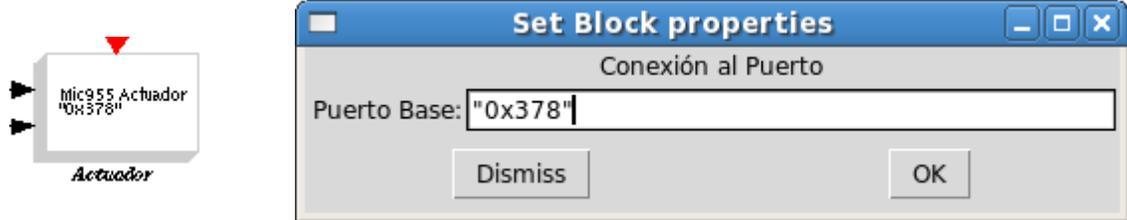
4. **Inicialización:** Al iniciar la tarea de tiempo real, el bloque MIC 955 Sensor lee el estado actual del registro de control y lo procesa mediante lógica “and” con el valor binario “11011011b”, lo que establece los valores de los *bits* C5 y C2 en bajo, sin importar sus valores anteriores, conservando el valor actual de los demás *bits*. Seguidamente se cambia el valor de C5 y C2 a uno (1) sumando “100100b”. La respuesta se envía a la salida del registro de control, lo que genera, gracias a C5=1, que el registro de datos pase a modo de lectura y por C2=1 que la señal de inicio de conversión este en uno (1). Nótese que también puede obtenerse el mismo resultado con un operador lógico “or” procesado con el valor “100100b”, sin embargo se prefirió esta forma con el fin de mantener la misma estructura en todas las etapas tanto del bloque sensor, como del bloque actuador.
5. **Lectura:** Se ejecuta con cada salto de la temporización de un reloj en Scicos, inicia leyendo el valor actual del registro de control y lo procesa mediante lógica “and” con el valor binario “11111011b”, con esto se establece el *bit* C2, (señal de inicio de conversión) en bajo, el resultado se envía a la salida del registro de control. Al resultado anterior se le suma el valor binario “100b”, estableciendo el *bit* C2 en alto y se vuelve a enviar el resultado al registro de control. Debido a que en la inicialización C2 se fijo en alto, con esta operación se envía la secuencia 1-0-1 necesaria para la activación del ADC. Como se conoce el tiempo estimado de conversión del ADC ZN427, se ejecuta un retardo de 50 microsegundos, tiempo suficiente para garantizar una lectura confiable del valor de conversión, la cual tarda aproximadamente 20 microsegundos en promedio. Al finalizar el retardo, el valor del registro de datos es leído y enviado a la salida del bloque MIC 955 Sensor.
6. **Finalización:** Se ejecuta al detener correctamente la tarea de tiempo real, coloca el estado del registro de control en su estado original, procesando su valor actual mediante un operador “and” con el valor binario “11011011b” colocando en bajo los *bits* C5 y C2, correspondientes a la configuración del puerto bidireccional y a la señal de inicio de conversión.

#### 2.4.2 Bloque MIC 955 Actuador

La función del bloque MIC 955 Actuador es la de enviar el valor de las señales correspondientes al calentador y al ventilador desde el RTAI-Lab hasta el MIC 955 a través del registro de control del puerto paralelo del PC. En la Ilustración 2.10 se

muestra el bloque y sus propiedades que pueden ser configuradas por el usuario para el uso de otras direcciones base del puerto paralelo.

#### Ilustración 2.40: Bloque MIC 955 Actuador



a) Bloque Actuador

b) Parámetros del bloque

Fuente: Propia

Al igual que con el bloque sensor, el funcionamiento del actuador no debe modificar ningún otro *bit* de la palabra de control, fuera de los asignados al calentador (C3) y al ventilador (C1), con el fin de no intervenir en las acciones de control de los demás dispositivos. Su funcionamiento puede resumirse en tres etapas Inicialización, Escritura y Finalización las cuales se explican a continuación:

7. **Inicialización:** Al iniciar la tarea de tiempo real, el bloque MIC 955 Actuador lee el estado actual del registro de control y lo procesa mediante lógica “and” con el valor binario “11110101b”, lo que establece los valores de los bits C3 y C1 en bajo, sin importar sus valores anteriores, conservando el valor actual de los demás bits. A continuación se cambia el valor de C3 y C1 sumando “1010b” al estado actual. La respuesta se envía a la salida del registro de control asegurando que el ventilador y el calentador se encuentren apagados al inicio de la tarea de tiempo real.
8. **Escritura:** Se ejecuta con cada salto de la temporización de un reloj en Scicos y efectúa las siguientes operaciones:
  - Lee el valor actual del registro de control y lo procesa mediante lógica “and” con el valor binario “11110101b”, estableciendo los *bits* C3 y C1 en bajo. Este valor es llamado S0.
  - Lee la entrada calentador del bloque de Scicos, y si el valor correspondiente es menor que 1 se asigna el valor binario “1000b” a una variable llamada S1, en caso contrario, si la entrada del calentador es mayor o igual a 1 se asigna a S1 un valor igual a 0.
  - Lee entrada del ventilador, si esta es menor que 1 asigna el valor binario “10b” a una variable llamada S2, en caso contrario, si la entrada del ventilador es mayor o igual a 1 se asigna a S2 un valor igual a 0.
  - Suma los valores de las variables S0, S1 y S2, y envía la respuesta a la salida del registro de control.

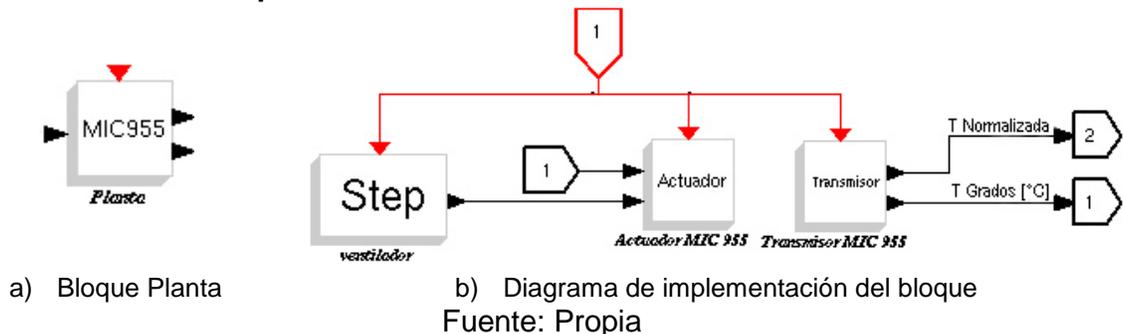
Cuando S1 y/o S2 (correspondientes a C3 y C1) tomen un valor igual a cero, se encenderá el dispositivo seleccionado ya que estos bits funcionan con lógica negativa.

9. **Finalización:** Se ejecuta al detener correctamente la tarea de tiempo real, establece el estado del registro de control en su estado original, procesando su valor actual mediante un operador “and” con el valor binario “11110101b” y sumando al resultado el valor binario “1010b”, lo que establece los bits C3 y C1 (sin importar su estado anterior) en alto, apagando el ventilador y el calentador.

## 2.5 ACONDICIONAMIENTO DE SEÑALES

El bloque *MIC955* se encarga de realizar la conexión en tiempo real entre el módulo MIC 955 y Scicos. La entrada a este bloque es la señal “Esfuerzo de Control” que admite un valor normalizado y como salidas tiene las señales “Temperatura en unidades de ingeniería” y “Temperatura normalizada” correspondientes a la variable controlada medida en grados centígrados y en un valor normalizado dependiente de los parámetros internos del bloque *transmisor*. Adicionalmente el bloque se temporiza mediante el generador de eventos de Scilab, marcado por una flecha roja que corresponde al tiempo de muestreo. En la Ilustración 2.11 se muestra el bloque *MIC955* y sus componentes: bloque *Actuador*, bloque *Trasmisor*, una función *Step* que es activada externamente para habilitar el ventilador y los respectivos puertos de entrada y salida.

**Ilustración 2.51: Bloque Planta MIC 955**



### 2.5.1 Transmisor de Temperatura

La función del bloque *Transmisor* de temperatura es recibir la señal generada por el sensor y convertirla en un valor normalizado apto para el funcionamiento del lazo realimentado. En la ilustración 2.12 se muestra el bloque *Transmisor* junto con su ventana de configuración, cuyos parámetros pueden ser modificados por el usuario.

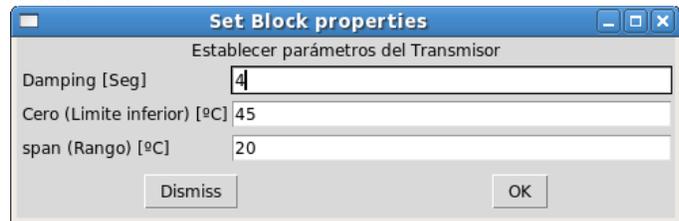
Como se mencionó anteriormente la salida del sensor de temperatura es una señal que varía entre los valores enteros 0 y 255 resultantes de la conversión de 8 bits del ADC ZN427. Esta contiene gran cantidad de ruido, el cual es generado por diferentes fuentes pertenecientes al sistema y externas, por lo tanto es necesario acondicionarla realizando funciones de linealización, conversión a unidades de ingeniería, filtrado y normalización. Las salidas resultantes son dos señales de temperatura: la primera en grados centígrados y la segunda en valores normalizados que dependen de los

parámetros *cero* y *span*, introducidos por el usuario. Adicionalmente el bloque contiene una entrada de temporización (flecha roja) que equivale al tiempo de muestreo establecido por el usuario.

**Ilustración 2.12: Bloque Transmisor MIC 955**



a) Bloque Transmisor

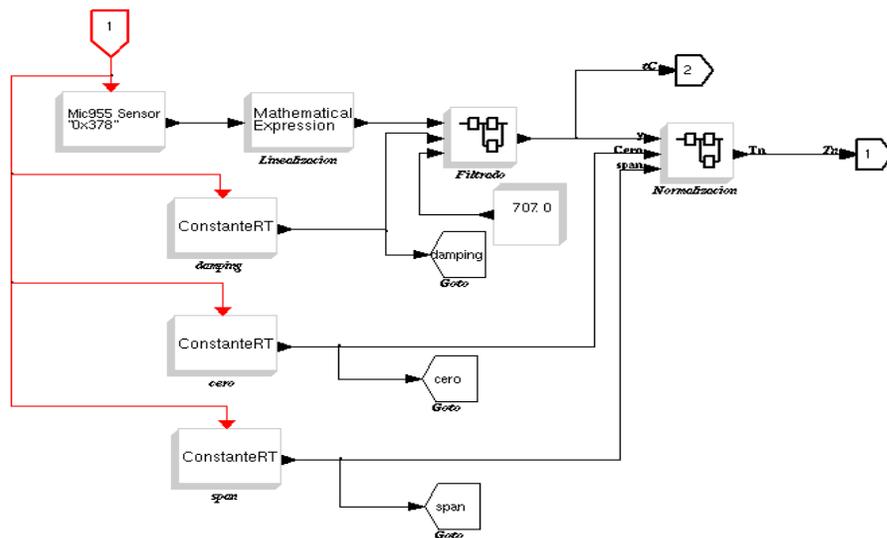


b) Parámetros del bloque

Fuente: Propia

En la Ilustración 2.13 se muestra la estructura interna del transmisor de temperatura, la cual está compuesta por el sensor de temperatura, bloques que entregan valores de los parámetros (que pueden ser modificados posteriormente por el usuario desde el xrtailab) y los bloques de Linealización, Filtrado y Normalización que se explicados a continuación y se detallan en el Anexo D:

**Ilustración 2.13: Estructura interna del Transmisor MIC 955**



Fuente: Propia

**Linealización:** Se realiza en el bloque *Mathematical Expression* y permite convertir la salida del sensor en unidades de ingeniería (grados centígrados) por medio de una ecuación de tercer orden, la cual se obtuvo mediante el proceso de toma de datos de la planta y el cálculo del polinomio por medio de un código en Scilab. La tabla con los datos tomados y el código utilizado para la linealización se detallan en el Anexo D.

**Filtrado:** Tiene como función eliminar cualquier sobre pico o ruido generado por los componentes físicos de la planta, para que no se vean reflejados sobre la variable medida. Esta tarea se realiza mediante un filtro pasa-bajos de segundo orden cuya frecuencia de corte, es suministrada por el inverso del parámetro Damping, el cual se mide segundos y es elegido por el usuario ( $F_c = 1/\text{Damping}$ ). Adicionalmente, como entrada al filtro se encuentra una constante equivalente al factor de amortiguamiento de la señal cuyo valor es igual a 0.707 (generando una respuesta subamortiguada) y sólo puede ser modificado por el usuario fuera de línea.

**Normalización:** se realiza a través de la implementación de la ecuación 2.1:

$$TN = \frac{y_{(t)} - \text{cero}}{\text{span}} \quad (2.1)$$

Donde:

$TN$  = Temperatura normalizada

$y_{(t)}$  = Temperatura en °C

$\text{cero}$  = límite inferior del rango de medición

$\text{span}$  = rango del transmisor

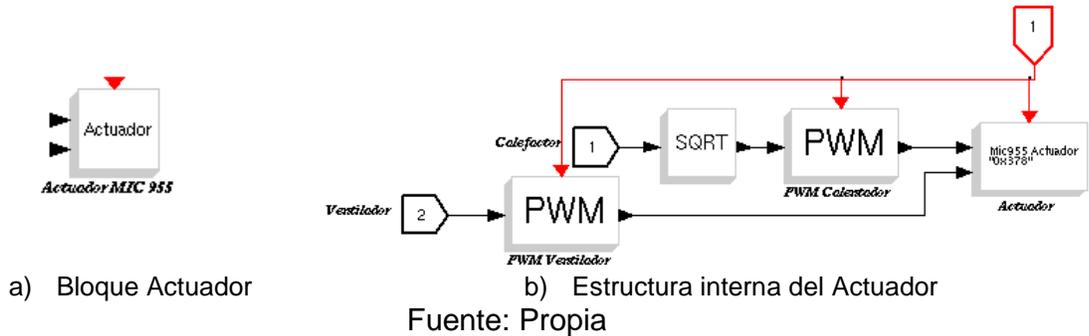
El  $\text{cero}$  es el límite inferior del rango de medición de temperatura, el  $\text{span}$  es la diferencia algebraica entre el límite superior y el límite inferior del rango de medición. En el caso del bloque Transmisor de temperatura los máximos límites son Temperatura ambiente y 86.7 grados centígrados. Los valores por defecto son  $\text{cero} = 45$  °C y  $\text{span} = 20$  °C.

## 2.5.2 Actuador

Se encarga de convertir la señal normalizada, recibida por el bloque *MIC955*, a una señal que permita desde el puerto LPT, realizar dos posibles acciones: Entregar el porcentaje adecuado de energía al proceso controlando la variable manipulada (potencia de entrada al calentador), con el fin de efectuar la acción de control o bien energizar el ventilador del MIC 955 para generar un disturbio.

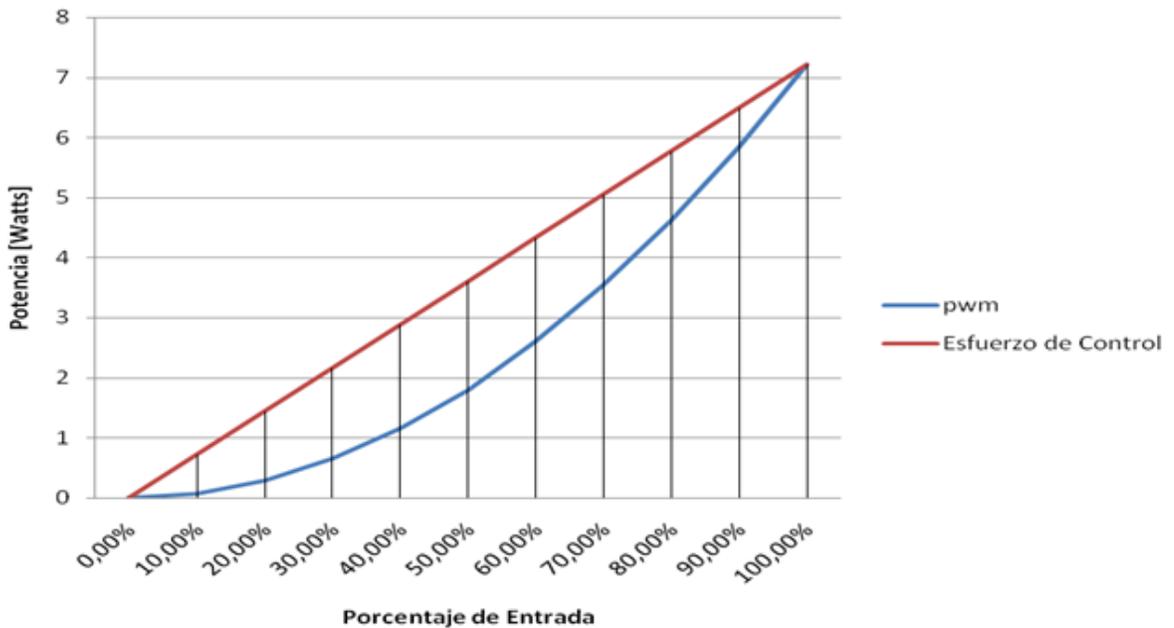
Debido a que los actuadores de la planta reciben señales todo o nada, se utiliza la técnica de modulación por ancho de pulso (PWM) con el objetivo de regular la potencia entregada a los calentadores del MIC 955 y activar con diferentes velocidades el ventilador. La Ilustración 2.14 muestra la estructura interna del bloque, donde las entradas del *MIC 955 Actuador* (calentador y ventilador) se conectan a diferentes bloques *PWM* los cuales permiten configurar individualmente cada dispositivo, dependiendo de sus características físicas, dinámicas y de las necesidades que requiere el usuario en la implementación del lazo de control.

### Ilustración 2.64: Actuador MIC 955



La función del bloque SQRT es la de garantizar linealidad entre el esfuerzo de control ( $E_c$ ) y la potencia entregada a los calentadores, haciendo el ajuste del porcentaje PWM necesario para que el actuador entregué la energía solicita por el controlador a la planta, realizando así la respectiva acción de control. La relación EC y PWM con respecto a la potencia es graficada en la Ilustración 2.15.

### Ilustración 2.74: Relación EC y PWM vs Potencia

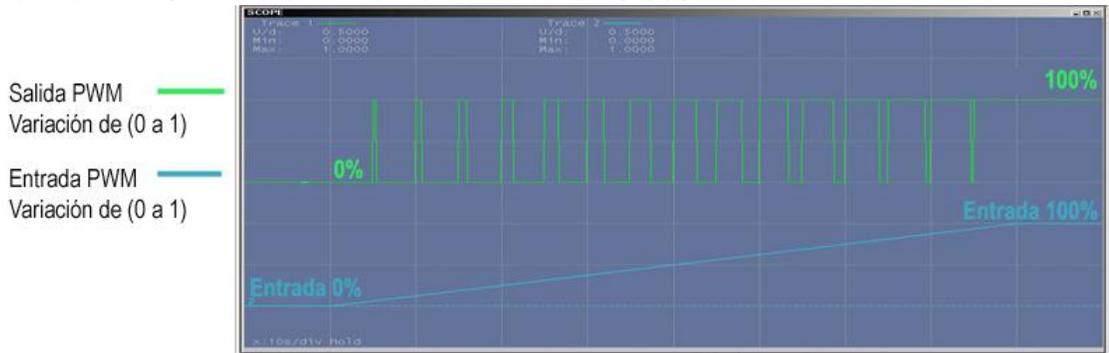


Fuente: Propia

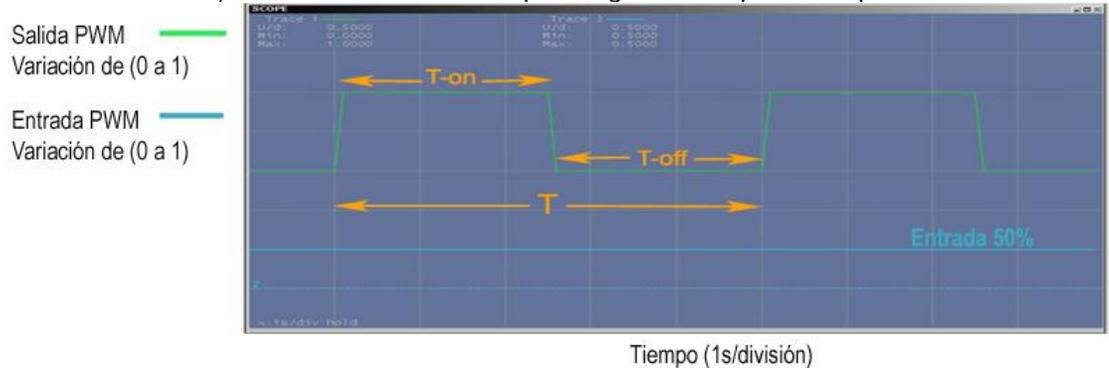
El bloque *PWM* permite cambiar el valor del periodo de PWM según las necesidades del usuario. El tiempo en ON de la señal de salida, se define mediante el valor de la entrada normalizado, donde 1 equivale al 100% de la señal en ON (toda la energía entregada) y 0 al 0% cuando la señal permanece en OFF (no se entrega energía). La ilustración 2.17 muestra el funcionamiento del bloque *PWM*.

En la Ilustración 2.17a, se puede observar que la señal de entrada es una rampa que varía entre 0 y 1 (azul) y la salida es un tren de pulsos (verde) de periodo constante pero de tiempo ON variable. En la Ilustración 2.17b se ve un periodo del tren de pulsos en detalle, se genera una señal con un periodo de PWM  $T = 5$  s, la señal de entrada se mantiene en 0.5 (equivalente al 50%), por lo que el tiempo en ON es  $T_{on}=2.5$  s y el tiempo en OFF es  $T_{off}=2.5$  s.

**Ilustración 2.8: Funcionamiento del PWM MIC 955**



a) Variación en el tren de pulsos generados por el bloque PWM



b) Modulación del ancho del tiempo de encendido del PWM  
Fuente: Propia

La resolución resultante del tren de pulsos de la señal de PWM se calcula a través de la ecuación 2.2:

$$Resolución = \frac{Periodo(T)}{Tiempo\_de\_muestreo(tm)} \tag{2.2}$$

Para el proceso térmico del módulo MIC955, el periodo de PWM del calentador es 5 s, y el tiempo de muestreo trabajado es de  $T_m=100$  ms, por lo que se cuenta con una resolución de 50 partes por periodo. Mientras que para el PWM del ventilador se fijo un periodo  $T = 400$  ms, con un tiempo de muestreo de 100 ms, que permite una resolución de 4 partes por periodo. Cuando el tiempo en ON del ventilador es de 25%

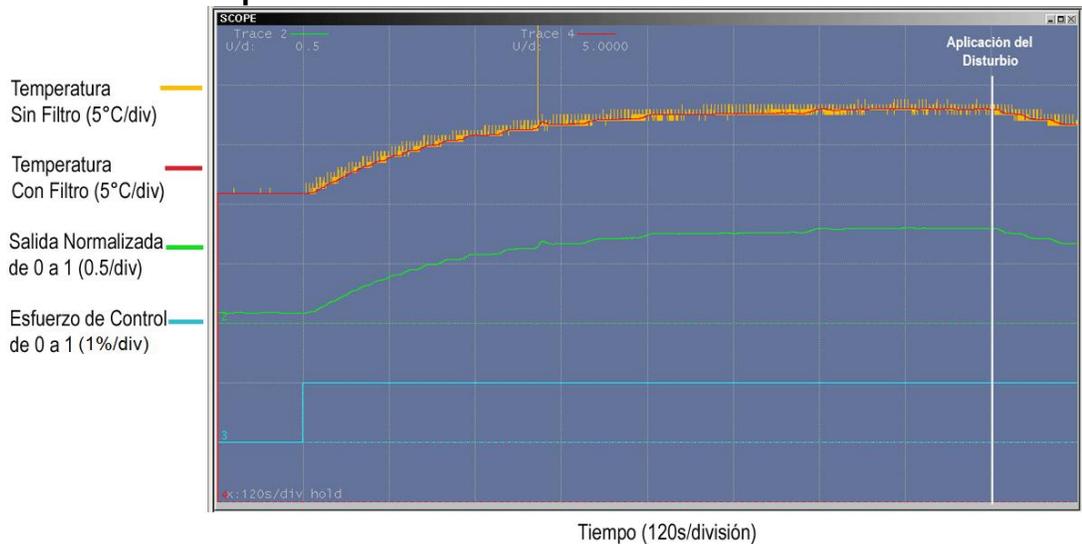
se tiene un disturbio bajo, pero cuando el tiempo en ON es del 100% se genera un disturbio crítico.

## 2.6 PRUEBAS DE IMPLEMENTACIÓN

Para probar el funcionamiento correcto del bloque *MIC955*, se creó un sistema de control en lazo abierto, en el cual a través de la aplicación de una entrada escalón se realizó un cambio del 1% en potencia de entrada. Las salidas resultantes fueron las señales de temperatura en grados centígrados, verificados directamente con un termómetro sobre la platina de aluminio negro del módulo *MIC955* y la salida de la señal normalizada, en donde se tiene en cuenta que los valores estuvieran de acuerdo a los parámetros *zero* y *span* configurados en el bloque *Transmisor*. Adicionalmente se generó un disturbio con el ventilador, mediante la activación de la señal desde el *Xrtailab*.

La Ilustración 2.18 muestra la respuesta de tres señales en el transmisor, ante una de las señales escalón aplicadas, equivalente a un cambio del 1% en el esfuerzo de control (0 a 0.01), aplicado al calefactor en el instante  $t=120$  segundos (señal azul). La señal naranja es la temperatura equivalente a la salida del conversor ADC, procesada sólo con el bloque de linealización y la señal roja es la variable temperatura linealizada y filtrada con un  $damping=4$  s, las dos señales están medidas en grados centígrados y las unidades de división por bloque son de  $5^{\circ}\text{C}$ . La línea verde equivale a la salida normalizada con los parámetros:  $zero=25^{\circ}\text{C}$  y  $span=10^{\circ}\text{C}$  y la división por bloque mostrada en la gráfica es de 0.5. Adicionalmente se aplica un disturbio generado por el ventilador y activado por el usuario en  $t = 1080$  segundos (señalado por la línea blanca).

**Ilustración 2.9: Respuesta de la Planta ante un escalón del 1% desde T-amb**



Fuente: Propia

La Ilustración 2.18 también muestra como se elimina un alto porcentaje del ruido producido por diferentes factores que alteran la medición, pero que no se manifiesta en la temperatura transmitida para el controlador o para la indicación. Sin embargo, es posible observar la perturbación generada por el disturbio del ventilador en la variable controlada.

## 2.7 RESUMEN

En este capítulo se explicó detalladamente la realización de la interfaz de comunicación, tanto hardware como software, que permite la conexión entre el módulo MIC955 y RTAI-Lab. En un principio se explicó detalladamente la estructura física y el funcionamiento del módulo MIC955, enfatizando el tipo de señales de entrada y salida generadas por el conversor ADC y las señales de entrada que se necesitan para la activación de los diferentes actuadores.

Luego se explicaron las distintas configuraciones del puerto paralelo, sus características físicas y los modos en los que puede ser utilizado. Una vez que se comprobó la compatibilidad entre el módulo MIC955 y el puerto paralelo, se explicó con detalle el mapeo de señales que serán transmitidas para el diseño y la implementación del cable de comunicación. Sin embargo, también se menciona que antes de conectar el MIC955 al PC, se procedió a hacer diferentes pruebas de configuración del puerto, para evaluar su desempeño en tareas de tiempo real.

Por último se comentó el diseño e implementación de los bloques *Mic955 Sensor* y *Mic955 actuador*, los cuales proveen los protocolos de comunicación entre la tarea de tiempo real y el módulo MIC955. También se explicó como son tratadas las señales de control y de salida del sensor para poderlas acoplar en el lazo de control.

### 3 CONTROLADORES PID

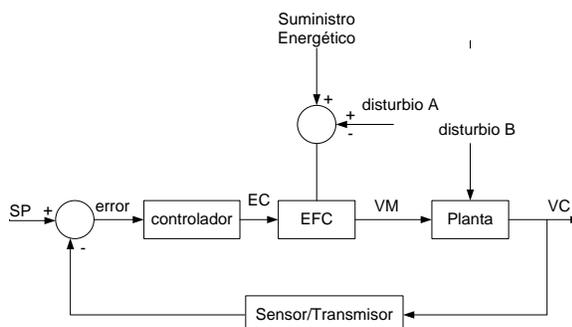
En este capítulo se presentan los conceptos, funcionamiento y estructuras de los controladores PID, al igual que su implementación en el Sistema de Prototipado Rápido de Control para el módulo MIC 955 de la empresa Feedback.

#### 3.1 CONCEPTOS BÁSICOS DE CONTROL REALIMENTADO

Los sistemas de control realimentado o en lazo cerrado se caracterizan por comparar una señal controlada con una señal de referencia y con base en la diferencia envían una señal de corrección a través del sistema con el fin de llevar la salida a un valor deseado (Kuo, 1996).

En la ilustración 3.1 se puede apreciar los diferentes elementos que componen un sistema de control realimentado en planta SISO.

#### Ilustración 3.1: Diagrama de bloques de un sistema de control realimentado



Fuente: Notas de clase, curso Instrumentación Industrial, Ingeniero Juan Fernando Flórez, Ingeniería en Automática Industrial, Universidad del Cauca, 2010

- El *set point* (SP) es una señal externa aplicada para indicar el valor deseado para la salida del proceso, también recibe el nombre de consigna o señal de referencia.
- La variable controlada (VC) es la señal que se obtiene a la salida de la planta, la cual se desea medir y controlar, en algunas ocasiones recibe el nombre de variable de proceso (VP).
- La señal de error (error) es la diferencia entre el *set point* y la variable controlada.

- El controlador es un elemento que basado en la señal de error y en una ecuación de control, genera una señal de corrección con el objetivo de eliminar o disminuir esta señal de error.
- La ecuación de control, también llamada ley de control o algoritmo de control, es la que genera finalmente la señal de corrección hacia el elemento final de control (EFC).
- El esfuerzo de control (EC) es la salida en términos de cantidad de energía, que el controlador le ordena al elemento final de control entregue a la planta.
- Se denomina elemento final de control (EFC) al instrumento utilizado para manipular el flujo de material o energía dentro del sistema.
- La variable manipulada (VM) es la cantidad o condición, normalmente un fluido energético, que el controlador ordena modificar, a través del elemento final de control, para afectar el valor de la variable controlada.
- La planta, donde se lleva a cabo un proceso, es el sistema bajo control.
- El sensor, en conjunto con el transmisor, es el instrumento utilizado para medir y transmitir la variable de proceso, como temperatura, presión, nivel, etc.
- Los disturbios son variables físicas que cuando cambian de valor y se propagan en el proceso perturban el valor de la variable controlada. Normalmente se presentan en la VM (Disturbio A) o en el fluido procesado (Disturbio B).

Existen dos enfoques de control diferentes en los sistemas de control realimentado: los sistemas diseñados para rechazar el efecto de los disturbios que desvían la variable controlada del *set point* reciben el nombre de control regulatorio; por otro lado, si el *set point* es una señal que está variando en el tiempo, constituye el disturbio más importante dentro del sistema de control, los sistemas diseñados para que la variable controlada siga estas variaciones del *set point* reciben el nombre de servo control (Smith, 2002).

## 3.2 ALGORITMOS DE CONTROL

La forma en la que un controlador responderá a la señal de error está determinada por el algoritmo de control implementado (control ON/OFF, control P, control PI o control PID). En esta sección se describen los algoritmos de control comunes en el control de procesos industriales.

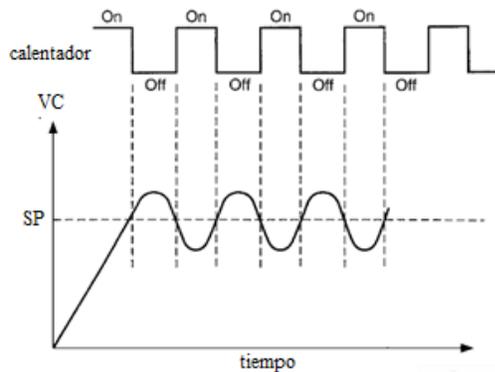
### 3.2.1 Controlador ON/OFF

En este controlador el esfuerzo de control  $U_{(t)}$  sólo puede tomar dos valores,  $U_{\max}$  y  $U_{\min}$ , dependiendo de la señal de error. La ley de control ON/OFF esta expresada en la ecuación 3.1:

$$U_{(t)} = \begin{cases} U_{\max} & \forall e_{(t)} > 0 \\ U_{\min} & \forall e_{(t)} < 0 \end{cases} \quad (3.1)$$

La ilustración 3.2 muestra el comportamiento de un controlador ON/OFF en un lazo de control de temperatura para la calefacción de una habitación mediante un calentador. Se observa como el estado del calentador conmuta entre las posiciones de encendido (on) y apagado (off) dependiendo del valor de la VC (temperatura en la habitación), mientras la VC sea menor que la temperatura deseada SP, el esfuerzo de control tendrá el máximo valor por lo tanto el calentador estará encendido, en el momento que la VC sobrepase el valor del SP, el esfuerzo de control tendrá el mínimo valor y el calentador se apagará.

### Ilustración 3.2: Respuesta de un controlador On-Off



Fuente: modificada de (McMillan and Cosidine, 1999)

La principal característica del control ON/OFF es que la variable de proceso siempre estará oscilando alrededor del SP. La tasa a la cual la VC oscila y la desviación de ésta respecto al SP dependen de las características dinámicas del sistema (McMillan and Cosidine, 1999).

En la práctica el controlador ON/OFF ideal no es utilizado, debido a que los disturbios del proceso y las interferencias eléctricas pueden causar que el esfuerzo de control esté oscilando rápidamente, lo cual se vería reflejado en la disminución de la vida útil de los elementos finales de control, como relés y válvulas. Una implementación común es adicionar una zona de histéresis para minimizar estas oscilaciones a la salida del controlador.

### 3.2.2 Controlador Proporcional

El esfuerzo de control que se implementa en los controladores proporcionales (P) se caracteriza por tener una relación constante entre la entrada y la salida del controlador. En este tipo de controladores existe un parámetro ajustable  $K_c$ , denominado ganancia proporcional. En algunas ocasiones este parámetro se expresa en términos de banda proporcional (BP). La relación entre la ganancia proporcional  $K_c$  y la banda proporcional, está dada por ecuación 3.2 (Lipták, 2006):

$$K_c = \frac{100}{BP} \quad (3.2)$$

Donde:

$K_c$  = Ganancia proporcional

$BP$  = Banda Proporcional

Cómo su nombre lo indica, la corrección generada por este tipo de controladores será proporcional a la señal de error. La ecuación 3.3 describe la operación del controlador P:

$$U_{(t)} = K_c * e_{(t)} + b = \frac{100}{BP} * e_{(t)} + b \quad (3.3)$$

Donde:

$U_{(t)}$  = esfuerzo de control

$e_{(t)}$  = señal de error, desviación de la VC con respecto al SP

$K_c$  = Ganancia proporcional

$BP$  = Banda Proporcional

$b$  = bias, salida del controlador cuando el error es cero

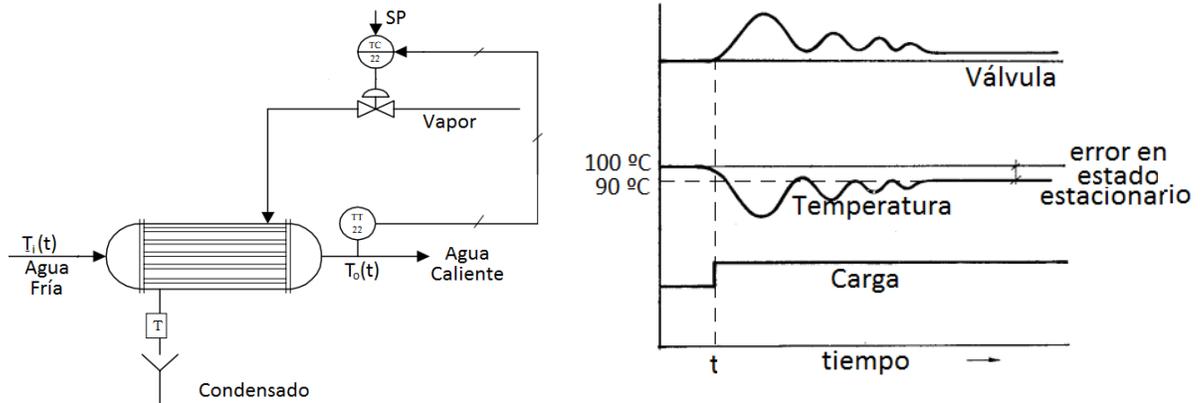
El término  $K_c$  establece la sensibilidad que tendrá el controlador a la señal de error, es decir cuánto cambia la salida del controlador por unidad de cambio en el error. El control proporcional disminuye el error, sin embargo no lo elimina completamente dejando una desviación llamada error en estado estacionario.

El error en estado estacionario también conocido como *offset*, es una característica indeseable del control proporcional. En la ilustración 3.3 se observa un sistema de control de temperatura para un intercambiador de calor y los regímenes de carga, temperatura y la posición de la válvula de control. Inicialmente el valor deseado (SP) es 100 °C. En el instante t, se presenta un cambio en la carga, originado por un aumento en el consumo de agua caliente. En la ilustración 3.3b se puede observar que la VC no se estabiliza en el valor deseado sino en 90 °C. La diferencia entre el SP y el nuevo valor de la VC se denomina error en estado estacionario.

Al aumentar la ganancia proporcional  $K_c$ , el error en estado estacionario disminuye, sin embargo ganancias más grandes hacen la respuesta más oscilatoria (Smith, 2002). Para la mayoría de procesos industriales existe un valor límite para la ganancia proporcional  $K_c$  antes de que el proceso se desestabilice, esta ganancia última  $K_{cu}$  es el valor máximo que puede ajustarse en el controlador P para mantener la estabilidad del sistema, por lo tanto el error en estado estacionario no puede ser eliminado

completamente con un controlador P (Smith and Corripio, 1997). Una observación importante en este tipo de controladores, es que si el valor de  $K_c$  genera oscilaciones sostenidas en la VC, el error es cero en un período de oscilación; este caso particular del controlador P es el mismo control ON/OFF.

**Ilustración 3.3: Control Proporcional de temperatura para un intercambiador de calor**



a) Lazo de control de temperatura para un intercambiador de calor

b) Regímenes de carga, temperatura y posición de la válvula

Fuente: modificado de (Creus, 1997)

**3.2.3 Controlador Proporcional Integral**

La ecuación 3.4 muestra el EC de un controlador proporcional integral (PI):

$$U_{(t)} = K_c \left[ e_{(t)} + \frac{1}{T_i} \int e_{(t)} \right] \tag{3.4}$$

Donde:

$U_{(t)}$  = esfuerzo de control

$e_{(t)}$  = señal de error, desviación de la VC con respecto al SP

$K_c$  = ganancia proporcional

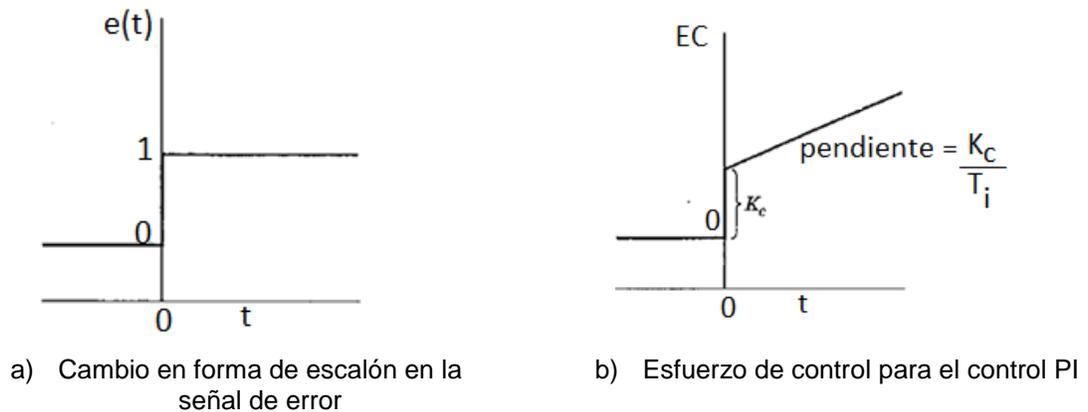
$T_i$  = tiempo integral o tiempo de reajuste, min/rep, rep/min

La principal ventaja de utilizar un controlador PI es la eliminación del error en estado estacionario: para que un proceso determinado se encuentre en condiciones de estado estable, el término I del esfuerzo de control  $U_{(t)}$  de la ecuación 3.4, debe ser constante, por lo tanto la variable manipulada también será constante. La ecuación 3.4 implica que el EC varía en el tiempo a menos que  $e_{(t)} = 0$ , de esta manera cuando

se usa un controlador PI, el EC cambia automáticamente hasta alcanzar el valor requerido para hacer el error en estado estacionario igual a cero. Esta situación es válida siempre y cuando la salida del controlador o el elemento final de control no se encuentren en condiciones de saturación (ver sección 3.5), cuando esto sucede es imposible llevar la VC otra vez al valor del SP (Seborg et al, 2004).

En la ilustración 3.4 se puede apreciar la respuesta de un controlador PI para un cambio en forma de escalón en la señal de error:

**Ilustración 3.4: Respuesta de un controlador PI**



Fuente: modificada (Seborg et al, 2004)

En el instante  $t=0$ , el EC cambia instantáneamente debido a la ganancia  $K_c$  del término proporcional P. La acción integral genera el incremento en forma de rampa en el EC. Cuando  $t=T_i$  el término integral ha contribuido una cantidad igual en magnitud que el término proporcional, es decir la acción integral ha repetido la acción proporcional.

Algunos controladores comerciales permiten la calibración del componente integral en unidades de  $\frac{1}{T_i}$  (repeticiones por minuto) en lugar de  $T_i$  (minutos o minutos por repetición).

**3.2.4 Controlador Proporcional Derivativo**

El propósito del controlador derivativo (D) es anticipar el comportamiento de la señal de error basándose en su tasa de cambio  $de/dt$ . La expresión matemática para un controlador D ideal se muestra en la ecuación 3.5 (Seborg et al, 2004).

$$U_{(t)} = T_d \frac{de_{(t)}}{dt} \tag{3.5}$$

Donde:

$U_{(t)}$  = esfuerzo de control

$e_{(t)}$  = señal de error, desviación de la VC con respecto al SP

$T_d$  = tiempo derivativo, [min]

En la ecuación 3.5 se observa que si el error es muy grande y cambia lentamente o en un caso extremo permanece constante, el EC de un controlador D será cero, por esta razón los controladores D nunca se usan solos, por lo general se usan en conjunto con controladores P o controladores PI (ver sección 3.2.5). La configuración mínima de un controlador PD está dada por la ecuación 3.6:

$$U_{(t)} = K_c \left[ e(t) + T_d \frac{de_{(t)}}{dt} \right] \quad (3.6)$$

Donde:

$U_{(t)}$  = esfuerzo de control

$e_{(t)}$  = señal de error, desviación de la VC con respecto al SP

$K_c$  = ganancia proporcional

$T_d$  = tiempo derivativo, [min]

La ilustración 3.5 muestra el comportamiento de un controlador PD a una señal de error que varía en el tiempo de la siguiente manera:

$$\begin{aligned} 0 < t \leq t_1 & \quad e_{(t)} = 0 \\ t_1 < t \leq t_2 & \quad e_{(t)} = \text{rampa con pendiente } m \\ t > t_2 & \quad e_{(t)} = \text{constante} \end{aligned}$$

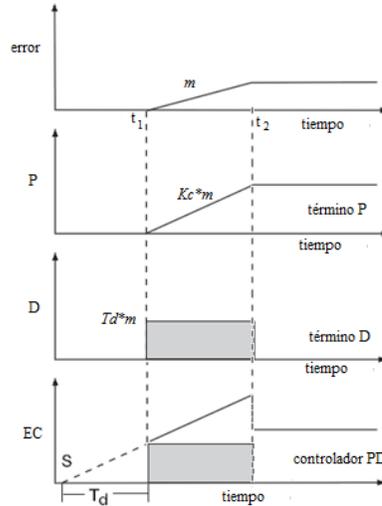
Cómo se puede observar en la ilustración 3.5, hasta el instante  $t_1$  la señal de error es cero, por consiguiente la corrección generada por los términos P y D es nula.

A partir del instante  $t_1$ , la señal de error presenta una forma de rampa cuya pendiente es igual a  $m$ , por lo tanto la corrección que genera el término P también es una señal en forma de rampa cuya pendiente será igual a  $K_c * m$ .

La corrección generada por el término D a partir del instante  $t_1$ , es una función constante que depende del valor de la tasa de cambio de la señal de error  $m$  y del valor del tiempo  $T_d$ . El parámetro  $T_d$  o tiempo derivativo, es el tiempo que le toma al

componente proporcional alcanzar el mismo valor de la componente derivativa, se mide en minutos o segundos.

**Ilustración 3.5. Respuesta de un controlador PD en lazo abierto**



Fuente: modificada de (Svrcek et al, 2006)

La ecuación 3.7 muestra la función de transferencia (FT) para el controlador D ideal representado en la ecuación 3.5.

$$\frac{U_{(s)}}{E_{(s)}} = T_d s \tag{3.7}$$

Cómo se puede observar en la ecuación 3.7, la FT de transferencia del controlador D es impropia (el orden del numerador es mayor que el del denominador) por lo tanto esta FT es físicamente irrealizable con elementos digitales o analógicos (Chen, 2006). En la práctica sin embargo, la acción de control D se aproxima a un filtro de primer orden denominado filtro derivativo (ver sección 3.3.1.3) que reduce la sensibilidad de la acción de control D a los componentes de alta frecuencia (ruido industrial) en la VC (Seborg et al, 2004).

**3.2.5 Controlador PID**

Un controlador tipo PID incluye elementos de los tres algoritmos de control vistos anteriormente. La ecuación 3.8 representa el algoritmo básico de un controlador tipo PID:

$$U_{(t)} = K_c \left[ e_{(t)} + \frac{1}{T_i} \int e_{(t)} + T_d \frac{de_{(t)}}{dt} \right] + b \tag{3.8}$$

Donde:

$U_{(t)}$  = esfuerzo de control

$e_{(t)}$  = señal de error, desviación de la VP con respecto al SP

$K_c$  = ganancia proporcional

$T_d$  = tiempo integral, [rep/min]

$T_d$  = tiempo derivativo, [min]

$b$  = bias, salida del controlador cuando el error es cero

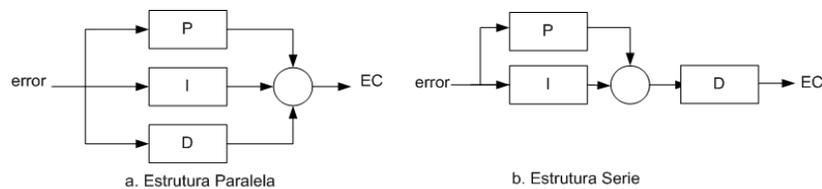
Debido a su relativa simplicidad y efectividad, los controladores tipo PID son utilizados en la gran mayoría de los lazos de control a nivel industrial; casi 95 % de estos lazos de control son de tipo PID (Li, et al, 2005).

Los controladores PID han evolucionado a la par con la tecnología y hoy en día son implementados en formas digitales más que con componentes neumáticos o eléctricos. Se pueden encontrar como equipos *stand – alone*, en sistemas embebidos o como bloques funcionales de un Controlador Lógico Programable (PLC) y en Sistemas de Control Distribuidos (DCS) (Visioli, 2006).

### 3.3 IMPLEMENTACIONES DEL CONTROLADOR PID

Aunque el controlador PID que representa la ecuación 3.8 a menudo se presenta como la estructura estándar, muchos fabricantes ofrecen una variedad de versiones modificadas. Esta variada gama de opciones se convierte en un inconveniente al intentar sintonizar un controlador PID, ya que la falta de estandarización en las estructuras de los algoritmos de control, implementados por parte de los fabricantes, ha provocado una confusión en la forma en la cual las acciones de control están relacionadas; inclusive en algunas ocasiones, se suele utilizar los mismos nombres para estructuras diferentes (Alfaro, 2002). La literatura técnica ofrece muchas formas de referirse al controlador PID, sin embargo las implementaciones más utilizadas a nivel industrial se derivan de alguna de las estructuras estándares: la estructura paralela y la estructura serie (Li et al, 2005), que se muestran en la ilustración 3.6.

**Ilustración 3.6: Diagrama de bloques de las estructuras serie y paralelo en un controlador PID**



Fuente: modificado de (Jhonson and Moradi, 2005)

### 3.3.1 Controladores PID de estructura Paralela

Los controladores PID de estructura paralela son los más utilizados en los libros de texto, sin embargo existen diferentes maneras de implementar este tipo de controladores entre las que encontramos el PID Ideal, el PID Paralelo y el PID Paralelo Industrial.

#### 3.3.1.1 Controlador PID Ideal

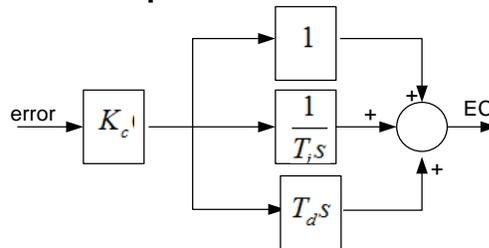
El controlador PID ideal es la forma más común empleada en la teoría para representar los controladores tipo PID, también es conocido como controlador PID no interactivo, debido a que el componente integral no tiene influencia sobre el componente derivativo y viceversa, es decir estos componentes no interactúan entre sí en el dominio del tiempo, aunque la ganancia proporcional si influye en los tres modos. Este algoritmo admite ceros complejos lo cual resulta útil cuando se controla sistemas con polos oscilatorios (Aström, 2006).

La función de transferencia de un controlador PID ideal se muestra en la ecuación 3.9 y su representación mediante diagrama de bloques en la ilustración 3.7.

$$G_{PIDI}(s) = K_c \left( 1 + \frac{1}{T_i s} + T_d s \right) \quad (3.9)$$

Otro nombre con el cual se conoce al controlador PID ideal es controlador PID ISA (Smith and Corripio, 1997).

#### Ilustración 3.7: Diagrama de bloques controlador PID ideal



Modificado de (Visioli, 2006)

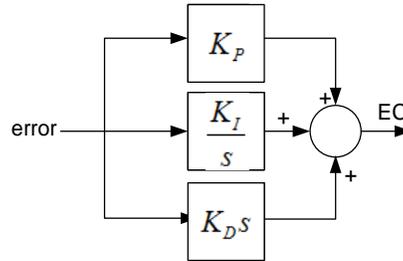
#### 3.3.1.2 Controlador PID Paralelo

La función de transferencia de la ecuación 3.10 corresponde a la forma paralela de un controlador PID, en este caso las acciones de control están completamente separadas tal como se evidencia en la ilustración 3.8.

$$G_{PIDP}(s) = K_p + \frac{K_I}{s} + K_D s \quad (3.10)$$

Estos controladores corresponden a la forma más general de implementación no interactiva, ya que a diferencia del controlador PID ideal, en el cual la ganancia P influye en las acciones de control integral y derivativa, en el controlador PID paralelo es posible obtener acciones de control por separado al establecer un valor nulo en sus parámetros  $K_p$ ,  $K_I$  y  $K_D$  (Visioli, 2006)

### Ilustración 3.8: Controlador PID Paralelo



Modificado de: (Jhonson and Moradi, 2005)

Si se conocen los parámetros de un controlador PID ideal es posible obtener los parámetros del controlador PID paralelo mediante las ecuaciones 3.11:

$$K_p = K_c, \quad K_I = \frac{K_c}{T_i}, \quad K_D = K_c K_d \quad (3.11)$$

#### 3.3.1.3 Controlador PID Paralelo Industrial

A continuación se presentan tres modificaciones que usualmente se implementan en los controladores PID tanto de estructura paralela como serie, para que sean funcionales en ambientes industriales.

#### Problemas relacionados con el componente derivativo

En un ambiente industrial la VC puede contener demasiado ruido eléctrico, tal ruido es modelado como un fenómeno de alta frecuencia y será amplificado por el componente derivativo ideal del controlador PID. Para prevenir esta amplificación indeseable, se aproxima el componente D mediante el filtro pasa bajos representado en la ecuación 3.12:

$$T_d s \cong \frac{T_d s}{\frac{T_d}{N} s + 1} \quad (3.12)$$

Donde  $N$  generalmente toma valores entre  $1 \leq N \leq 33$  aunque en la mayoría de las aplicaciones prácticas caen en un rango de  $8 \leq N \leq 16$  (Li et al, 2005).

Otro problema importante relacionado con el componente derivativo tiene que ver con los cambios repentinos en forma de escalón que se hacen en el SP; cuando estos

cambios ocurren, la acción de control D se hace grande y como resultado se genera un pico indeseable en el EC. Una solución es aplicar la acción de control D a la VC en lugar de aplicarlo directamente al error tal como se muestra en la ecuación 3.13 (Lipták, 2006):

$$U_{(t)} = K_c \left[ E_{(s)} + \frac{1}{T_I s} E_{(s)} - \frac{T_d s}{\frac{T_d}{N} s + 1} Y_{(s)} \right] \quad (3.13)$$

Otra solución consiste en aplicar el SP de forma suave (rampa) desde su valor actual hasta su valor final con una tasa de cambio específica, esto genera un cambio gradual en la VC y reduce considerablemente el sobreimpulso (Wade, 2004)

**Ponderación del set point:** Un problema frecuente en el diseño de un controlador PID es lograr al mismo tiempo un alto desempeño cuando se lo use como servomecanismo y como regulador, generalmente un controlador sintonizado para un óptimo seguimiento de los cambios del SP es demasiado conservador en el rechazo a los disturbios. Por otro lado, los parámetros del controlador que ofrecen un óptimo rechazo a los disturbios tienden a causar grandes sobreimpulsos en la VC, cuando se presentan cambios en el SP. Este problema puede ser tratado haciendo una ponderación de la señal de error en el componente proporcional, como lo muestra la ecuación 3.14 (Lipták, 2006):

$$U_{(s)} = K_p \left( bR_{(s)} - Y_{(s)} + \frac{R_{(s)} - Y_{(s)}}{T_i s} \right) \quad (3.14)$$

Donde  $b$  toma valores entre 0 y 1.

Cuando un controlador tiene una ponderación en la señal de error, recibe el nombre de controlador de dos grados de libertad (Seborg, 2004).

Un controlador PID paralelo industrial presenta todas las modificaciones anteriormente expuestas. La ecuación 3.15 muestra el EC para un controlador PID paralelo industrial:

$$U_{(s)} = K_c \left( (bR_{(s)} - Y_{(s)}) + \frac{1}{T_I} E_{(s)} - \frac{T_d s}{\frac{T_d}{N} s + 1} Y_{(s)} \right) \quad (3.15)$$

### 3.3.2 Controladores PID de estructura Serie

Los controladores PID de estructura serie fueron extensamente empleados en los antiguos reguladores analógicos, dado que suponían un ahorro económico al permitir

la implementación de control PID analógico con dos amplificadores, a diferencia de los controladores PID de estructura paralela que requieren de tres amplificadores.

Actualmente y a pesar que ya no existen los inconvenientes en la implementación digital de los controladores PID serie, algunos fabricantes siguen ofreciéndolo, de esta manera se cubre la demanda de quienes desean mantener la validez de las técnicas de ajuste habituales en controladores analógicos y sacar el máximo provecho a la experiencia (*know how*) de los operadores de planta (Visioli, 2006).

### 3.3.2.1 Controlador PID Serie

El controlador PID serie también recibe el nombre de interactivo, debido a que una modificación en el componente derivativo afecta al componente integral y viceversa, esa interacción es notoria en la ecuación 3.16:

$$U_{PIDS(s)} = K'_c \left( 1 + \frac{1}{T'_i s} \right) (T'_d s + 1) \quad (3.16)$$

Cabe resaltar que un controlador PID serie siempre puede ser representado en forma de controlador PID ideal, aplicando las expresiones de la ecuación 3.17:

$$K_p = K'_c \frac{T'_i s + T'_d s}{T'_i s}, \quad T_i = T'_i s + T'_d s, \quad T_d = \frac{T'_i s T'_d s}{T'_i s + T'_d s} \quad (3.17)$$

Por el contrario no siempre es posible hacer la conversión de controlador PID ideal en controlador PID serie, solamente se puede hacer si se cumple la restricción dada en la ecuación 3.18:

$$T_i > 4T_d \quad (3.18)$$

A través de las expresiones de la ecuación 3.19:

$$K'_c = K_c \left( 1 + \sqrt{1 - 4 \frac{T_d}{T_i}} \right), \quad T'_i = \frac{T_i}{2} \left( 1 + \sqrt{1 - 4 \frac{T_d}{T_i}} \right), \quad T'_d = \frac{T_i}{2} \left( 1 - \sqrt{1 - 4 \frac{T_d}{T_i}} \right) \quad (3.19)$$

Un nombre común que se le da a esta forma de representación es controlador PID clásico, este nombre se debe a que los controladores antiguos eran implementados con dispositivos neumáticos basados en sistemas de balance de fuerzas. El controlador serie tiene una interesante interpretación en el dominio de la frecuencia: los ceros corresponden a los valores inversos de los tiempos derivativo e integral. Todos los ceros del controlador son reales y las acciones integral y proporcional puras no pueden ser obtenidas mediante valores finitos de los parámetros del controlador (Aström, 2006).

### 3.3.2.2 Controlador PID Serie Industrial

Al igual que en los controladores PID de estructura paralela, se hace algunas modificaciones relacionadas con el término derivativo en el algoritmo básico de los controladores PID serie, para que sea funcional en ambientes industriales. El esfuerzo de control para un controlador PID serie industrial se muestra en la ecuación 3.20 (O'Dwyer, 2009)

$$U_{(s)} = K_c \left( 1 + \frac{1}{T_i s} \right) \left( R_{(s)} - \frac{T_d s + 1}{\frac{T_d}{N} s + 1} Y_{(s)} \right) \quad (3.20)$$

## 3.4 MODIFICACIONES ADICIONALES DE UN CONTROLADOR PID

Aunque los controladores PID vistos en los apartados anteriores son funcionales, aún es necesario tener en cuenta algunos fenómenos no lineales que hacen que su desempeño no sea el más óptimo.

### 3.4.1 Efecto *Windup* del Integrador

En aplicaciones industriales los sistemas de control deben operar con restricciones que los sistemas físicos imponen, por ejemplo la capacidad finita de almacenamiento de compresores y tanques o el rango limitado de velocidad que tienen los motores. Un problema común tiene que ver con la saturación de los actuadores, por ejemplo una válvula de control solamente puede operar entre el rango de completamente abierta y completamente cerrada. Estas restricciones pueden llevar a situaciones poco deseables, tales como el *windup* en el componente integral de los controladores.

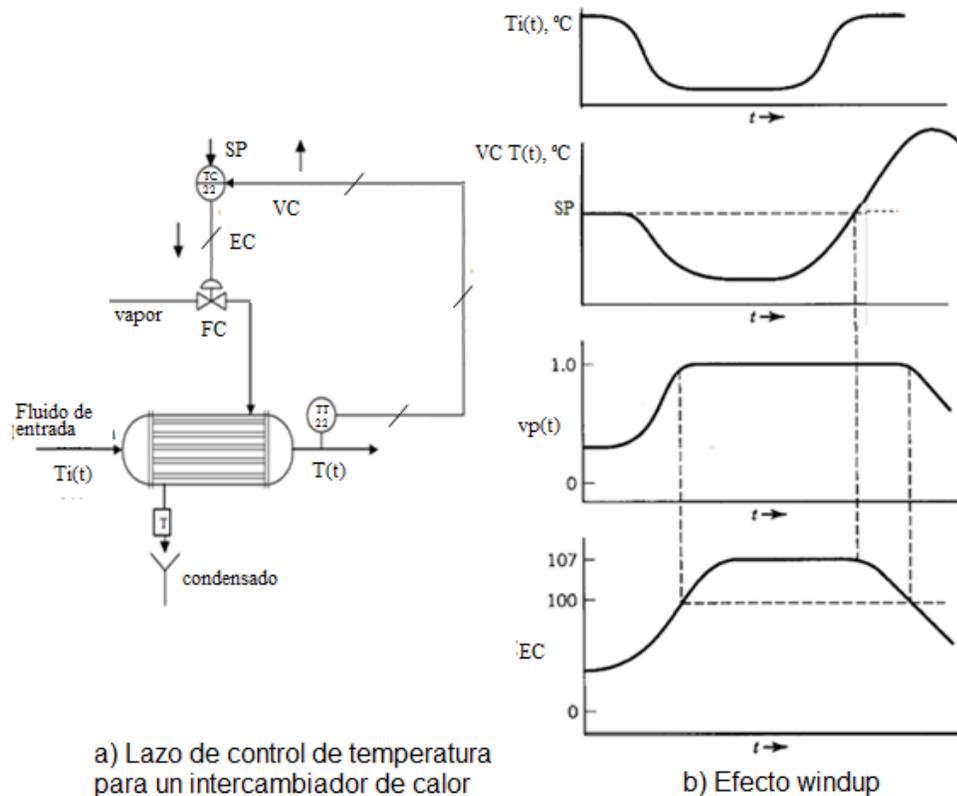
La ilustración 3.10 muestra un lazo de control de temperatura para un intercambiador de calor y es útil para explicar el problema del *windup* en el integrador.

Si la temperatura del fluido de entrada al proceso,  $T_i(t)$ , disminuye en una cantidad inusualmente grande, la VC  $T(t)$  también va a disminuir. El controlador le exigirá a la válvula de control, mediante el EC, entregar más vapor al intercambiador de calor; la válvula de control llega a su valor límite, sin embargo la VC se estabiliza muy por debajo del SP. Debido a que el error persiste, el EC crece un poco más allá de su valor de saturación, hasta que finalmente la salida del controlador queda totalmente saturada. Durante todo este tiempo la componente integral del controlador continúa incrementándose acumulando bastante área bajo la curva.

Después de un tiempo el disturbio desaparece y por tanto la VC empieza a subir; en el momento en que la VC alcanza y sobrepasa el SP, la válvula permanece completamente abierta, cuando debería comenzar a cerrarse. La razón por la cual

esto sucede es porque el EC se encuentra en saturación. Como puede observarse en la ilustración 3.9, durante el tiempo que la válvula permanece en la posición de completamente abierta, la VC se desvía del SP generando un sobreimpulso considerablemente grande o como se dice coloquialmente en inglés, “*It’s windup*”.

**Ilustración 3.9. Efecto *windup* en un lazo de control de temperatura para un intercambiador de calor.**



Fuente modificado (Smith and Corripio, 1997)

### 3.4.2 *Bump Transfer* o Salto de transferencia entre modos de control

Un controlador PID comercial puede operar de dos maneras diferentes:

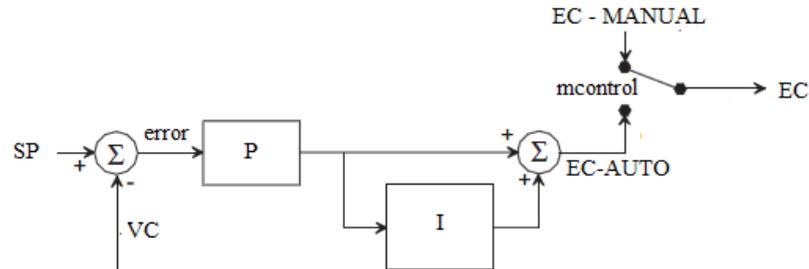
**Modo manual:** el operador controla directamente la operación sobre el proceso y puede incrementar o disminuir el EC en forma manual.

**Modo automático:** el controlador PID opera sobre el proceso.

La ilustración 3.10 muestra el diagrama de bloques de un sistema de control PI cuando es operado en modo manual. El operador selecciona el modo de control manual mediante el *switch* *mcontrol*. Cuando se han hecho los ajustes necesarios en el EC – MANUAL, puede hacerse la transferencia de modo manual a modo

automático. En ese momento el EC – MANUAL permanecerá constante en el valor que fue establecido en modo manual.

### Ilustración 3.10: Diagrama de bloques controlador PI operado en modo manual



Fuente: modificado de (Wade, 2004)

Al cambiar de modo de operación desde automático a manual para realizar ajustes y volver al modo automático, se puede presentar un cambio repentino en el EC debido a la diferencia de valores entre el EC – MANUAL y el EC – AUTO al momento de la transición, debido a que el término I del controlador basa su operación en la señal de error a la entrada del controlador y no toma en cuenta ninguna señal introducida manualmente, por lo tanto el EC - AUTO no será el mismo que el EC - MANUAL.

Esta situación hace que se genere un salto (*Bump*) en la salida EC del controlador, que a su vez se manifiesta en un salto en la VC de la planta (*bump transfer*).

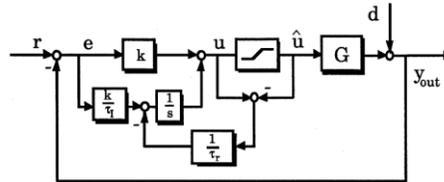
### 3.4.3 Técnicas Antiwindup Bumpless Transfer

En (Kothare, et al, 1993) se describen algunas técnicas para eliminar los efectos indeseables que puedan presentarse en la respuesta de un proceso, como es el caso del *windup* del integrador o el *bump transfer* en la conmutación de modos de control. A continuación se muestran las técnicas más relevantes usadas para implementar estrategias *antiwindup*. Cabe resaltar que usualmente técnicas *antiwindup* también son usadas como técnicas *bumpless transfer*, razón por la cual usualmente se refiere a ellas como estrategias *AntiWindup-BumplessTransfer (AWBT)*.

#### 3.4.3.1 Antireset windup – Seguimiento del modo integral

En esta técnica, la solución para evitar el *windup* es medir la salida del actuador  $\hat{u}$  y formar una señal de error que sería la diferencia entre la salida  $u$  del controlador y la salida del actuador  $\hat{u}$ . Esta señal de error se realimenta a la entrada del controlador a través de una ganancia  $\frac{1}{T_r}$ , como se observa en la ilustración 3.11:

### Ilustración 3.11: Antireset windup



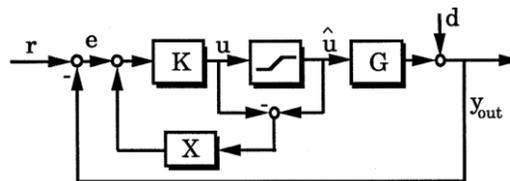
Fuente: (Kothare et al, 1993)

Cuando el actuador se satura, la señal de realimentación intenta conducir el error  $u - \hat{u}$  a cero, re-calculando la acción integral, de tal forma que la salida del controlador sea exactamente igual que el límite de saturación.

### 3.4.3.2 Conventional Anti-windup (CAW)

Esta técnica adopta una filosofía similar a la del *antireset windup*. En algún sentido la técnica CAW puede ser considerada como una extensión directa de la técnica *antireset windup* para controladores generales. La ilustración 3.12 muestra la implementación de la técnica CAW, como se puede observar la realimentación AWBT se hace a través de una matriz  $X$  de alta ganancia hacia la entrada del controlador  $e$ . Típicamente  $X = \alpha I$  donde  $\alpha \gg 1$  y es un número escalar.

### Ilustración 3.12: Estrategia Conventional Anti-windup



Fuente: (Kothare, et al, 1993)

### 3.4.3.3 Controlador condicionado Hanus

En esta técnica, el efecto *windup* se interpreta como una falta de consistencia entre los estados internos del controlador y la entrada a la planta; para restaurar a consistencia se modifica las entradas al controlador de tal manera que si las entradas modificadas también llamadas "referencias realizables" han sido aplicadas al controlador, su salida no tendría que ser diferente a la entrada de control a la planta.

### 3.4.3.4 Antiwindup basado en observador

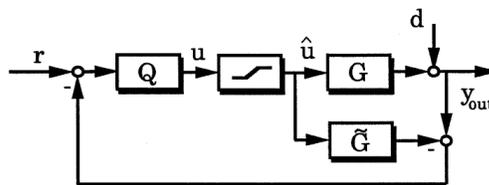
Una interpretación del problema del *windup* es que los estados del controlador no corresponden a la señal del controlador que está siendo realimentada a la planta, esta imprecisión en el vector de estados del controlador se debe a la falta de estimaciones

correctas de los estados del controlador en presencia de no linealidades del actuador. Para obtener estimaciones correctas de los estados y evitar el *windup*, Aström propone introducir un observador en el controlador. En lugar de tener un controlador y un observador por separado, ambos son integrados en un esquema en forma de compensador AWBT, por lo tanto el observador viene dentro de la estructura del controlador y actúa solamente en presencia de presencia de no linealidades del actuador y por lo tanto no afecta al controlador lineal.

### 3.4.3.5 Control por Modelo Interno (IMC)

Aunque la estructura de control por modelo interno (IMC) nunca fue propuesta para ser un esquema *antiwindup*, tiene una potencial aplicación para la solución de este problema, en el caso donde el sistema sea estable en lazo abierto. Las aplicaciones AWBT del IMC han sido estudiadas en (Cohen et al,1985) y (Debelle, 1979), la ilustración 3.13 se muestra una estructura IMC con una no linealidad del actuador.

#### Ilustración 3.13: Estructura de Control por Modelo Interno



Fuente: (Kothare, et al, 1993)

Si el controlador se implementa en la configuración IMC, las restricciones del actuador no causan ningún problema de estabilidad y se envía la señal de control restringida  $\hat{u}$  tanto a la planta como al modelo. Suponiendo que hay diferencias entre la planta y el modelo  $(G \equiv \tilde{G})$ , es fácil ver que la estructura IMC permanece efectivamente en lazo abierto y la estabilidad es garantizada por la estabilidad de la planta  $(G)$  y el controlador IMC  $(Q)$ . La estructura IMC ofrece la oportunidad de implementar complejos algoritmos de control, posiblemente no lineales, sin generar problemas de estabilidad donde no haya diferencia entre el modelo y la planta.

### 3.4.3.6 Diseño AntiWindup para control por modelo interno (IMC)

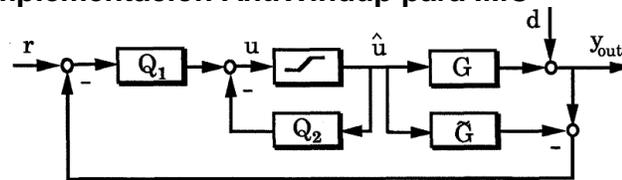
La ventaja más importante del control por modelo interno (IMC) es que no presenta problemas de estabilidad debidos a las restricciones del actuador, desafortunadamente el costo de dicha estabilidad se traduce en un desempeño lento. Esto se debe a que la salida del controlador es independiente de la salida de la planta tanto en el régimen lineal como en el régimen no lineal. Mientras esto no tenga

importancia en el régimen lineal, la implicación en el régimen no lineal será que el controlador “pasa desapercibido” del efecto que esta acción produce a la salida, lo cual resulta en algunas lentitudes.

Este efecto es más visible cuando el control IMC tiene dinámicas rápidas las cuales son “desbaratadas” por la saturación. Por otro lado, si el IMC es diseñado para optimizar el desempeño no lineal, no habrá un rendimiento satisfactorio para la saturación del sistema.

Una estructura modificada para el control IMC para optimizar el desempeño frente a la saturación del sistema se muestra en la ilustración 3.14, donde Q1 y Q2 pueden considerarse funciones de transferencia estables (Zheng et al, 1994).

**Ilustración 3.14: Implementación AntiWindup para IMC**



Fuente: (Kothare, et al, 1993)

### 3.5 CONTROLADORES PID EN Scilab/Scicos

Para implementar en el sistema de Prototipado Rápido de Control (RPC) para el módulo MIC955 de la empresa Feedback, se seleccionaron dos controladores PID de los propuestos en (O’Dwyer, 2009). En la selección se tuvo en cuenta algunas restricciones como la implementación de la acción de control D, así como otras restricciones derivadas del modelado del proceso, entre las cuales están el método de identificación utilizado y la relación tiempo muerto/constante de tiempo del modelo.

#### 3.5.1 Controlador PID con Filtro Derivativo

La ecuación 3.9 representa a un controlador PID ideal, generalmente esta implementación se utiliza para explicar cómo funcionan los controladores PID, sin embargo cómo se explicó anteriormente la acción de control derivativa no es realizable con elementos reales.

En la práctica, para implementar la acción de control derivativa se utiliza un filtro pasa bajos que incluye un polo real con una constante de tiempo igual a  $\frac{T_d}{N}$  (ver ecuación 3.12), este filtro limita las componentes de alta frecuencia presentes en la variable controlada (ver sección 3.3.1.3.).

La función de transferencia para un controlador PID con Filtro Derivativo se muestra en la ecuación 3.21:

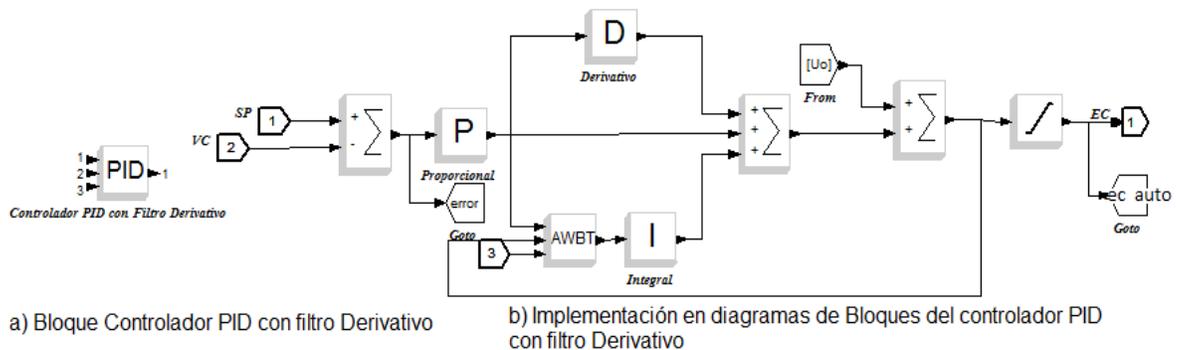
$$G_c(s) = K_c \left( 1 + \frac{1}{T_i s} + \frac{T_d s}{\frac{T_d}{N} s + 1} \right) \quad (3.21)$$

Según (O'Dwyer, 2009), el controlador PID con Filtro Derivativo está presente en los siguientes productos comerciales:

1. Bailey Net 90 PID, derivada sobre el error,  $N = 10$  (McMillan, 1994) y Bailey FC156 (Algoritmo PID independiente) (EZYtune, 2003)
2. Modicon Concept PIDP1 y PID1 (EZYtune, 2003)
3. Fischer & Porter DCU 3200 CON PID,  $N = 8$  (EZYtune, 2003)
4. Foxboro EXACT I/A series PIDA (Foxboro, 1994)
5. PID Hartmann & Braun Freelance 2000 (EZYtune, 2003)
6. Modicon 984,  $2 \leq N \leq 30$  (McMillan, 1994; EZYtune, 2003)
7. Siemens Teleperm/PSC7 ContC/PCS7 CTRL PID,  $N = 10$  (ISMC, 1999) y PLC S7 FB41 (EZYtune, 2003).

La ilustración 3.15 muestra el diagrama de bloques del controlador PID con filtro derivativo implementado en la herramienta Scicos, se observa claramente la forma en la cual las acciones de control están relacionadas:

**Ilustración 3.15: Implementación en diagrama de bloques del Controlador PID con Filtro Derivativo.**

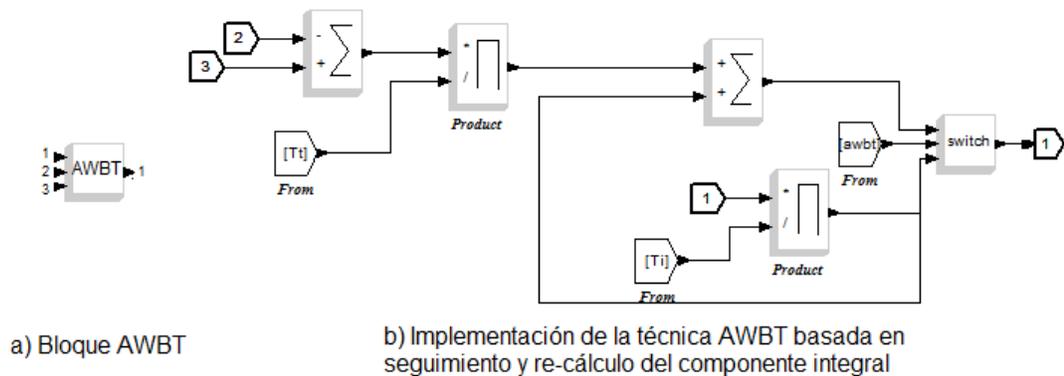


Fuente: propia

En la ilustración 3.15 se pueden observar las entradas 1 y 2 que corresponden al SP y a la VC, respectivamente. La entrada 3 corresponde a la señal de seguimiento del modo integral para el bloque AWBT. Para una mayor comprensión del diagrama de bloques del controlador, las acciones de control se separaron mediante el uso de súper-bloques. Dentro de cada súper-bloque existen conmutadores que permiten convertir el controlador PID con filtro derivativo en otro tipo de controladores como un controlador P o controlador PI. También se puede observar que el controlador trae implementado la adición del valor base ( $U_o$ ) y la técnica *Antiwindup-Bumpless Transfer* (súper bloque AWBT).

La ilustración 3.16 muestra el bloque AWBT y la implementación mediante diagramas de bloques de la técnica AWBT basada en re-cálculo y seguimiento del modo integral (aström, 2006). El bloque AWBT tiene tres entradas y una salida, la entrada 1 corresponde a la señal de error (SP-VC), las entrada 2 y la entrada 3 corresponden a la realimentación extra que permite el re-cálculo y seguimiento del modo de control integral. Dependiendo de la señal de habilitación (awbt) que ingrese al conmutador (switch), la salida 1 corresponderá a la señal de error dividida por el tiempo integral (Ti) cuando la protección AWBT este desactivada (awbt=0) o a la señal de seguimiento del modo integral cuando la protección AWBT este activa (awbt=1).

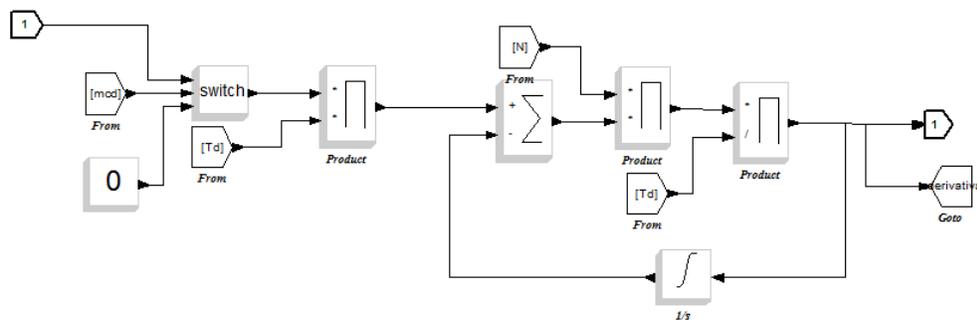
**Ilustración 3.16: Implementación mediante diagrama de bloques de la técnica AWBT basada en re-cálculo y seguimiento del modo integral**



Fuente: Propia

En la ilustración 3.17 se puede apreciar la implementación del filtro D para este tipo de controladores. En el anexo E se detalla la implementación del filtro derivativo.

**Ilustración 3.17: Implementación componente D para el controlador PID con filtro derivativo.**



Fuente: Propia

### 3.5.2 Controlador PID Clásico

La función de transferencia de un controlador PID clásico se muestra en la ecuación 3.22:

$$G_c(s) = K_c \left( 1 + \frac{1}{T_i s} \right) \left( \frac{T_d s + 1}{\frac{T_d s}{N} + 1} \right) \quad (3.22)$$

En la literatura técnica existen muchos nombres con los cuales se denomina a este controlador, entre los más comunes encontramos:

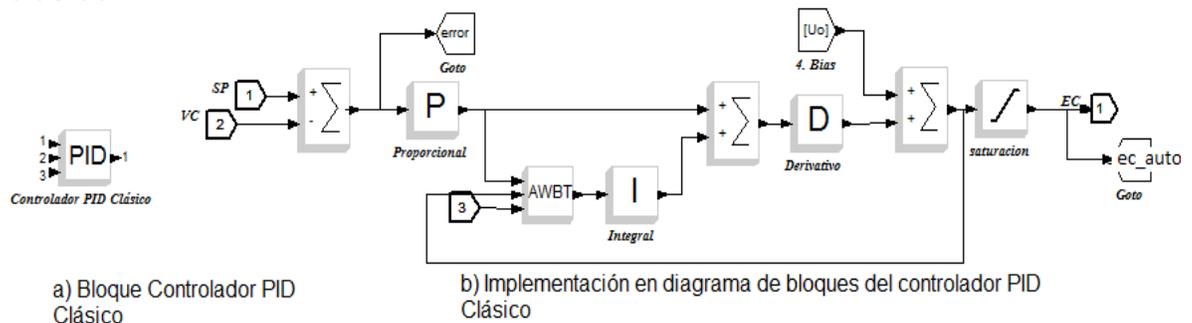
- Controlador “en cascada” (Witt and Waggoner, 1990).
- Controlador “interactuante” o controlador “serie” (Poulin and Pomerleau, 1996).
- Controlador “interactivo” (Tsang and Rad, 1995).
- Controlador “rate-before-reset” (Smith and Corripio, 1997).
- Controlador “análogo” (St. Clair, 2000)
- Controlador “comercial” (Luyben, 2001).

Según (O’Dwyer, 2009), este controlador está presente en los siguientes productos:

- Honeywell TDC *Basic/Extended/Multifunction* Tipo A and B,  $N = 8$  (McMillan, 1994)
- Toshiba TOSDIC 200,  $3.33 \leq N \leq 10$  (McMillan, 1994)
- Foxboro EXACT Modelo 761,  $N = 10$  (McMillan, 1994)
- Honeywell UDC6000,  $N = 8$  (Åström and Hägglund, 1995)
- Honeywell TDC3000, Tipo A, modo interactivo,  $N = 10$  (ISMC, 1999)
- Honeywell TDC3000 Universal, *Multifunction & Advanced Multifunction*,  $N = 8$  (ISMC, 1999)
- Foxboro EXACT I/A Series PIDA (Foxboro, 1994).

En la ilustración 3.18, se muestra el diagrama de bloques para un controlador PID clásico, en ella se observa claramente la dependencia de las acciones de control representados por los bloques P; I, D.

**Ilustración 3.18: Implementación en diagrama de bloques del controlador PID clásico.**



Fuente: Propia



## 4 SINTONIZACIÓN DE CONTROLADORES PID

El poder de los controladores PID radica en que pueden ser ajustados para obtener el comportamiento deseado de una amplia variedad de procesos a través de la elección apropiada de sus parámetros. La determinación de los valores aceptables de esos parámetros se conoce como sintonización del controlador. En este capítulo se presenta información y conceptos teóricos sobre el proceso de sintonización de controladores tipo PID.

### 4.1 ESPECIFICACIONES DE DISEÑO

Las especificaciones de diseño se usan para describir qué debe hacer el sistema de control y cómo hacerlo. Típicamente las especificaciones de diseño incluyen:

1. El sistema en lazo cerrado debe ser estable.
2. Proveer un excelente rechazo a disturbios: los efectos de las perturbaciones deben ser minimizados.
3. Proveer un excelente seguimiento de los cambios del SP: respuestas rápidas y suaves ante estos cambios.
4. Eliminar el error en estado estacionario.
5. Evitar demasiada variación en el esfuerzo de control.
6. El sistema de control debe ser robusto, es decir, insensible a cambios en las condiciones del proceso y a inexactitudes en el modelo del proceso.

Es prácticamente imposible cumplir con todas estas especificaciones simultáneamente, debido a que existen ciertos conflictos entre estas. El primer conflicto se presenta por que se deben equilibrar dos objetivos importantes: desempeño y robustez. Un sistema de control con un alto grado de desempeño, provee una rápida respuesta ante disturbios y ante cambios en el SP, con oscilaciones pequeñas o ausencia total de estas. Por otro lado cuando un sistema de control es robusto, provee un desempeño satisfactorio para un alto rango de condiciones de operación y para un grado razonable de inexactitud en el modelado del proceso. La robustez se consigue estableciendo parámetros conservadores en el controlador PID, es decir valores pequeños de  $K_c$  y grandes valores de  $T_i$ , sin embargo elegir estos parámetros resulta en un desempeño muy pobre, por lo tanto se estaría sacrificando el desempeño para lograr robustez (Seborg, 2004).

El segundo tipo de conflicto ocurre porque los parámetros del controlador PID que proveen un excelente rechazo a disturbios, pueden producir grandes sobrepasos ante cambios del SP. Por otro lado, si se eligen parámetros que proveen un excelente seguimiento de los cambios en el SP, la respuesta ante disturbios puede ser muy lenta. Como se observó en la sección 3.3.1.3 este conflicto puede ser evitado utilizando controladores con dos grados de libertad.

Las especificaciones de diseño se pueden expresar de distintas maneras, una posibilidad es utilizar las características de la respuesta temporal del proceso a entradas conocidas. Otra opción es utilizar las características de la respuesta en frecuencia (Astrom, 1995). Por ejemplo, las especificaciones como el sobrepaso máximo, el tiempo de levantamiento y el tiempo de establecimiento, están definidas para una entrada escalón unitario, por lo tanto son especificaciones de diseño en el dominio del tiempo. Entre las especificaciones típicas del dominio de la frecuencia están el margen de ganancia, el margen de fase y el pico de resonancia máximo; por lo general estas especificaciones se emplean junto a otras herramientas como la traza de Bode, la traza polar, la traza de ganancia-fase y la carta de Nichols (Kuo, 1996).

## **4.2 CRITERIOS DE DESEMPEÑO**

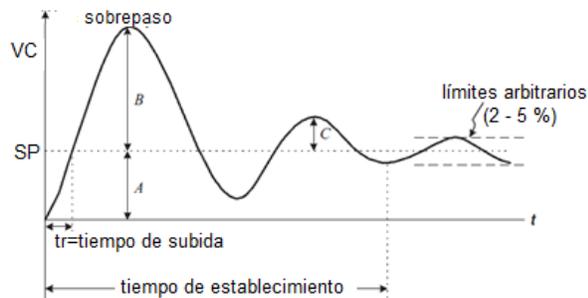
Los criterios de desempeño son medidas cuantitativas del comportamiento del sistema de control, en la mayoría de los casos las especificaciones en el dominio del tiempo como el sobrepaso máximo, el tiempo de subida y el tiempo de establecimiento normalmente se emplean como criterios del desempeño temporal del sistema de control (Kuo, 1996).

Es importante mencionar que ciertos métodos de sintonización de controladores PID están diseñados para optimizar alguna función de costo dependiente del error. El objetivo principal de estos métodos es determinar la combinación de parámetros que minimicen esta función de costo. Comúnmente se utilizan criterios como la integral del error absoluto (IAE) o la integral absoluta del tiempo por el error absoluto (ITAE) como función de costo. Estas funciones se conocen como criterios integrales de desempeño y generalmente son usados en conjunto con los criterios de desempeño temporales para dar una calificación cuantitativa de la respuesta del sistema de control (Dorf and Bishop, 2008).

### **4.2.1 Criterios de desempeño temporales**

En esta sección se detallan algunos conceptos relacionados con la respuesta temporal del proceso, ya que dependiendo del proceso se debe decidir qué tipo de respuesta es óptima o por lo menos aceptable. En la ilustración 4.1 se observa la típica respuesta temporal de un proceso a un cambio en el SP.

### Ilustración 4.1: Respuesta de un proceso a cambios en el SP



Fuente: modificado de (Svrcek et al, 2006)

**Sobrepaso máximo** es la cantidad en la cual la respuesta excede el valor final en estado estacionario. De la ilustración 4.1 el sobrepaso máximo puede ser definido según la ecuación 4.1:

$$\frac{B}{A} = e^{-\pi\zeta/\sqrt{1-\zeta^2}} \quad (4.1)$$

Donde  $A$  es el valor del SP,  $B$  es el valor en el que la VC excede el valor del SP y  $\zeta$  es el coeficiente de amortiguamiento.

**Razón de amortiguación:** es la cantidad en la cual la amplitud de una oscilación disminuye con respecto a la amplitud de la oscilación precedente, en la ilustración 4.1, se puede apreciar esta relación  $C/B$ . Una definición más específica es la razón de amortiguamiento a un cuarto (QDR<sup>1</sup>) definida por la ecuación 4.2, la cual se sitúa entre una respuesta críticamente amortiguada y sub amortiguada (svrcek, et al, 2006).

$$QDR = \frac{C}{B} = \frac{1}{4} \quad (4.2)$$

Donde  $C$  es el valor que alcanza la VC en la segunda oscilación y  $B$  es el valor máximo de la VC en la oscilación precedente. La razón de amortiguamiento es usada a menudo para establecer si un controlador provee una respuesta satisfactoria.

**Tiempo de subida ( $tr$ ):** es el tiempo inicial requerido por la respuesta transitoria para alcanzar el valor final en estado estacionario, algunos autores lo definen como el tiempo que requiere la respuesta para elevarse desde el 10 % hasta el 90 % del valor final en estado estacionario.

**Tiempo de respuesta o de establecimiento:** es el tiempo requerido por la respuesta para establecerse dentro de ciertos límites especificados. Típicamente estos límites están entre 2 y 5 % del valor en estado estacionario de la VC.

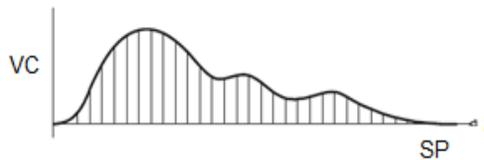
<sup>1</sup> Quarter Decay Ratio

## 4.2.2 Criterios Integrales de Desempeño

Los criterios que se describen a continuación tienen la ventaja de ser más precisos ya que sólo una única combinación de parámetros minimizará la integral correspondiente. La ilustración 4.2 muestra la curva de respuesta de un proceso cualquiera, cómo se ilustra en la ecuación 4.3, la integral del error IE, es el área bajo la curva siendo  $e = VC - SP$ . Cuando se usan criterios integrales de desempeño el objetivo es minimizar esta área bajo la curva.

$$IE = \int_0^{\infty} e dt \quad (4.3)$$

### Ilustración 4.2: Curva de respuesta de un proceso



Fuente: (Svrcek et al, 2006) p. 120

Es imposible minimizar la integral del error IE, sin pagar un costo en algún aspecto del desempeño del sistema de control, por ejemplo la subamortiguación produce mínima área bajo la curva pero se presentan oscilaciones considerables (Svrcek et al, 2006).

**Integral Absoluta del Error (IAE):** se define según la ecuación 4.4 y consiste en minimizar la integral del valor absoluto del error, por lo tanto se toman en cuenta las áreas negativas.

$$IAE = \int_0^{\infty} |e| dt \quad (4.4)$$

Este criterio provee una respuesta similar a las obtenidas con el criterio de desempeño de la razón de amortiguación de  $\frac{1}{4}$ .

**Integral del error al cuadrado (ISE):** En este criterio se usa el cuadrado del error, por lo que se penalizan los errores grandes más que los errores pequeños. Aquí se provee una respuesta más conservadora, por ejemplo, un rápido retorno al SP. La ISE está definida por la ecuación 4.5:

$$ISE = \int_0^{\infty} e^2 dt \quad (4.5)$$

**Integral Absoluta del Error por el tiempo (ITAE):** consiste en integrar el valor absoluto del error multiplicado por el tiempo, se define en la ecuación 4.6.

$$ITAE = \int_0^{\infty} t |e| dt \quad (4.6)$$

### 4.3 SINTONIZACIÓN DE CONTROLADORES PID

La sintonización de los controladores PID consiste en la determinación de los parámetros óptimos ( $K_c$ ,  $T_i$  y  $T_d$ ) del controlador, que logren un comportamiento aceptable del sistema de control, de acuerdo a ciertas especificaciones de diseño.

Por lo general los métodos de sintonización de controladores PID caen en alguna de estas categorías (Seborg, 2004):

1. Métodos basados en la respuesta temporal.
2. Métodos basados en síntesis directa (DS).
3. Métodos basados en control por modelo interno (IMC).
4. Métodos basados en la respuesta en frecuencia.
5. Métodos de sintonización en línea (*autotuning*).

Las cuatro primeras categorías se basan en modelos del proceso y pueden ser utilizados para configurar los parámetros del controlador antes de instalarlo en el lazo de control. En algunas ocasiones, esta configuración inicial debe ser ajustada manualmente para obtener las características deseadas del sistema de control. Esta sintonización en línea, se basa en simples pruebas experimentales en lazo cerrado sobre el proceso (Astrom, 2006) y es requerida porque los modelos de proceso usados para calcular la configuración inicial del controlador PID, no son exactos. El objetivo de los métodos que caen en estas categorías es proveer una buena configuración inicial del controlador PID, que posteriormente debe ser mejorada mediante una sintonía fina en línea, en caso de ser necesaria. Es importante resaltar que la sintonía fina es un proceso que puede consumir una gran cantidad de tiempo, por tal razón es útil tener una buena configuración inicial del controlador PID para minimizar el esfuerzo y tiempo requeridos en la sintonización del controlador (Seborg, 2004).

Los métodos de *autotuning* son métodos en los cuales el controlador PID está conectado al proceso y se sintoniza automáticamente basado en las demandas del usuario. En (Astrom, 2006) se puede encontrar una excelente referencia sobre los métodos de sintonía automática.

Los métodos de respuesta temporal se basan en la curva de reacción del proceso o en la función de transferencia para modelos de primer orden más tiempo muerto (POMTM). En estos métodos los parámetros del controlador se expresan mediante simples formulas o reglas de sintonización. La sencillez de estos métodos hacen que puedan ser utilizados en cursos introductorios de control de procesos (Astrom, 2006).

Para sintonizar un controlador PID partiendo de las características dinámicas del proceso, se deben seguir los siguientes pasos:

1. Identificar los parámetros dinámicos de un modelo POMTM asociado al proceso, mediante un gráfico de la curva de reacción.
2. Calcular los parámetros del controlador PID ( $K_c, T_i, T_d$ ), a través de fórmulas apropiadas y con los parámetros del modelo POMTM obtenido.

#### 4.3.1 Identificación de Procesos

Para realizar la sintonización de los controladores PID es importante tener un buen entendimiento del proceso a controlar, ya que dependiendo de las características de éste, se encuentran los parámetros adecuados para poder controlarlo; este entendimiento puede ser obtenido desarrollando un modelo teórico del proceso usando balances de energía, balance de masa y leyes físico-químicas, sin embargo esto no siempre es fácil, además es un procedimiento que consume demasiado tiempo y esfuerzo. Otra forma es utilizar una aproximación de las dinámicas del proceso mediante modelos lineales de primer o segundo orden (Liptak, 2006).

Muchos procesos industriales pueden ser representados mediante el modelo de primer orden más tiempo muerto (POMTM) representado por la ecuación 4.7

$$G_{(s)} = \frac{K_p}{\tau s + 1} e^{-t_m s} \quad (4.7)$$

Donde:

$K_p$  = Ganancia del proceso

$\tau$  = constante de tiempo del proceso

$t_m$  = tiempo muerto

Una forma de obtener los parámetros del modelo POMTM es usando métodos basados en la curva de reacción. Estos procedimientos son útiles y rápidos para determinar modelos lineales del proceso, aunque estos modelos solamente son válidos para un punto de operación determinado y no toman en cuenta los comportamientos de orden superior y la no linealidad del proceso (Liptak, 2006).

El procedimiento para obtener la curva de reacción del proceso consiste en los siguientes pasos:

1. Establecer la salida del controlador en modo manual.
2. Esperar a que la variable controlada (salida del sistema) llegue a un valor constante.
3. Aplicar un cambio de valor al esfuerzo de control. Usualmente este cambio es del 5 o 10 %. La duración de este cambio debe ser lo bastante grande para permitir que la variable controlada llegue a un nuevo valor de estado estable.
4. Registrar la respuesta de la variable controlada y el esfuerzo de control.

5. Aplicar un método de identificación basado en la curva de reacción para obtener los parámetros del modelo POMTM.

En el presente caso de estudio se utilizaron dos métodos basados en la curva de reacción para la obtención de los parámetros del modelo POMTM, sin embargo cabe mencionar que existen diferentes métodos para la obtención de modelos del proceso, algunos de estos métodos se detallan en el anexo F.

#### 4.3.1.1 Método de los dos puntos

En el método de los dos puntos propuesto por Smith (Smith, 2002), los valores de  $\tau$  y  $t_m$  se seleccionan de tal forma que el modelo coincida con la respuesta del proceso en dos puntos; estos puntos son los tiempos en los cuales la VC alcanza el 28.3 % ( $t_{28.3\%}$ ) y el 63.8% ( $t_{63.2\%}$ ) de su valor final en estado estable. Los parámetros del modelo POMTM se calculan usando expresiones de la ecuación 4.8:

$$K_p = \frac{\left( \frac{\Delta y(t)}{span} \right)}{\Delta u(t)} \quad \tau = \frac{3}{2} (t_{63.2\%} - t_{28.3\%}) \quad t_m = t_{63.2\%} - \tau \quad (4.8)$$

Donde:

$K_p$  = Ganancia del proceso

$\tau$  = constante de tiempo del proceso

$t_m$  = tiempo muerto del proceso

$\Delta y(t)$  = variación de la salida del proceso  $y(t) = y_f(t) - y_0(t)$

$\Delta u(t)$  = variación de la entrada al proceso  $u(t) = u_f(t) - u_0(t)$

$span$  = rango del transmisor

$t_{28.3\%}$  = tiempo en que la salida alcanza el 28.3 % de su valor final en estado estable

$t_{63.2\%}$  = tiempo en que la salida alcanza el 63.2 % de su valor final en estado estable

#### 4.3.1.2 Método de Arrieta y Alfaro

En el método de Arrieta y Alfaro (Arrieta and Ruiz, 2003) los valores de  $\tau$  y  $t_m$  se seleccionan de tal forma que el modelo coincida con la respuesta del proceso en dos puntos; estos puntos cuales la VC alcanza el 25 % ( $t_{25\%}$ ) y el 75 % ( $t_{75\%}$ ) de su valor final en estado estable. Los parámetros del modelo POMTM se calculan usando las expresiones de la ecuación 4.9:

$$K_p = \frac{\left( \frac{\Delta y(t)}{span} \right)}{\Delta u(t)} \quad \tau = 0.910(t_{75\%} - t_{25\%}) \quad t_m = 1.262t_{25\%} - 0.262t_{75\%} \quad (4.9)$$

Donde:

$K_p K_p$  = Ganancia del proceso

$\tau$  = constante de tiempo del proceso

$t_m$  = tiempo muerto del proceso

$\Delta u(t)$  = variación de la entrada al proceso  $u(t) = u_f(t) - u_0(t)$

$span$  = rango del transmisor

$t_{25\%}$  = tiempo en que la salida alcanza el 25 % de su valor final en estado estable

$t_{75\%}$  = tiempo en que la salida alcanza el 75 % de su valor final en estado estable

### 4.3.2 Métodos de sintonización para controladores PID

Cómo se mencionó anteriormente, en la literatura técnica se pueden encontrar diversos métodos de sintonización para controladores PID. Muchos de estos métodos parten del método desarrollado por Ziegler - Nichols basado en la respuesta temporal del sistema. En esta sección se detallan los métodos de Ziegler - Nichols y algunos métodos relacionados.

#### 4.3.2.1 Método de Ziegler - Nichols en lazo abierto

Este método consiste en aplicar un escalón unitario a la planta, medir la respuesta temporal del sistema y a partir de esta respuesta, identificar los parámetros de un modelo POMTM asociado al proceso. En el anexo F se detallan los pasos que se llevan a cabo en este procedimiento de identificación conocido como método de la tangente.

Una vez se obtengan los parámetros del modelo POMTM, se aplican las formulas de ajuste dadas en la tabla 4.1 para calcular los parámetros de controladores P, PI y PID (Ziegler and Nichols, 1942).

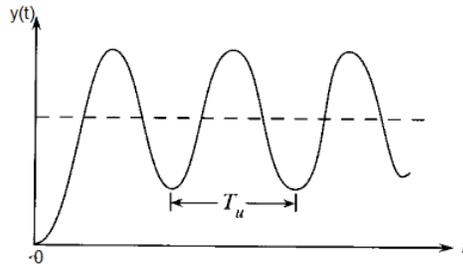
**Tabla 4.1: Formulas de ajuste de controladores P, PI y PID para el método de Ziegler - Nichols en lazo abierto.**

Controlador	$K_c$	$T_i$	$T_d$
P	$\tau t_m$		
PI	$0.9\tau t_m$	$0.3t_m$	
PID	$1.2\tau t_m$	$2t_m$	$0.5t_m$

#### 4.3.2.2 Método de Ziegler – Nichols en lazo cerrado

El segundo método desarrollado por Ziegler – Nichols consiste en utilizar un controlador P y encontrar la ganancia  $K_c$  que genere oscilaciones sostenidas en el proceso. En la sección 3.2.2 se comentó que esta ganancia  $K_c$  se denomina ganancia última y corresponde al valor límite que puede tener la ganancia proporcional antes que el sistema se desestabilice. La ilustración 4.4 muestra la oscilación sostenida que se genera con este controlador a la salida del proceso.  $T_u$ , corresponde al valor del período último de oscilación.

#### Ilustración 4.3: Oscilaciones sostenidas de la variable controlada



Fuente: modificado de (Chen, 2006)

Con la información obtenida, es decir los valores de la ganancia última ( $K_u$ ) y el período último de oscilación ( $T_u$ ), se aplican las formulas de ajuste dadas en la tabla 4.2 para calcular los parámetros de controladores P, PI y PID (Chen, 2006):

**Tabla 4.2: Formulas de ajuste de controladores P, PI y PID para el método de Ziegler - Nichols en lazo cerrado**

Controlador	$K_c$	$T_i$	$T_d$
P	$0.5K_u$		
PI	$0.4K_u$	$0.8T_u$	
PID	$0.6K_u$	$0.5T_u$	$0.12T_u$

Se han realizado muchas propuestas de modificación a los métodos de Ziegler – Nichols (Astrom, 2006). A continuación se describen dos métodos basados en los métodos de Ziegler - Nichols:

#### 4.3.2.3 Método de Chien, Hrones, and Reswick (CHR)

Este método ofrece los parámetros del controlador que proveen una "respuesta más rápida sin sobre impulso" o una "respuesta más rápida con 20% de sobre impulso". Para sintonizar un controlador PID de acuerdo al método CHR, los parámetros del modelo POMTM se calculan mediante el método de la tangente. Chien, Hrones y

Reswick encontraron que la sintonización para controladores que operan en lazos regulatorios era diferente para controladores que operan en lazos servocontrolados (Astrom, 2006). La tabla 4.3 muestra las formulas de ajuste en el método CHR:

**Tabla 4.3: Formulas de ajuste del método CHR**

Modo de operación	$K_c$	$T_i$	$T_d$	Comentarios
<b>Regulatorio</b>	$\frac{0.95\tau}{K_p t_m}$	$2.38t_m$	$0.42t_m$	0 % sobreimpulso: $0.11 < \frac{t_m}{\tau} < 1$
	$\frac{1.2\tau}{K_p t_m}$	$2t_m$	$0.42t_m$	20 % sobreimpulso: $0.11 < \frac{t_m}{\tau} < 1$
<b>Servocontrol</b>	$\frac{0.6\tau}{K_p t_m}$	$\tau$	$0.5t_m$	0 % sobreimpulso: $0.11 < \frac{t_m}{\tau} < 1$
	$\frac{0.95\tau}{K_p t_m}$	$1.36\tau$	$0.47t_m$	20 % sobreimpulso: $0.11 < \frac{t_m}{\tau} < 1$

#### 4.3.2.4 Método de Cohen – Coon

El principal criterio de diseño es el rechazo a disturbios en lazos de control regulatorio y los criterios de desempeño que utilizaron Cohen – Coon fueron: mínima integral del error (IE), sobreimpulso mínimo y razón de amortiguamiento de un cuarto (Astrom, 2006). La tabla 4.4 muestra las formulas de ajuste en el método Cohen – Coon (O'Dwyer, 2009).

**Tabla 4.4: Formulas de ajuste del método Cohen - Coon**

Modo de operación	$K_c$	$T_i$	$T_d$	Comentarios
<b>Regulatorio</b>	$\frac{\tau}{K_p t_m} \left( \frac{4}{3} + \frac{t_m}{4\tau} \right)$	$t_m \left( \frac{32 + 6 \frac{t_m}{\tau}}{13 + 8 \frac{t_m}{\tau}} \right)$	$t_m \left( \frac{4}{11 + 2 \frac{t_m}{\tau}} \right)$	

Es importante recordar que los métodos de sintonización comentados en esta sección, no son los únicos métodos existentes ni los mejores, la elección de estos métodos dependerá de las especificaciones de diseño y los requerimientos del sistema a controlar. En (O'Dwyer, 2009) se resume una gran variedad de métodos de sintonización para controladores PID.

#### 4.4 RESUMEN

En este capítulo se discutieron conceptos básicos referentes a la sintonización de controladores PID. Se comentó la importancia de las especificaciones de diseño, dado que estas se utilizan para describir que debe hacer el sistema de control y cómo hacerlo. También se hizo referencia a los criterios de desempeño integrales y temporales, los cuales son útiles para evaluar el comportamiento de los sistemas de control, se mencionó que existen otros criterios de desempeño además de los mencionados en el presente trabajo y se sugirió bibliografía en la cual se puede encontrar referencia sobre estos criterios.

Se mencionaron las principales categorías en las cuales se pueden catalogar los métodos de sintonización de controladores PID, sin embargo se hizo hincapié en los métodos basados en la respuesta temporal, muchos de los cuales hacen uso de un modelo POMTM cuyos parámetros son calculados mediante algún método de identificación basado en la curva de reacción del proceso.

Por último, se mencionaron los métodos de Sintonización de Ziegler - Nichols en lazo abierto y en lazo cerrado. Se comentó que muchos métodos de sintonización basados en la respuesta temporal son modificaciones a los métodos de Ziegler - Nichols, como ejemplo se mencionaron los métodos de Cohen - Coon y Chien, Hrones y Reswick. También se sugirió bibliografía en la cual se puede encontrar ampliación sobre métodos de sintonización para controladores PID.

## **5 PROTOTIPADO RÁPIDO DE CONTROL DE TEMPERATURA PARA EL MÓDULO MIC955 DE LA EMPRESA FEEDBACK**

En el capítulo 1 se comentó la importancia de los sistemas RCP y los diferentes enfoques de RCP que existen. En este capítulo se detallan los pasos que deben seguirse para realizar Prototipado Rápido de Control de temperatura en el módulo MIC 955 de la empresa Feedback.

### **5.1 PROCEDIMIENTO PARA REALIZAR RCP DE TEMPERATURA EN EL MÓDULO MIC955**

En esta sección se detalla el procedimiento para realizar RCP que se llevó a cabo en el presente caso de estudio. En este procedimiento se definieron siete (7) actividades, en las cuales están incluidas todas las fases necesarias para realizar RCP (Hölttä, Palmroth, y Eriksson, 2004):

1. Definición de los objetivos y/o requerimientos del sistema de control y selección de la estructura del controlador PID.
2. Identificación y validación de un modelo POMTM del proceso térmico que ocurre en el módulo MIC955.
3. Sintonización de los parámetros del controlador PID mediante métodos de ajuste válidos para el modelo POMTM.
4. Diseño, implementación y simulación de una versión prototipo del sistema de control mediante diagramas de bloques.
5. Diseño y generación del código ejecutable (tarea en tiempo real) de la versión prototipo del sistema de control en el sistema de RCP.
6. Pruebas operacionales de la versión prototipo del sistema de control en el módulo MIC955 (ejecución de la tarea en tiempo real).
7. Evaluación mediante criterios de desempeño integrales y temporales del comportamiento del sistema de control.

Es importante recordar que el procedimiento anterior no corresponde en ningún caso a una guía obligatoria para realizar RCP, al contrario, el RCP es una herramienta que da libertad al diseñador para seguir cualquier otro procedimiento con miras a cumplir sus objetivos.

### **5.1.1 Definición de los objetivos y/o requerimientos del sistema de control y selección de la estructura del controlador PID**

La primera actividad consiste en determinar las especificaciones de diseño (ver sección 4.1) y los objetivos que el sistema de control debe conseguir. En el capítulo 3 se comentó que un controlador PID puede operar de dos formas:

- a) Regulador: en este caso el valor deseado o SP permanece constante y se desea que el sistema sea insensible a disturbios.
- b) Servocontrol: el SP puede cambiar en el tiempo, por lo tanto se requiere un buen seguimiento de este.

Es obligatorio especificar con claridad los requerimientos del sistema de control, pues la respuesta un controlador PID que opera en lazos servocontrolados es muy diferente a la respuesta del mismo controlador en lazos de control regulatorio.

Por último, con base en estos requerimientos se puede decidir cuál será la estructura del controlador PID (Controlador con Filtro Derivativo o Controlador PID Clásico) que se va a utilizar en el control del proceso térmico del módulo MIC955 y las características adicionales que puedan agregarse al controlador PID, por ejemplo, si se requiere activar la protección AWBT, si es necesario un módulo de seguimiento del esfuerzo de control manual, etc.

### **5.1.2 Identificación y validación de un modelo POMTM del proceso térmico que ocurre en el módulo MIC955**

Esta actividad consiste en definir un punto de operación determinado (*Potencia, Temperatura*) y con base en este punto de operación, identificar los parámetros de un modelo POMTM asociado al proceso térmico que ocurre en el módulo MIC955, utilizando el procedimiento detallado en la sección 4.3.1. En la sección 5.3 se muestran los resultados de la identificación del proceso.

### **5.1.3 Sintonización de los parámetros del controlador PID mediante métodos de sintonización válidos para el modelo POMTM**

El objetivo de esta actividad es determinar los parámetros del controlador PID que permitan cumplir con los requerimientos y especificaciones que se establecieron para el sistema de control. El procedimiento de sintonización del controlador PID para operar en lazo de control regulatorio y servo controlados consiste de los siguientes pasos:

1. Con base en los requerimientos del sistema de control (operación en lazos regulatorios o servo controlados), la estructura del controlador PID (Controlador con Filtro Derivativo o Controlador PID Clásico) y la información que provee el modelo POMTM obtenido, se debe elegir los métodos de sintonización válidos para el modelo POMTM.

2. Calcular los parámetros del controlador mediante las fórmulas de ajuste que provee cada método de sintonización.

En la sección 5.4 se detallan los métodos de sintonización seleccionados para las dos estructuras de controladores PID implementadas en el sistema de RCP (ver sección 3.5)

#### **5.1.4 Diseño y simulación de una versión prototipo del sistema de control mediante diagramas de bloques.**

El diseño de las versiones prototipo de sistema de control en el sistema de RCP para el módulo MIC955 de la empresa Feedback, se realiza con la herramienta Scicos incluida en el software DSCAC Scilab. En la sección 5.5 se detallan los pasos para diseñar la versión prototipo mediante diagramas de bloques. En esta fase se realiza una simulación del comportamiento del sistema de control ante una serie eventos definidos en el tiempo, con el modelo POMTM que se obtuvo en el procedimiento de identificación. Estas pruebas operacionales se efectúan para el controlador PID operando en lazos de control regulatorio y servocontrolados, con la protección AWBT tanto activada como desactivada.

##### **5.1.4.1 Pruebas operacionales de la versión prototipo del sistema de control en lazo de control regulatorio**

Inicialmente se debe llevar la variable controlada a un valor cercano al punto de operación elegido, realizando en forma manual los siguientes pasos:

1. Configurar el controlador PID en modo automático.
2. Sintonizar el controlador PID con los valores de los parámetros obtenidos en la fase 3.
3. Habilitar la protección AWBT.
4. Seleccionar un SP (45 °C) cercano al punto de operación en el cual el modelo fue obtenido (16% de Potencia, 50°C).
5. Ejecutar el sistema de control. Esperar hasta que el proceso se estabilice.
6. Deshabilitar la protección AWBT.

El orden en el cual debe ejecutarse la serie de eventos que ocurren en el lazo de control regulatorio, es el siguiente:

1. Aplicar un cambio en el SP (55 °C) en el instante  $t = 360$  segundos (6 minutos). Esperar hasta el proceso se estabilice.
2. Aplicar un disturbio de magnitud pequeña (encender el ventilador) con una duración igual a 180 segundos (3 minutos) en el instante  $t = 840$  (14 minutos).
3. Aumentar la magnitud del disturbio en el instante  $t = 1020$  segundos (17 minutos).
4. Cancelar el disturbio (apagar el ventilador) en el instante  $t = 1200$  (20 minutos). Esperar hasta que el proceso se estabilice.

5. Cambiar de modo de operación de automático a manual en el instante  $t = 1560$  segundos (26 minutos).
6. Cambiar el valor del EC manual de 25 % a 36 % en el instante  $t = 1620$  (27 minutos).
7. Cambiar el valor del EC manual de 36 % a 16 % en el instante  $t = 1740$  (29 minutos).
8. Cambiar el valor del EC manual de 16 % a 25 % en el instante  $t = 1960$  (31 minutos).
9. Cambiar de modo de operación de manual a automático en el instante  $t = 1980$  segundos (33 minutos).
10. Registrar la respuesta del proceso y las señales de interés hasta el instante  $t = 2500$  segundos (41.6 minutos).
11. Repetir los pasos 1 a 10 con la protección AWBT activada.

Esta serie de pasos se deben automatizar, pues al realizarlos de manera manual resulta complejo garantizar que se cumplan los instantes de tiempo en los cuales estos pasos deben ejecutarse. En la sección 5.6.1 se muestran los resultados obtenidos en la fase de simulación de la versión prototipo del sistema de control regulatorio.

#### **5.1.4.2 Pruebas operacionales de la versión prototipo del sistema de control en lazos servocontrolados**

Inicialmente se debe llevar la variable controlada a un valor cercano al punto de operación elegido, realizando en forma manual los siguientes pasos:

1. Configurar el controlador PID en modo automático.
2. Sintonizar el controlador PID con los valores de los parámetros obtenidos en la fase 3.
3. Habilitar la protección AWBT.
4. Seleccionar un SP (45 °C) cercano al punto de operación en el cual el modelo fue obtenido (16% de Potencia, 50°C).
5. Ejecutar el sistema de control. Esperar hasta que el proceso se estabilice.
6. Deshabilitar la protección AWBT.

El orden en el cual debe ejecutarse la serie de eventos que ocurren en el lazo de servo control, es el siguiente:

1. Aplicar un cambio en el SP (SP=50 °C) en el instante  $t = 300$  segundos (5 minutos). Esperar hasta el proceso se estabilice.
2. Aplicar un cambio en el SP (SP=55 °C) en el instante  $t = 600$  segundos (10 minutos). Esperar hasta el proceso se estabilice.
3. Aplicar un cambio en el SP (SP=45 °C) en el instante  $t = 900$  segundos (15 minutos). Esperar hasta el proceso se estabilice.

4. Aplicar un cambio en el SP (SP=50 °C) en el instante  $t = 1200$  segundos (20 minutos).
5. Registrar la respuesta del proceso y las señales de interés hasta el instante  $t = 1500$  segundos (25 minutos).
6. Repetir los pasos 1 a 5 con la protección AWBT activada.

Esta serie de pasos se deben automatizar, pues al realizarlos de manera manual resulta complejo garantizar que se cumplan los instantes de tiempo en los cuales estos pasos deben ejecutarse. En la sección 5.6.2 se muestran los resultados obtenidos en la fase de simulación de la versión prototipo del sistema servocontrolado.

#### **5.1.5 Diseño y generación del código ejecutable (tarea en tiempo real) de la versión prototipo del sistema de control en el sistema de RCP**

En esta fase se diseña la versión prototipo del sistema de control con ayuda de bloques de tiempo real adecuados. En la sección 5.5 se detalla los pasos necesarios para realizar la implementación y posterior generación de la tarea de tiempo real asociada a la versión prototipo del sistema de control.

#### **5.1.6 Pruebas operacionales de la versión prototipo del sistema de control en el módulo MIC955 (ejecución de la tarea en tiempo real)**

Esta fase consiste en realizar pruebas operacionales de la versión prototipo del sistema de control ante la misma serie de eventos que se definió en la simulación. La versión prototipo del sistema de control se aplica directamente sobre el proceso térmico que ocurre en el módulo MIC955, mediante la ejecución de la tarea en tiempo real en el sistema de RCP. En las secciones 5.6.1 y 5.6.2 se muestran los resultados obtenidos en la fase de ejecución en tiempo real de las versiones prototipo del sistema de control regulatorio y servocontrolado.

#### **5.1.7 Evaluación mediante criterios de desempeño integrales y temporales del comportamiento de la versión prototipo del sistema de control, en lazos de control tanto regulatorios como servocontrolados**

La última actividad consiste en medir cuantitativamente el desempeño de la versión prototipo del sistema de control mediante los criterios de desempeño detallados en la sección 4.2, para hacer esto se utiliza el DSCAC Scilab/Scicos. En la sección 5.7 se presentan los resultados obtenidos en el presente caso de estudio.

### **5.2 REQUERIMIENTOS Y OBJETIVOS QUE DEBE CUMPLIR EL SISTEMA DE CONTROL DE TEMPERATURA EN EL MÓDULO MIC955**

La implementación de versiones prototipo de sistemas de control en el sistema de RCP para el módulo MIC955 debe permitir la comparación cuantitativa mediante

criterios de desempeño temporales e integrales del comportamiento de estos prototipos, ante diferentes métodos de ajuste de controladores PID.

Por tal razón se someterá las versiones prototipo de los sistemas de control a pruebas operacionales en simulación y en ejecución de tareas de tiempo real, en lazos de control regulatorio con protección AWBT activada y desactivada. Luego se harán pruebas operacionales en lazos servocontrolados con protección AWBT activada y desactivada. Estas pruebas se realizan para dos estructuras de controladores PID implementadas en el sistema de RCP: el controlador PID con Filtro Derivativo y el controlador PID clásico. Es importante aclarar que no se definen valores concretos para las especificaciones de diseño en el dominio del tiempo, pues la evaluación de los métodos de sintonización arrojará diferentes valores de estos parámetros, los cuales ayudarán a realizar la comparación del desempeño de la versión prototipo del sistema de control.

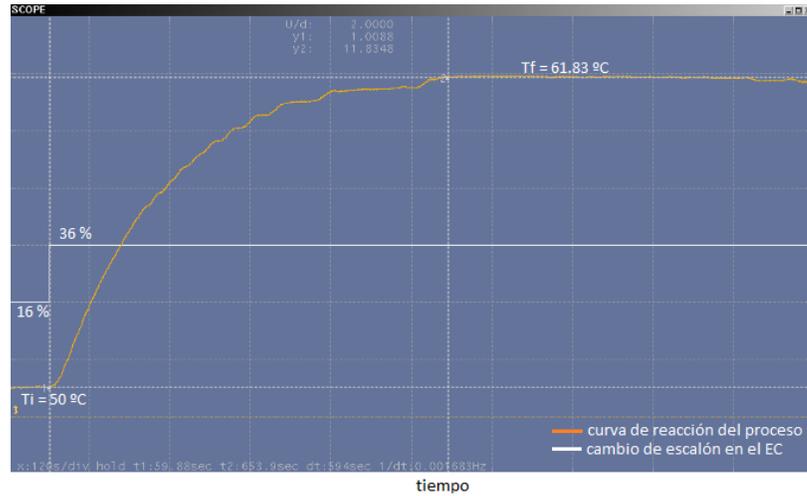
### **5.3 IDENTIFICACIÓN Y VALIDACIÓN DE UN MODELO POMTM DEL PROCESO TÉRMICO QUE OCURRE EN EL MÓDULO MIC955**

En el presente caso de estudio, el punto de operación elegido para la identificación de un modelo POMTM para el proceso de temperatura era (16% de Potencia, 50°C). El procedimiento para la identificación de un modelo POMTM, en ese punto de operación es el siguiente:

1. Con el controlador en modo de operación manual, establecer el EC en un valor de 16 % (el actuador entrega el 16 % de la potencia disponible). Esperar hasta que la VC llegue al valor en estado estacionario.
2. Establecer el valor del EC a 36 % con el objetivo de aplicar un cambio de 20 % en la potencia entregada por el actuador (el actuador entrega el 36 % de la potencia disponible). Esperar hasta que la VC llegue al nuevo valor en estado estacionario.
3. Registrar la respuesta del proceso y la variación en el esfuerzo de control.
4. Con base en los métodos de identificación detallados en la sección 4.3 obtener los parámetros dinámicos del proceso de temperatura (ganancia, constante de tiempo y tiempo muerto).
5. Comparar la respuesta del modelo POMTM con la respuesta del proceso.

La ilustración 5.1 muestra la curva de reacción del proceso térmico que ocurre en el módulo MIC955 ante un cambio del 20 % en el EC manual en el punto de operación elegido (16% de Potencia, 50°C). La curva blanca corresponde al cambio en el EC manual y la traza naranja corresponde a la temperatura en la tira de aluminio anodizado del módulo MIC955.

**Ilustración 5.1: Curva de reacción del proceso térmico que ocurre en el módulo MIC955**



Fuente: propia

Para la identificación de los parámetros de un modelo POMTM asociado al módulo MIC955 en el punto de operación (16% de Potencia, 50°C), se utilizaron dos métodos basados en la curva de reacción del proceso (ver sección 4.3.1):

Con el método de los dos puntos propuesto por Smith (Smith, 2002), se obtuvo el siguiente modelo

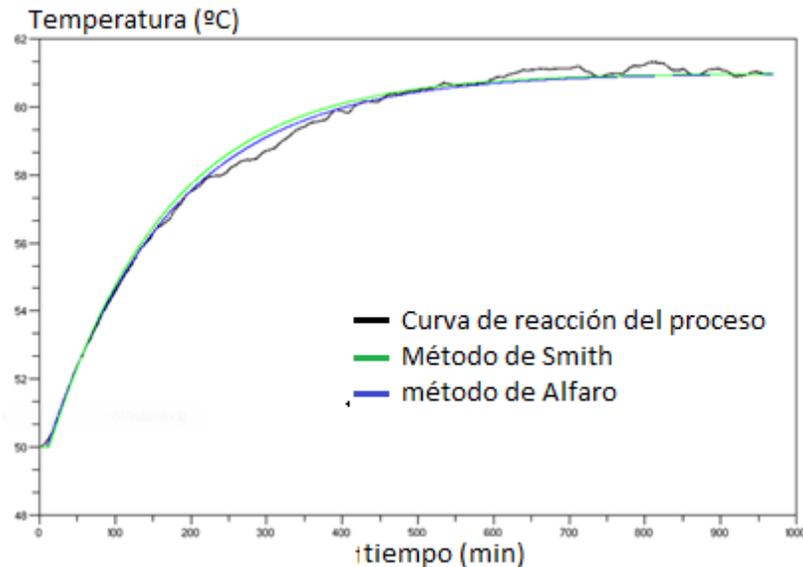
$$G_p(s) = \frac{2.75e^{-12.5s}}{1+154.5s} \quad (5.1)$$

Con el método propuesto por Arrieta y Alfaro (Arrieta Orozco - Alfaro Ruiz, 2003), se obtuvo el siguiente modelo:

$$G_p(s) = \frac{2.75e^{-9.709s}}{1+164.25s} \quad (5.2)$$

La ilustración 5.2 muestra la comparación entre la curva de reacción del proceso térmico representada por la traza de color negro y los modelos POMTM obtenidos con el método de Alfaro (traza azul) y el método de Smith (traza verde).

**Ilustración 5.2: Comparación de los modelos POMTM y Temperatura real**



Fuente: propia

#### **5.4 SELECCIÓN DE LOS MÉTODOS DE SINTONIZACIÓN PARA EL CONTROL DE TEMPERATURA EN EL SISTEMA RCP**

Generalmente las reglas de sintonización de controladores PID para procesos autoregulados se clasifican de acuerdo a la información requerida para su uso o a criterios de desempeño (O'Dwyer, 2009):

En este trabajo se limitaron los métodos de sintonización a los métodos basados en modelos de primer orden más tiempo muerto (POMTM). También es importante conocer la estructura del controlador PID a sintonizar, si los métodos de sintonización fueron desarrollados para controladores operando en lazos de control regulatorio o en lazos servocontrolados y cuál fue el método de identificación utilizado para obtener los parámetros del modelo POMTM.

Para el control del proceso térmico que tiene lugar en el módulo MIC955 se utilizaron dos estructuras de controladores PID, los cuales fueron implementados en el Sistema de Prototipado Rápido de Control (ver sección 3.5).

##### **5.4.1 Métodos de sintonización para el controlador PID con filtro derivativo**

El cuadro 5.1 resume los métodos de ajuste escogidos para sintonizar el controlador PID con filtro derivativo, en lazos de control regulatorios y servocontrolados. Este cuadro está organizado de la siguiente manera: en la parte superior se encuentra el nombre del método de ajuste y el año en el cual fue desarrollado. En la parte superior se muestra información relativa al criterio de desempeño que el método de ajuste

intenta minimizar, el método de identificación utilizado, el modelo POMTM obtenido con este método de identificación y la validez del método de sintonización respecto a los parámetros del modelo POMTM. En la parte inferior izquierda se puede observar las ecuaciones de sintonización para el controlador PID operando en lazos de control regulatorio y la respectiva estimación de los parámetros del controlador PID. En la parte inferior derecha se muestran las ecuaciones de sintonización para lazos servocontrolados y la estimación de los parámetros del controlador PID.

**Cuadro 5.1: Método de sintonización *Minimum IAE* Alfaro – Ruiz (2005a) para lazos de control regulatorio. Fuente: (O’Dwyer, 2009)**

<b>MINIMUN IAE ALFARO – RUIZ (2005a)</b>			
<b>Criterio de desempeño:</b>		<i>minimum IAE</i>	
<b>Método de identificación:</b>		Arrieta Orozco - Alfaro Ruiz	
<b>Modelo del proceso:</b>		$G_p(s) = \frac{2.75e^{-9.709s}}{1+164.25s} \quad \frac{t_m}{\tau} = 0.055$	
<b>Validez:</b>		$0.05 \leq \frac{t_m}{\tau} \leq 2.0$	
<b>Ecuaciones de sintonización y estimación de parámetros del controlador:</b>			
<b>Modo de operación controlador PID</b>			
<b>Regulador</b>		<b>Servocontrolado</b>	
$K_c = \frac{1}{K_m} \left[ 0.2068 + 1.1597 \left( \frac{\tau}{t_m} \right)^{1.0158} \right]$	$K_c = 7.53$	$K_c = \frac{1}{K_m} \left[ 0.3295 + 0.7182 \left( \frac{\tau}{t_m} \right)^{0.9971} \right]$	$K_c = 4.51$
$T_i = \tau \left[ -0.2228 + 1.3009 \left( \frac{t_m}{\tau} \right)^{0.5022} \right]$	$T_i = 15.003$	$T_i = \tau \left[ 0.9781 + 0.3723 \left( \frac{t_m}{\tau} \right)^{0.8456} \right]$	$T_i = 166.24$
$T_d = 0.3953\tau \left( \frac{t_m}{\tau} \right)^{0.8469}$	$T_d = 5.91$ $N = 10$	$T_d = 0.3416\tau \left( \frac{t_m}{\tau} \right)^{0.9414}$	$T_d = 3.91$ $N = 10$
$T_t = \sqrt{T_i T_d}$	$T_t = 9.43$	$T_t = \sqrt{T_i T_d}$	$T_t = 25.51$

#### 5.4.2 Métodos de sintonización para el controlador PID clásico

El cuadro 5.2 resume la información de uno de los métodos de ajuste escogidos para sintonizar el controlador PID clásico, en lazos de control regulatorio y servocontrolados. Este cuadro está organizado de la siguiente manera: en la parte superior se encuentra el nombre del método de ajuste y el año en el cual fue desarrollado. En la parte superior se muestra información relativa al criterio de desempeño que el método de ajuste intenta minimizar, el método de identificación utilizado, el modelo POMTM obtenido con este método de identificación y la validez del método de sintonización respecto a los parámetros del modelo POMTM. En la

parte inferior izquierda se puede observar las ecuaciones de sintonización para el controlador PID operando en lazos de control regulatorio y la respectiva estimación de los parámetros del controlador PID. En la parte inferior derecha se muestran las ecuaciones de sintonización para lazos servocontrolados y la estimación de los parámetros del controlador PID.

**Cuadro 5.2: Método de sintonización *Minimum* ITAE Huang-Chao (1982) para lazos de control regulatorio y servocontrolados. Fuente: (O'Dwyer, 2009)**

<b>MINIMUN ITAE HUANG – CHAO (1982)</b>			
<b>Criterio de desempeño:</b>		<i>minimum</i> ITAE	
<b>Método de identificación:</b>		Método de Smith	
<b>Modelo del proceso:</b>		$G_p(s) = \frac{2.75e^{-12.5s}}{1+154.5s}$	
<b>Validez:</b>		N/A	
<b>Ecuaciones de sintonización y estimación de parámetros del controlador:</b>			
<b>Modo de operación controlador PID</b>			
<b>Regulador</b>		<b>Servocontrolado</b>	
$K_c = \frac{0.745}{K_m} \left( \frac{\tau}{t_m} \right)^{1.036}$	$K_c = 3.66$	$K_c = \frac{0.684}{K_m} \left( \frac{\tau}{t_m} \right)^{0.995}$	$K_c = 3.03$
$T_i = 0.771\tau \left( \frac{t_m}{\tau} \right)^{0.595}$	$T_i = 26.68$	$T_i = \frac{\tau}{0.114 \left( \frac{t_m}{\tau} \right) + 0.986}$	$T_i = 155.24$ , 77.62
$T_d = 0.597\tau \left( \frac{t_m}{\tau} \right)^{1.006}$	$T_d = 7.35$ $N = 8$	$T_d = 0.491\tau \left( \frac{t_m}{\tau} \right)^{1.049}$	$T_d = 5.42$ $N = 8$
$T_t = \sqrt{T_i T_d}$	$T_t = 14$	$T_t = \sqrt{T_i T_d}$	$T_t = 29.02$

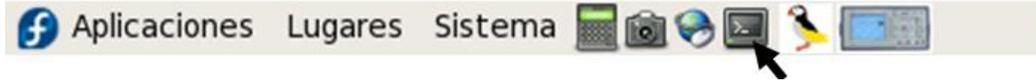
Los cuadros relativos a los demás métodos de sintonización, se pueden ver en el anexo G.

## 5.5 IMPLEMENTACIÓN DE SISTEMAS DE CONTROL EN EL SISTEMA DE RCP

La implementación de los diagramas de bloques en el sistema RCP se realiza mediante el software Scilab/Scicos, el cual puede ser ejecutado de dos maneras:

- Mediante el “Terminal”de GNU Linux: En la ilustración 5.3 se puede ver la barra de accesos directos del Sistema Operativo GNU Linux Fedora 7. Inicialmente se hace clic sobre el icono del “Terminal”, y en la ventana que aparece se debe digitar *scilab* y presionar la tecla enter. Cuando se ejecute este comando, se abrirá la ventana de aplicación de Scilab/Scicos tal como se muestra en la ilustración 5.5.

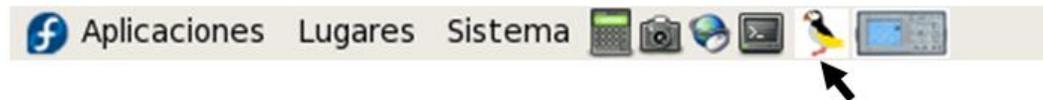
### Ilustración 5.3: Acceso directo a la consola de GNU Linux



Fuente: propia

- La segunda forma de ejecutar el software Scilab/Scicos es dar clic directamente sobre el icono del programa, tal como se muestra en la ilustración 5.4.

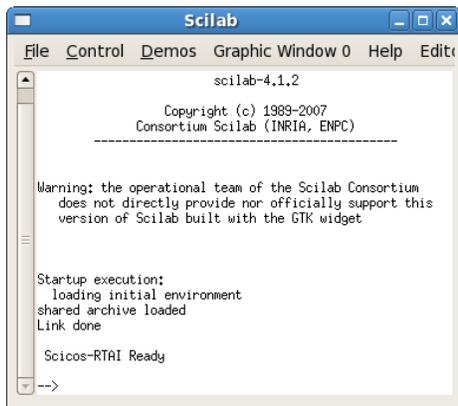
### Ilustración 5.4: Acceso directo al software Scilab/Scicos



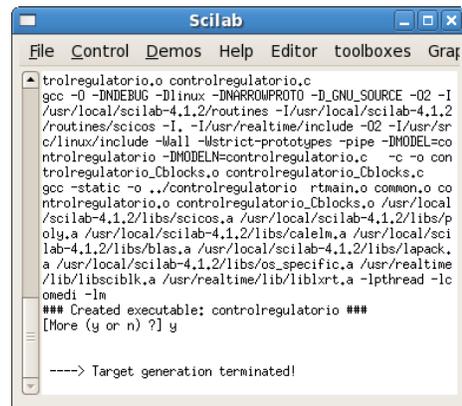
Fuente: propia

La ilustración 5.5 muestra la ventana principal del Software Scilab/Scicos. En la figura 5.5a se observa la ventana inicial y el *prompt* para digitar comandos en Scilab. Es importante recordar que para la generación automática de tareas de tiempo real, esta ventana debe maximizarse. En caso de que la ventana de aplicación de Scilab no se maximice el software hará una confirmación de la creación exitosa del ejecutable en tiempo real, tal como se muestra en la ilustración 5.5b.

### Ilustración 5.5: Ventana de Aplicación del Software Scilab



a) Ventana inicial de Scilab



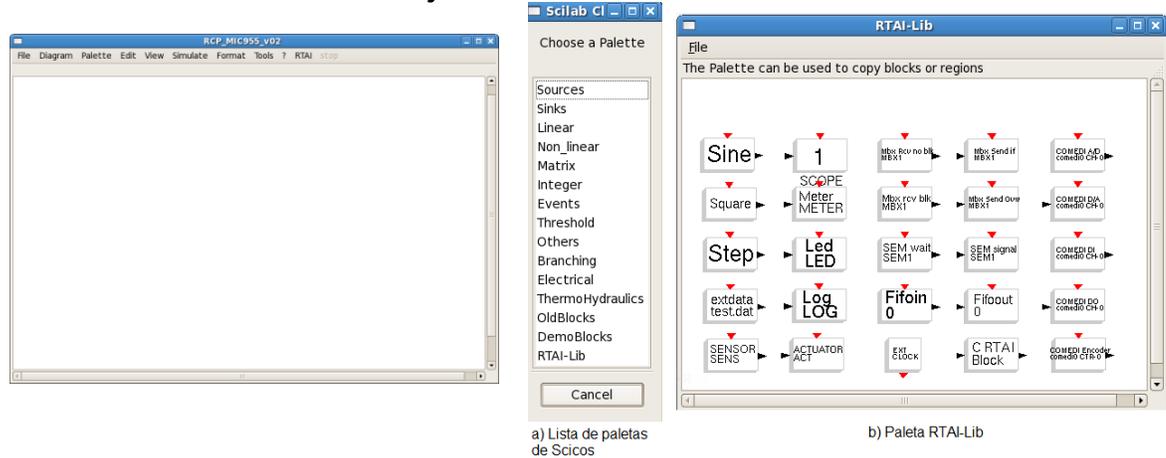
b) Ventana de confirmación de la creación exitosa de un ejecutable

Fuente: Propia

Para ejecutar Scicos, se debe digitar *scicos* en el *prompt* de Scilab. La ilustración 5.6a muestra la ventana de trabajo de Scicos. Para iniciar el diseño del sistema de control

se debe dar clic sobre el menú *Palette*, seleccionar *Palettes* o *Pal Tree* y arrastrar los bloques desde las librerías que provee Scicos, entre las cuales se encuentran: *Sources*, *Sinks*, *Linear*, *Non Linear*, *Branching*, *Events* y la paleta adicional que provee RTAI-Lab para la generación de tareas en tiempo real, llamada *RTAI-Lib*. En la ilustración 5.6b se puede observar la lista de paletas que están incluidas en Scicos y como ejemplo en la parte derecha, se puede observar los bloques incluidos en la paleta *RTAI-Lib*.

**Ilustración 5.6: Área de trabajo de Scicos**



a) Área de trabajo de Scicos

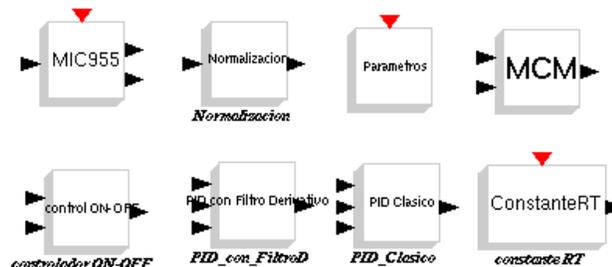
b) Lista de paletas de Scicos y paleta RTAI-Lib

Fuente: Propia

### 5.5.1 Paleta para realizar RCP de temperatura en el Módulo MIC955 de la empresa Feedback.

La ilustración 5.7 muestra los bloques necesarios para realizar RCP en el módulo MIC955. A continuación se detallan algunos de los bloques que hacen parte de la paleta RCP-MIC955. Es importante recordar que algunos de estos bloques, como los controladores PID y el bloque MIC955 fueron detallados en la sección 3.5 y la sección 2.5, respectivamente.

**Ilustración 5.7: Paleta RCP-MIC955**



Fuente: propia

El bloque de normalización, que se muestra en la ilustración 5.8, cumple con la función de transformar la VC que ingresa en unidades de ingeniería, en una variable normalizada que tiene un valor entre 0 y 1. Para hacer esto el bloque implementa la fórmula de conversión dada en la ecuación 5.3:

$$VN = \frac{VC - \text{cero}}{\text{span}} \quad (5.3)$$

Donde:

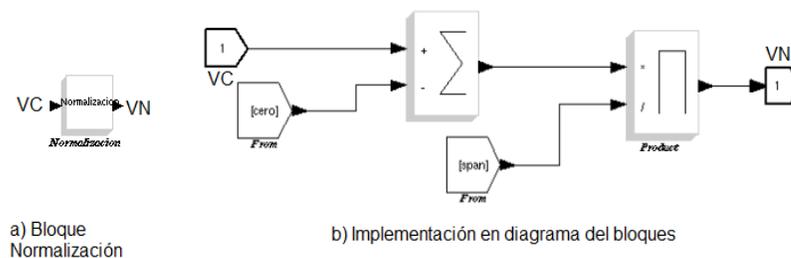
$VN$  = variable normalizada

$VC$  = variable controlada

$\text{cero}$  = límite inferior, configurado en el transmisor.

$\text{span}$  = rango de medición del transmisor

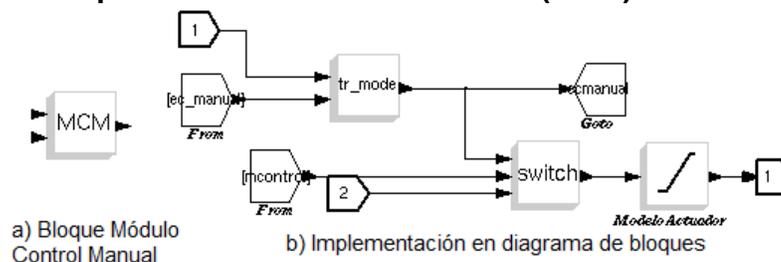
### Ilustración 5.8: Bloque Normalización



Fuente: propia

La ilustración 5.9 muestra el bloque Módulo de Control Manual, el cual es un módulo accesorio para el controlador PID, que se conecta según los requerimientos para el sistema de control.

### Ilustración 5.9: Bloque Módulo de Control Manual (MCM)



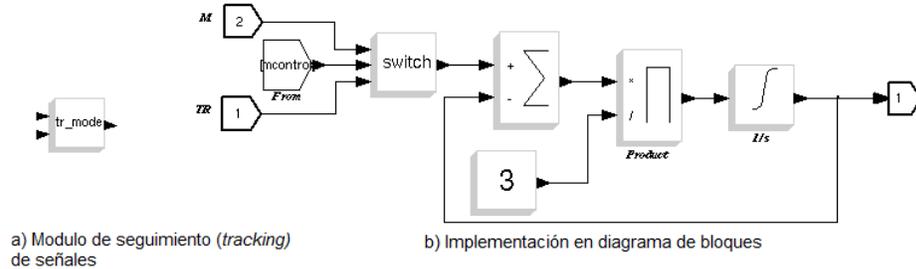
Fuente: propia

Como se puede observar en la ilustración 5.9, el bloque Módulo de Control Manual, incorpora un bloque de seguimiento ( $tr\_mode$ ), cuya implementación puede observarse en la ilustración 5.10. El conmutador ( $switch$ ) cumple con la función de

permitir el paso de la señal de seguimiento o el esfuerzo de control automático, dependiendo de la señal (*mcontrol1*).

La ilustración 5.10 muestra la implementación en diagrama de bloques del módulo de seguimiento (*tr\_mode*). Este bloque incluye un conmutador el cual permite el paso del EC – Manual (entrada 2: M) o la señal de seguimiento (entrada 1, TR), al integrador puro dependiendo de la señal *mcontrol*.

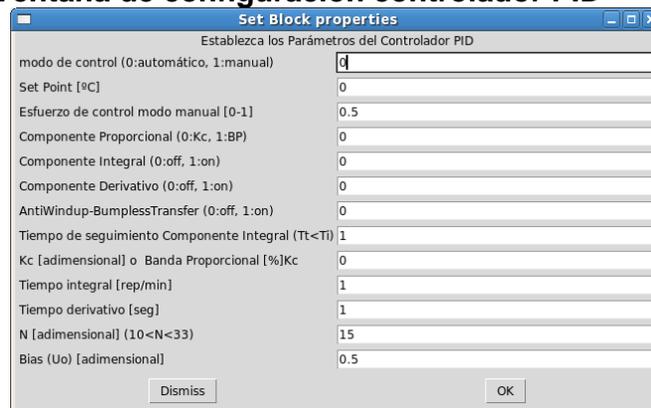
**Ilustración 5.10: Bloque *tr\_mode***



Fuente: propia

La ilustración 5.11 muestra la ventana de configuración del controlador PID, la cual se abre al hacer doble clic sobre el bloque Parámetros. A través de esta ventana es posible configurar el modo de operación (automático o manual), establecer el valor deseado Set Point, habilitar y deshabilitar modos de control, habilitar y deshabilitar la protección AWBT, y establecer los parámetros del controlador PID que fueron obtenidos mediante los métodos de sintonización.

**Ilustración 5.11: Ventana de configuración controlador PID**



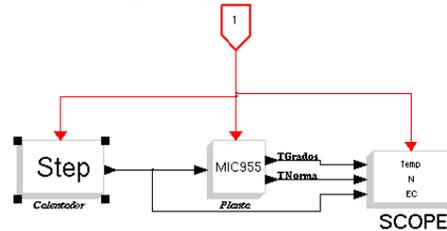
Fuente: propia

### 5.5.2 Diagrama de bloques para la identificación del modelo del proceso

Con ayuda de los bloques *Step*, *Scope* incluidos en la paleta RTAI-Lib y el bloque MIC955 incluido en la paleta RCP-MIC955 se implementó el diagrama de bloques que se muestra en la ilustración 5.12, que permite obtener la curva de reacción del proceso ante una entrada escalón. El bloque *Step* representa el EC necesario para

llevar la variable controlada al punto de operación (16% de Potencia, 50°C), por tal razón en el presente caso de estudio este bloque se encuentra configurado en un valor del EC igual a 16 % (el actuador entrega el 16 % de la potencia disponible).

**Ilustración 5.12: Diagrama de bloques para la identificación del proceso**



Fuente: propia

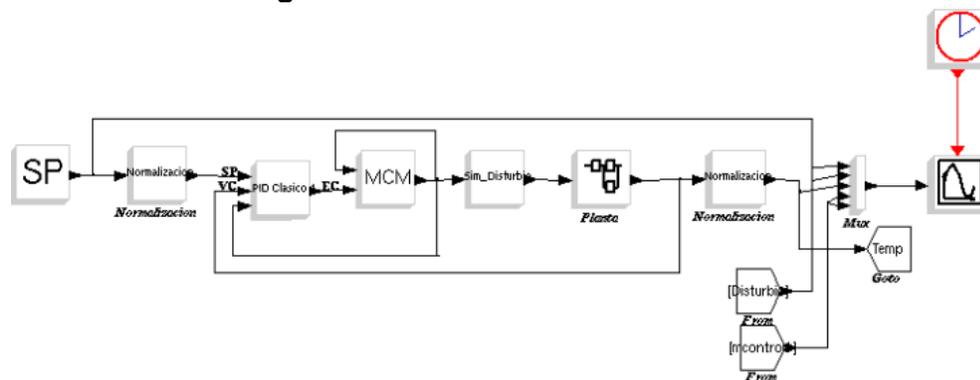
### 5.5.3 Diagramas de bloques de la versión prototipo del sistema de control para el lazo de control regulatorio

En esta sección se detallan los diagramas de bloques de la versión prototipo del sistema de control para el lazo de control regulatorio, utilizados tanto en la fase de simulación como para la generación de la tarea de tiempo real en el sistema de RCP de temperatura para el módulo MIC955.

#### 5.5.3.1 Diagramas de bloques utilizados en la simulación de la versión prototipo del sistema de control regulatorio

La ilustración 5.13 muestra el diagrama de bloques para la simulación de la versión prototipo del sistema de control, se utilizan los bloques *Normalización*, *MCM*, y el bloque *PID Clásico*. Es importante resaltar que este bloque puede ser fácilmente cambiado por el bloque *PID con Filtro Derivativo*.

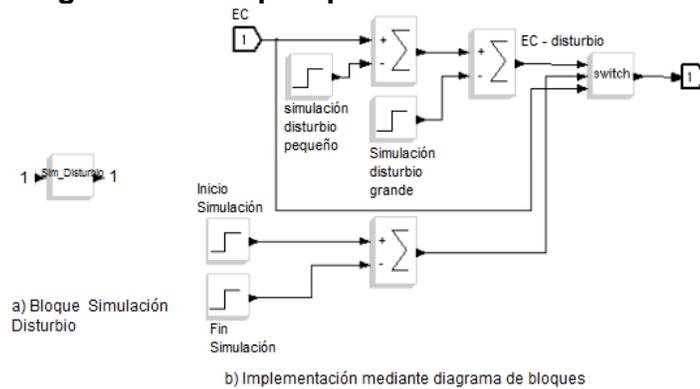
**Ilustración 5.13: Diagrama de bloques para la simulación de la versión prototipo del sistema de control regulatorio**



Fuente: propia

En el diagrama de bloques de la ilustración 5.13 también se utilizan otros bloques como el bloque *Sim\_Disturbio* cuya implementación se muestra en la ilustración 5.14. El disturbio se simula como una caída en el esfuerzo de control, para esta simulación se utilizan bloques tipo *step*. En la parte superior del diagrama de bloques, puede observarse un conmutador (*switch*), el cual habilita y deshabilita la presencia del disturbio dependiendo del valor de la suma de dos bloques *step*, ubicados en la parte inferior del diagrama, los cuales están programados para activarse y desactivarse en los mismos intervalos de tiempo establecidos en la sección 5.1.4. El bloque *Planta*, contiene la implementación del modelo POMTM.

**Ilustración 5.14: Diagrama de bloques para la simulación del disturbio.**

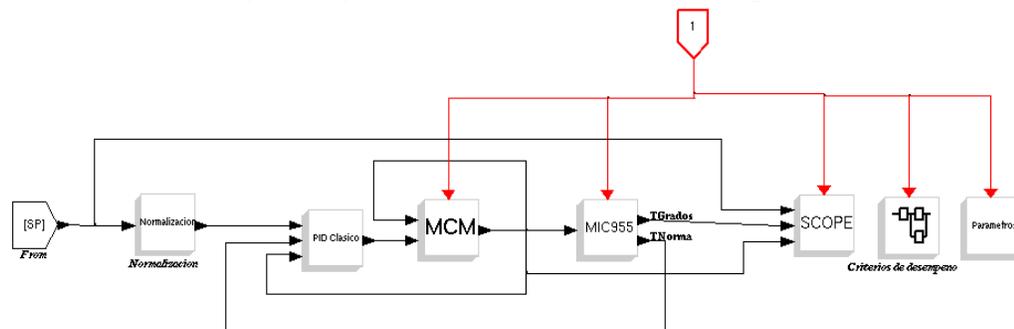


Fuente: propia

### 5.5.3.2 Diagramas de bloques utilizados para la generación de la tarea en tiempo real asociada a la versión prototipo del sistema de control regulatorio.

La ilustración 5.15 muestra el diagrama de bloques que se utiliza para generar la tarea en tiempo real asociada a la versión prototipo del sistema de control regulatorio. En este diagrama se usan la mayoría de bloques de la paleta RCP-MIC955 y el bloque *Scope* de la paleta RTAI-Lib.

**Ilustración 5.15: Diagrama de bloques para generar la tarea en tiempo real asociada a la versión prototipo del sistema de control regulatorio**



Fuente: propia

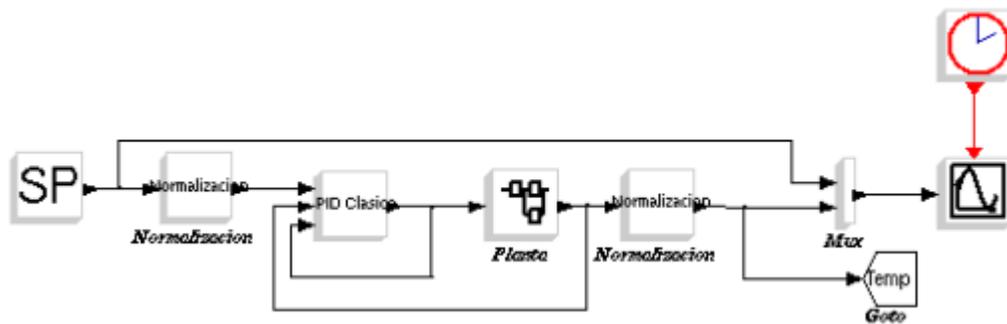
#### 5.5.4 Diagramas de bloques de la versión prototipo del sistema de control para el lazo servocontrolado

En esta sección se detallan los diagramas de bloques de la versión prototipo del sistema de control para el lazo servocontrolado, utilizados tanto en la fase de simulación como para la generación de la tarea de tiempo real en el sistema de RCP de temperatura para el módulo MIC955 de la empresa Feedback.

##### 5.5.4.1 Diagramas de bloques utilizados en la simulación de la versión prototipo del sistema servocontrolado

La ilustración 5.16 muestra el diagrama de bloques para la simulación de la versión prototipo del sistema servocontrolado, se utilizan los bloques *Normalización*, *MCM*, y el bloque *PID Clásico*. Es importante resaltar que este bloque puede ser fácilmente cambiado por el bloque *PID con Filtro Derivativo*.

**Ilustración 5.16: Diagrama de bloques para la simulación del lazo servocontrolado**

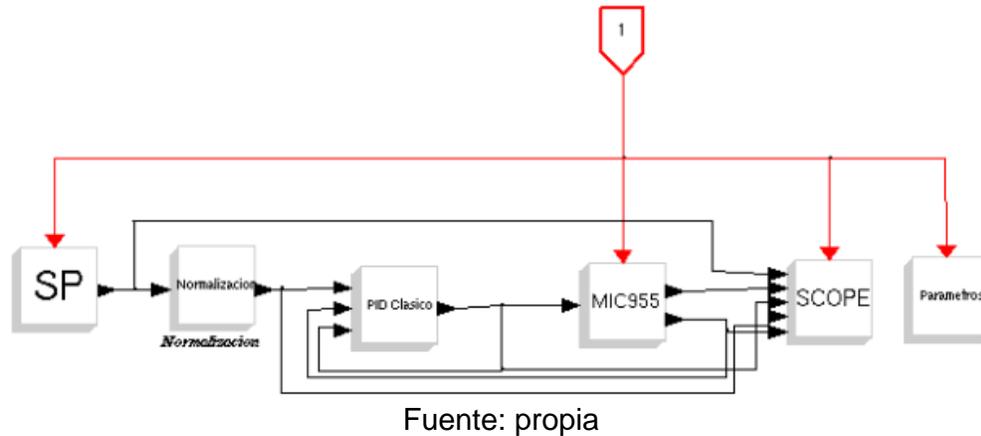


Fuente: propia

Para esta simulación no se utiliza el bloque *MCM*, pues en las pruebas operacionales no existe el cambio de modos de operación entre modos de control. El bloque *SP* contiene la programación de los cambios de *set point*, según los tiempos que se definieron en la sección 5.1.4.

La ilustración 5.17 muestra el diagrama de bloques que se utilizó para generar la tarea en tiempo real asociada a la versión prototipo del sistema de servocontrol. En este diagrama se usan la mayoría de bloques de la paleta RCP-MIC955 y el bloque *Scope* de la paleta RTAI-Lib.

**Ilustración 5.17: Diagrama de bloques para generar la tarea en tiempo real asociada al sistema servocontrolado**



## 5.6 RESULTADOS DE LAS PRUEBAS OPERACIONALES DE LOS PROTOTIPOS DE CONTROL

En esta sección se explican las gráficas de la respuesta de los prototipos de control, las cuales se obtuvieron durante el desarrollo de las pruebas operacionales. Para esta explicación se escogió el método *minimum ITAE* de Huang –Chaos, pues como se muestra en el anexo G fue el que consiguió la mejor calificación en los índices de desempeño. En el anexo G también se muestran las gráficas de respuesta con los demás métodos de sintonización que se escogieron en el presente caso de estudio y los resultados de la evaluación cuantitativa.

### 5.6.1 Pruebas operacionales en el lazo de control regulatorio

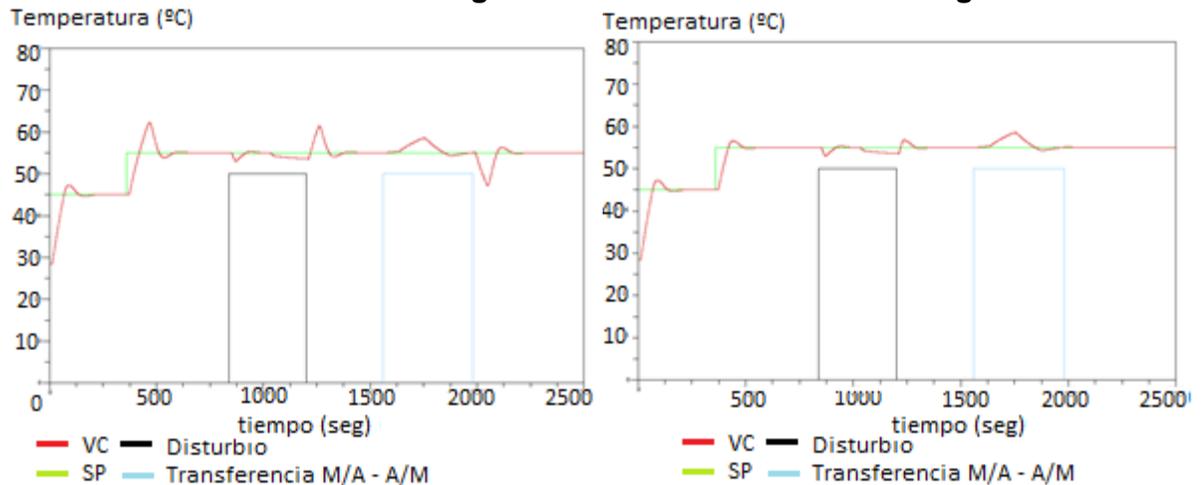
En la ilustración 5.18 se muestran los resultados obtenidos tanto en la fase de simulación como en la ejecución del sistema de control regulatorio, utilizando un controlador PID clásico sintonizado mediante el método *minimum ITAE* de Huang –Chaos.

La ilustración 5.18a y la ilustración 5.18b muestran la respuesta del sistema simulado, la traza verde corresponde al valor del SP, la traza roja a la VC, la traza negra a la presencia del disturbio y la traza azul celeste a la duración de la transferencia entre modos de control. La ilustración 5.18c y la ilustración 5.18d muestran la respuesta de la versión prototipo ejecutada en el sistema de RCP para el módulo MIC955. La traza verde corresponde al valor del SP, la traza roja a la VC, la traza blanca a la presencia del disturbio y la traza amarilla a la duración de la transferencia entre modos de control.

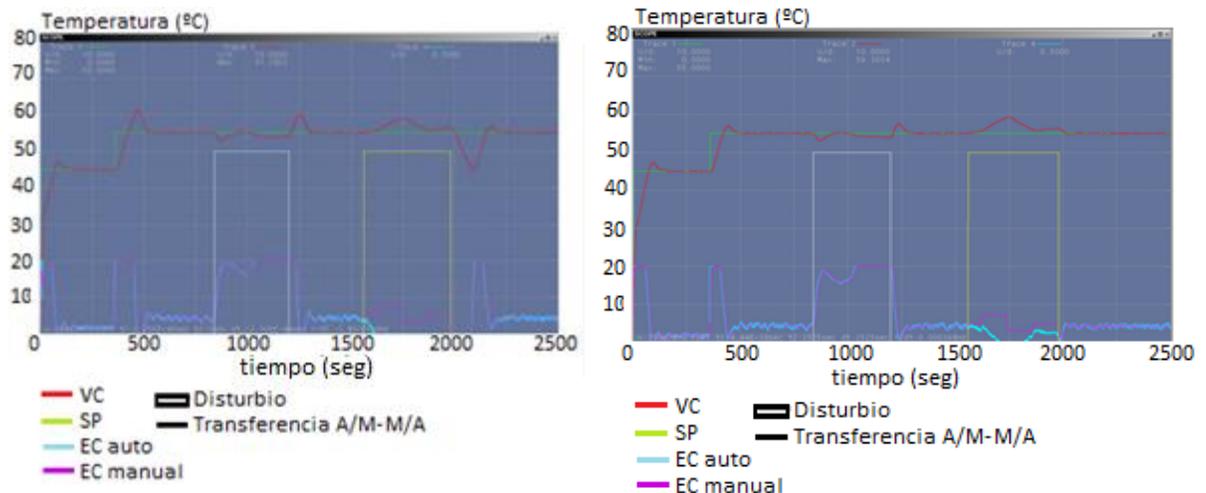
En la ilustración 18 se puede observar que no se presenta un gran sobre impulso cuando el controlador PID lleva la VC al SP de 45 °C, dado que el controlador PID está configurado en modo automático con protección AWBT activada. En el instante

$t=360$  segundos, se presenta un cambio de SP de 45 °C a 55 °C. En las ilustraciones 5.18a y 5.18c, se puede observar el gran sobre impulso que se presenta en la VC, esto se debe al área positiva que el término integral del controlador PID debe acumular y luego contrarrestar.

**Ilustración 5.18: Respuesta del controlador PID Clásico sintonizado mediante el método *mínimum* ITAE de Huang – Chaos en el lazo de control regulatorio**



- a) Simulación de la versión prototipo del sistema de control sin protección AWBT      b) Simulación de la versión prototipo del sistema de control con protección AWBT



- c) Respuesta de la versión prototipo del sistema de control en el sistema de RCP sin protección AWBT      d) Respuesta de la versión prototipo del sistema de control en el sistema de RCP con protección AWBT

Fuente: Propia

En las ilustraciones 5.18a y 5.18b se observa que el error en estado estacionario es igual a cero, a diferencia de las gráficas de la respuesta real (ver ilustraciones 5.18c y 5.18d), esto se debe a que en la simulación no se tienen en cuenta los disturbios externos que afectan al proceso, los cuales son difíciles de controlar, por ejemplo la variación de la temperatura ambiente y presencia de corrientes de aire fría.

En las ilustraciones 5.18b y 5.18d también se puede observar el efecto que provee la protección AWBT para el mismo cambio de SP en el instante  $t = 360$  segundos: El sobre impulso y las oscilaciones de la respuesta real del proceso son más pequeñas que las que se presentan cuando se desactiva la protección AWBT. Otra característica es que el sistema de control presenta un tiempo de establecimiento menor.

En el instante  $t=840$  segundos, se aplica un disturbio mediante la programación del encendido del ventilador. Este disturbio tiene una duración de 360 segundos, de los cuales 180 segundos corresponden a una magnitud pequeña en el disturbio y los restantes 180 segundos a una magnitud mayor. En la ilustración 5.18a y la ilustración 5.18c observa el gran sobre impulso que se presenta en la VC, cuando el disturbio deja de existir. En la ilustración 5.18c se muestra como el esfuerzo de control (traza violeta) llega al límite superior de saturación, sin embargo la magnitud del disturbio es tan grande que el valor de la VC comienza a disminuir. Durante este tiempo el término integral empieza a acumular esta área positiva. Cuando el disturbio desaparece, el EC estaba saturado en su límite superior debido a la acumulación del área positiva por parte de la componente integral, por esta razón la VC sobrepasa el valor del SP y continúa incrementado su valor: el sistema se comporta como un sistema en lazo abierto. Este fenómeno no lineal se conoce como efecto *windup* y se puede contrarrestar mediante la aplicación de técnicas *anti-windup* como se puede observar en la ilustración 5.18d.

En el instante  $t = 1560$  segundos se procede a cambiar de modo de operación de automático a manual. En la ilustración 5.18c se puede observar como el esfuerzo de control automático (traza azul) se satura en el límite inferior. Cuando el modo de operación vuelve a cambiar de manual a automático, se presenta una caída en el valor de la VC, debido a la saturación del EC automático. En la ilustración 5.18d se puede observar que el efecto de esta saturación es mínima, pues debido a la implementación de la técnica AWBT el esfuerzo de control automático intenta seguir las variaciones del EC manual (traza violeta).

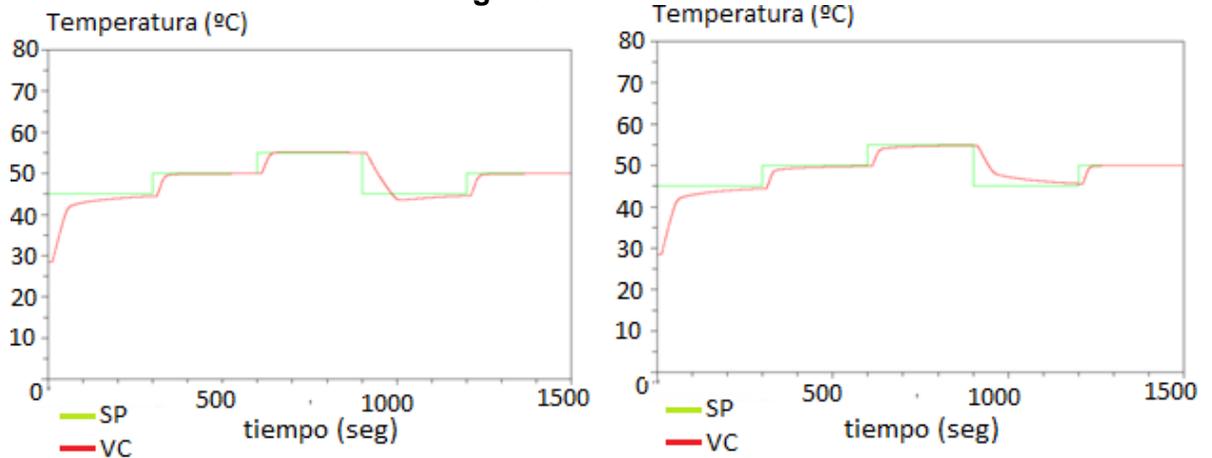
## **5.6.2 Pruebas operacionales en el lazo de servocontrol**

En la ilustración 5.19 se muestran los resultados obtenidos en la fase de simulación del sistema servocontrolado, utilizando un controlador PID clásico sintonizado mediante el método *mínimum* ITAE de Huang – Chaos.

Como se observa en la ilustración 5.19, aunque el seguimiento del SP con la protección AWBT activada es aceptable, es posible mejorarlo mediante un ajuste del

valor de  $T_i$ . En la ilustración 5.20, se muestran los resultados obtenidos tanto en la fase de simulación como en la ejecución del sistema servocontrolado, utilizando un controlador PID clásico sintonizado mediante el método *mínimum* ITAE de Huang – Chaos al cual se ha ajustado el valor del tiempo integral a  $\frac{T_i}{2}$ .

**Ilustración 5.19: Respuesta del controlador PID Clásico sintonizado mediante el método *mínimum* ITAE de Huang – Chaos en el lazo servocontrolado**



- a) Simulación de la versión prototipo del sistema de control sin protección AWBT      b) Simulación de la versión prototipo del sistema de control con protección AWBT

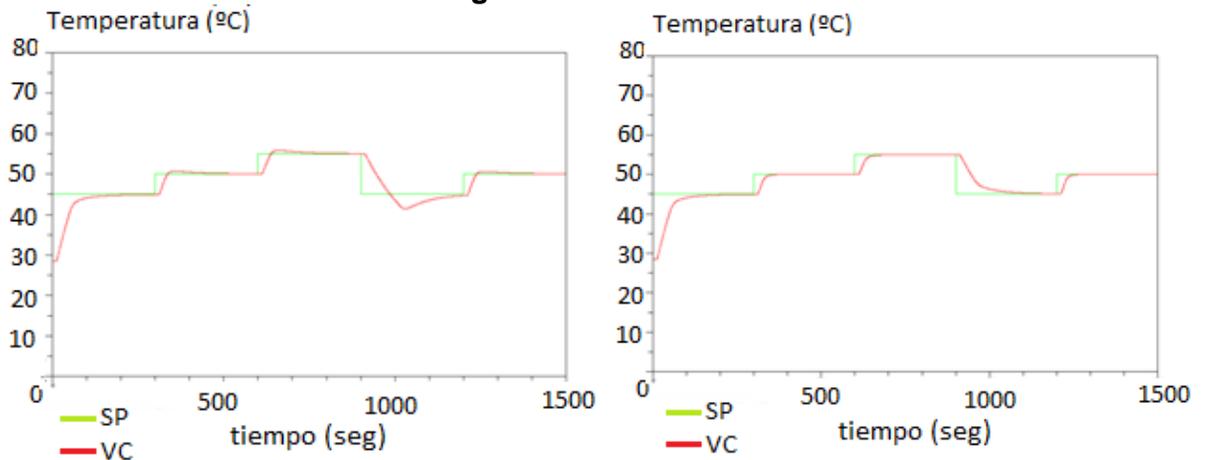
Fuente: Propia

La ilustración 5.20a y la ilustración 5.20b muestran la respuesta del sistema simulado, la traza verde corresponde al valor del SP el cual va cambiando de valor en intervalos de 300 segundos en la siguiente secuencia: 45 °C – 50 °C – 55°C – 45 °C – 50 °C. La traza roja corresponde a la VC. La ilustración 5.20c y la ilustración 5.20d muestran la respuesta de la versión prototipo ejecutada en el sistema de RCP para el módulo MIC955. La traza verde corresponde al valor del SP y la traza roja a la VC.

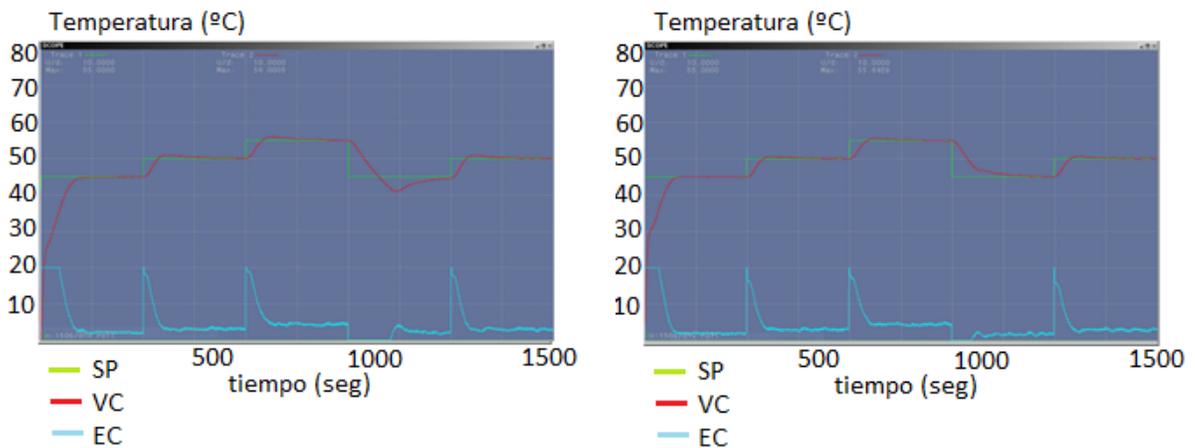
Se puede observar en la ilustración 5.20a y 5.20c que aunque existe un buen seguimiento para valores del SP por encima del punto de operación, el desempeño para valores por debajo del punto de operación no es aceptable. Sin embargo cuando se activa la protección AWBT el seguimiento es muy bueno.

Es importante recordar que esta respuesta aún puede mejorarse mediante una sintonía fina de los compontes P y D.

**Ilustración 5.20: Respuesta del controlador PID Clásico sintonizado mediante el método *mínimum* ITAE de Huang – Chaos en el lazo servocontrolado**



- a) Simulación de la versión prototipo del sistema de control sin protección AWBT      b) Simulación de la versión prototipo del sistema de control con protección AWBT



- c) Respuesta de la versión prototipo del sistema de control en el sistema de RCP sin protección AWBT      d) Respuesta de la versión prototipo del sistema de control en el sistema de RCP con protección AWBT

Fuente: Propia

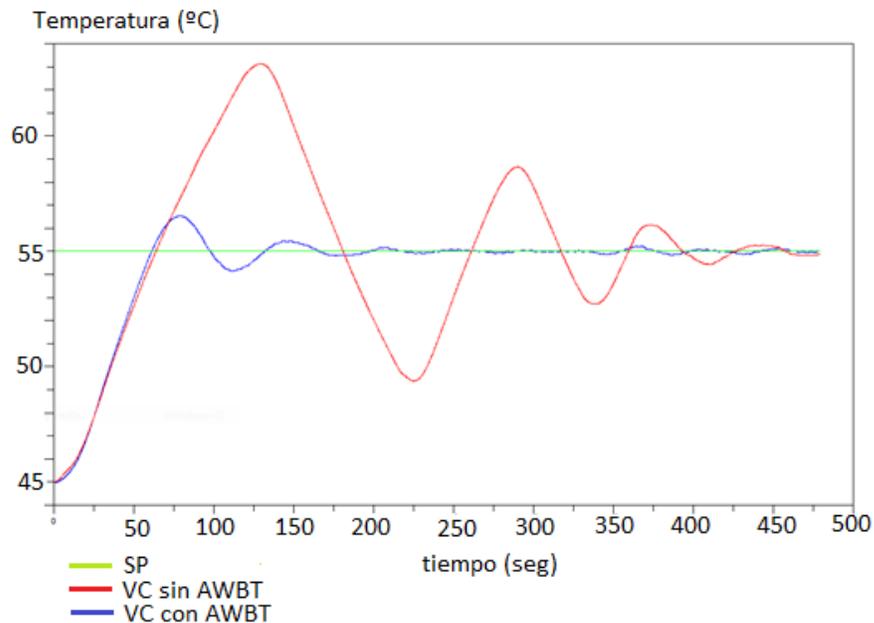
## 5.7 EVALUACIÓN MEDIANTE CRITERIOS DE DESEMPEÑO INTEGRALES Y TEMPORALES DEL COMPORTAMIENTO DEL SISTEMA DE CONTROL

En esta sección se muestran los resultados de la evaluación cuantitativa del desempeño de las versiones prototipo del sistema de control al utilizar el controlador PID con filtro derivativo y el controlador PID clásico.

### 5.7.1 Evaluación de la versión prototipo del sistema de control utilizando el control PID con filtro derivativo

La ilustración 5.21 muestra la respuesta del proceso en el lazo de control regulatorio, cuando se utiliza el controlador PID con Filtro Derivativo sintonizado mediante el método de Alfaro – Ruiz (ver cuadro 5.1). El valor del SP (traza verde) se establece en un valor de 55 °C. La traza roja corresponde a la respuesta del controlador sin protección AWBT y la traza azul corresponde a la respuesta del controlador con protección AWBT. Se puede observar las oscilaciones y el gran sobreimpulso que se presenta cuando no se activa la protección AWBT.

**Ilustración 5.21: Respuesta del Controlador PID con Filtro Derivativo sintonizado con el método minimun IAE de Alfaro Ruiz en el lazo de control regulatorio**



Fuente: propia

La tabla 5.1 muestra los resultados de la evaluación del desempeño del controlador PID con filtro derivativo sintonizado con el método minimun IAE de Alfaro – Ruiz. Esta evaluación se lleva a cabo mediante criterios temporales como el tiempo de subida (TR), el tiempo de establecimiento (TS), el máximo sobreimpulso (MP) y criterios integrales (ver sección 4.2.2) como la integral del error absoluto (IAE), la integral del cuadrado del error (ISE), la integral del error absoluto multiplicado por el tiempo (ITAE) y la integral del cuadrado del error multiplicada por el tiempo (ITSE).

**Tabla 5.1: Evaluación del desempeño del controlador PID con filtro derivativo sintonizado con el método *minimun* IAE de Alfaro – Ruiz en el lazo de control regulatorio.**

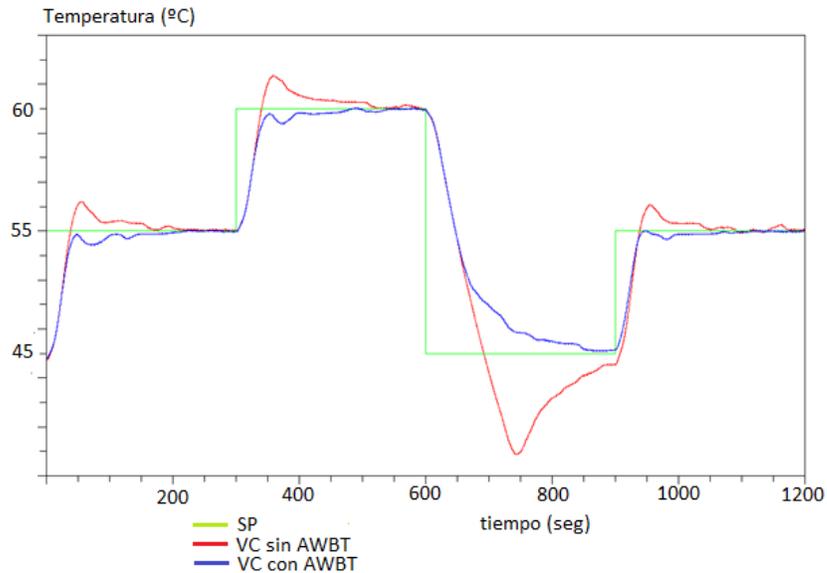
CONTROLADOR	Controlador PID con Filtro Derivativo			MODO DE OPERACIÓN			Control Regulatorio	
	AWBT	TR	TS(2%)	MP	IAE	ISE	ITAE	ITSE
Minimun IAE – Alfaro Ruiz (2005 <sup>a</sup> )	NO	63.76 seg	377 seg	14.79 %	686.2	183.6	101907	20549
Minimun IAE – Alfaro Ruiz (2005 <sup>a</sup> )	SI	61.16 seg	87.7 seg	2.77 %	216.8	67.81	9898.9	1217.8

Como se puede observar en la tabla 5.1, cuando se activa la protección AWBT los resultados son mejores. Una de las conclusiones que se pueden obtener de la tabla 5.1, es que no se necesita hacer sintonía fina si se activa la protección AWBT pues el tiempo de establecimiento y el máximo sobre impulso (MP) son pequeños. Los valores de los criterios de desempeño integral también califican de manera positiva la activación de la protección AWBT, pues cuando está se encuentra activada estos valores tienden a disminuir.

La ilustración 5.22 muestra la respuesta del proceso en el lazo de servocontrol, cuando se utiliza el controlador PID con Filtro Derivativo sintonizado mediante el método de Alfaro – Ruiz (ver cuadro 5.2), al cual se ha hecho un ajuste del valor de  $T_i$  igual al que se propuso en la sección 5.6. El valor del SP (traza verde) cambia en el tiempo en intervalos de 300 segundos, de acuerdo a la siguiente secuencia: 50 °C – 55 °C – 45 °C – 50 °C. La traza roja corresponde a la respuesta del controlador sin protección AWBT y la traza azul corresponde a la respuesta del controlador con protección AWBT. Se puede observar que se presentan sobre impulsos cuando no se activa la protección AWBT

La tabla 5.2 muestra los resultados de la evaluación del desempeño del controlador PID con filtro derivativo sintonizado con el método *minimun* IAE de Alfaro – Ruiz. Esta evaluación se lleva a cabo mediante criterios integrales (ver sección 4.2.2). como la integral del error absoluto (IAE), la integral del cuadrado del error (ISE), la integral del error absoluto multiplicado por el tiempo (ITAE) y la integral del cuadrado del error multiplicada por el tiempo (ITSE)

**Ilustración 5.22: Respuesta del Controlador PID con Filtro Derivativo sintonizado con el método mínimum IAE de Alfaro Ruiz en el lazo de servocontrol**



Fuente: propia

**Tabla 5.2: Evaluación del desempeño del controlador PID con filtro derivativo, sintonizado con el método mínimum IAE de Alfaro – Ruiz para el lazo de servocontrol**

CONTROLADOR	Controlador PID con filtro derivativo		MODO DE OPERACIÓN		Servocontrol
	AWBT	IAE	ISE	ITAE	ITSE
Mínimum IAE – Alfaro Ruiz (2005 <sup>a</sup> )	NO	731.91	150.58	431056.72	88769.86
Mínimum IAE – Alfaro Ruiz (2005 <sup>a</sup> )	SI	559.38	127.47	307092.27	71088.729

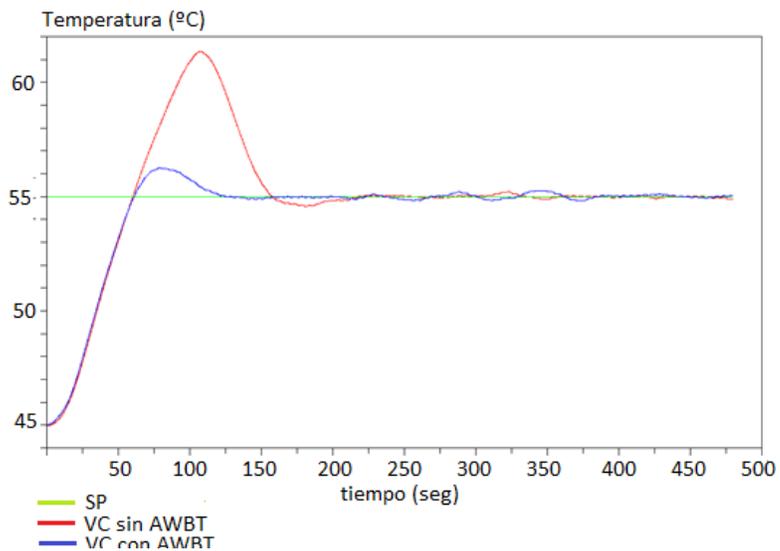
Como se puede observar en la tabla 5.2, cuando se activa la protección AWBT los resultados son mejores que sin la protección AWBT, sin embargo aunque ya se hizo un ajuste del valor del componente integral es posible mejorar la respuesta del controlador PID haciendo una sintonía fina en los componentes P y D.

### 5.7.2 Evaluación de la versión prototipo del sistema de control utilizando el control PID Clásico

La ilustración 5.23 muestra la respuesta del proceso en el lazo de control regulatorio, cuando se utiliza el controlador PID Clásico sintonizado mediante el método *mínimum* ITAE de Huang - Chaos, 1982 (ver cuadro 5.2). El valor del SP (traza verde) se establece en un valor de 55 °C. La traza roja corresponde a la respuesta del

controlador sin protección AWBT y la traza azul corresponde a la respuesta del controlador con protección AWBT. Se puede observar el gran sobreimpulso que se presenta cuando no se activa la protección AWBT

**Ilustración 5.23: Respuesta del Controlador PID Clásico sintonizado con el método minimun ITAE de Huang – Chaos, 1982, en el lazo de control regulatorio**



Fuente: propia

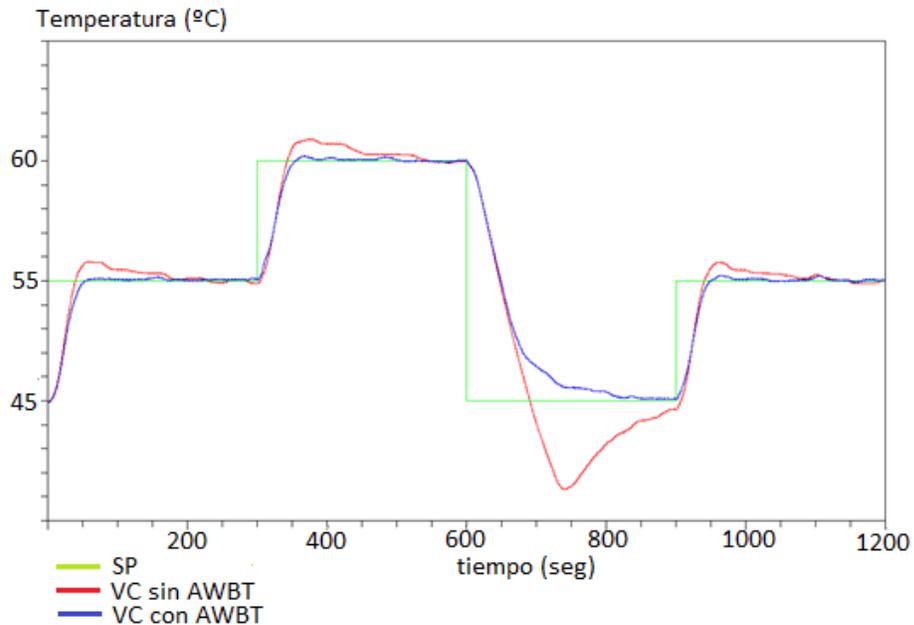
La tabla 5.3 muestra los resultados de la evaluación del desempeño del controlador PID clásico sintonizado con el método minimun ITAE de Huang - Chaos. Esta evaluación se lleva a cabo mediante criterios temporales como el tiempo de subida (TR), el tiempo de establecimiento (TS), el máximo sobreimpulso (MP) y criterios integrales (ver sección 4.2.2) como la integral del error absoluto (IAE), la integral del cuadrado del error (ISE), la integral del error absoluto multiplicado por el tiempo (ITAE) y la integral del cuadrado del error multiplicada por el tiempo (ITSE).

**Tabla 5.3: Evaluación del desempeño del controlador PID clásico, sintonizado con el método minimun ITAE de Huang – Chaos para el lazo de control regulatorio.**

CONTROLADOR	Controlador Clásico			MODO DE OPERACIÓN			Control Regulatorio	
	AWBT	TR (seg)	TS (seg)	MP	IAE	ISE	ITAE	ITSE
Minimun ITAE – Huang and Chao (1982)	NO	60.1	145.5	11.53	355.77	106.15	24650.	5265.3
Minimun ITAE – Huang and Chao (1982)	SI	61.0	90.4	2.27	207.02	65.90	9356.2	1143.2

La ilustración 5.24 muestra la respuesta del proceso en el lazo de servocontrol, cuando se utiliza el controlador PID clásico sintonizado mediante el método *minimum ITAE* de Huang and Chanos (ver cuadro 5.2), al cual se ha hecho un ajuste del valor de  $T_i$  igual al que se propuso en la sección 5.6. El valor del SP (traza verde) cambia en el tiempo en intervalos de 300 segundos, de acuerdo a la siguiente secuencia: 50 °C – 55 °C – 45 °C – 50 °C. La traza roja corresponde a la respuesta del controlador sin protección AWBT y la traza azul corresponde a la respuesta del controlador con protección AWBT. Se puede observar que se presentan sobre impulsos cuando no se activa la protección AWBT.

**Ilustración 5.24: Respuesta del Controlador PID Clásico sintonizado con el método *minimum IAE* de Huang – Chaos, -en el lazo de servocontrol**



Fuente: propia

La tabla 5.4 muestra los resultados de la evaluación del desempeño del controlador PID clásico sintonizado con el método *minimum ITAE* de Huang and Chaos. Esta evaluación se lleva a cabo mediante criterios integrales (ver sección 4.2.2) como la integral del error absoluto (IAE), la integral del cuadrado del error (ISE), la integral del error absoluto multiplicado por el tiempo (ITAE) y la integral del cuadrado del error multiplicada por el tiempo (ITSE).

**Tabla 5.4: Evaluación del desempeño del controlador PID Clásico, sintonizado con el método minimun ITAE de Huang – Chaos el lazo de servocontrol**

<b>CONTROLADOR</b>	Controlador PID Clásico		<b>MODO DE OPERACIÓN</b>		Servocontrol
	<b>AWBT</b>	<b>IAE</b>	<b>ISE</b>	<b>ITAE</b>	<b>ITSE</b>
Minimun ITAE – Huang and Chaos (1982)	NO	724.68	147.72	428472.91	87096.028
Minimun ITAE – Huang and Chaos (1982)	SI	517.42	126.97	287012.45	71283.204

Como se puede observar en la tabla 5.2, cuando se activa la protección AWBT los resultados son mejores que sin la protección AWBT, sin embargo aunque ya se hizo un ajuste del valor del componente integral es posible mejorar la respuesta del controlador PID haciendo una sintonía fina en los componentes P y D.

## **5.8 RESUMEN**

En este capítulo se presentó un procedimiento para realizar RCP en el módulo MIC955. Se detallaron todos los pasos que hacen parte de este procedimiento y se analizaron los resultados de las pruebas operacionales de los prototipos de control. En un principio se hizo una descripción cualitativa de las respuestas que se obtuvieron tanto en la fase de simulación como en la de ejecución de la tarea de tiempo real asociada al sistema de control. Luego se procedió a evaluar el comportamiento del sistema de control mediante criterios de desempeño integrales y temporales. Todos los resultados se consignaron en tablas de fácil lectura para su posterior análisis

## 6 CONCLUSIONES

Se instaló el sistema operativo GNU Linux Fedora 7 en el PC, esta instalación se realizó de manera gráfica, por lo tanto no hubo mayores inconvenientes. Una vez la instalación del sistema operativo terminó de manera satisfactoria, se procedió a instalar RTAI, el cual permite agregar funcionalidades de tiempo real al sistema operativo GNU Linux. Se comprobó que la instalación de RTAI no es complicada, sin embargo, es necesario tener conocimientos básicos de administración de sistemas GNU Linux.

Con el sistema operativo funcionando en tiempo real, se procedió a instalar la cadena de herramientas de RTAI-Lab y el software de diseño de sistemas de control asistido por computador, Scilab/Scicos. Se comprobó que para usar adecuadamente estas herramientas se requiere que el PC tenga capacidades de procesamiento altas.

Se diseñó e implementó un medio de comunicación versátil y dinámico, entre el módulo MIC955 de la empresa Feedback y RTAI-Lab. Este medio de comunicación hace uso del puerto paralelo del PC y permite el envío y la recepción de señales adecuadas, desde y hacia los diferentes componentes de la planta. Se concluye que el puerto paralelo del PC cumple los requerimientos para realizar tareas de control de tiempo real, lo cual permite implementar lazos de control de manera rápida y económica.

Se desarrolló un conjunto de bloques en Scicos, que permiten el diseño de prototipos de sistemas de control de manera rápida. Estos bloques se integraron en una paleta que se adicionó al Software CACSD Scilab/Scicos. Entre los bloques se encuentran los controladores tipo PID y los bloques que permiten la comunicación vía software entre el módulo MIC955 y el PC. Para crear bloques personalizados es necesario tener conocimientos de programación en lenguaje C o en lenguaje Fortran y del lenguaje nativo de Scilab/Scicos.

Las herramientas mencionadas anteriormente constituyen el Sistema de Prototipado Rápido de Control para el módulo MIC955, con el cual es posible realizar diferentes prácticas de control de temperatura y probar prototipos de control. En el presente trabajo de grado, se implementaron sistemas de control regulatorio y servocontrolados.

Se comprobó que mediante la identificación de modelos POMTM asociados al proceso, es posible sintonización correctamente los controladores PID, sin embargo,

cuando se utilizó estos prototipos de control en el proceso real, se pudo constatar que la adición de técnicas AWBT mejora notablemente la respuesta del sistema de control.

En cuanto a la identificación de modelos, el sistema de RCP, es muy versátil y permite realizar diferentes procedimientos de identificación.

Finalmente, se crearon nuevas guías de práctica, entre las cuales se incluyen control ON/OFF, control ON/OFF con histéresis, Control P, control PI, control PID, prácticas de sintonización de controladores PID y prácticas de control PID AWBT. El sistema de RCP permite la visualización de los valores de los modos de control PID, pudiendo observar cómo interactúan los modos de control en el tiempo.

## 7 BIBLIOGRAFIA

ACOSTA, Francisco. RTAI – Real Time Application Interface. Universidad Católica Nuestra Señora De La Asunción. Teoría y Aplicación de la Informática 2– T.A.I 2, 2009

ALFARO, Víctor. Métodos de sintonización de controladores PID que operan como reguladores. Artículo de divulgación, Universidad de Costa Rica, San José, Costa Rica, 2002b, p. 21-36.

ALFARO, Víctor. Métodos de sintonización de controladores PID que operan como servomecanismos. Artículo de divulgación, Universidad de Costa Rica, San José, Costa Rica, 2003, p. 13-29.

ALFARO, Víctor M. Identificación de procesos Sobreamortiguados Utilizando Técnicas De Lazo Abierto, proyecto N° 731-A0-169, San José, Costa Rica, 2001.

ALFARO, Víctor M. Identificación de modelos de orden reducido a partir de la curva de reacción del proceso, Ciencia y Tecnología, 24(2): 197-216. 2006

ÅSTRÖM, Karl and HÄGGLUND, Tore. Advanced PID Control. ISA, 2006. 456 p.

ÅSTRÖM, Karl and WITTENMARK, Björn. Computer Controlled Systems: Theory and Design. Prentice-Hall, 1984.

BUCHER, R.; MANNORI, S, NETTER, T. RTAI-Lab tutorial: Scilab, Comedi, and real-time control. Politecnico di Milano - Dipartimento di Ingegneria Aerospaziale, 2010.

BUCHER, R, DOZIO, L. CACSD under RTAI Linux with RTAI-LAB. Realtime Linux Work-shop, Valencia, España, 2003.

CAMPBELL, S., CHANCELIER , J.P.; NIKOUKHAH, R. Modeling and Simulation in Scilab/Scicos. Springer Science + Business Media, Inc. 2006.

CAPO i Capó, Tomeu, Programación del Puerto Paralelo con Linux I 2002 [En línea] Disponible en: <http://bulma.net/body.phtml?nIdNoticia=1150> [Consultado en 20 de Noviembre del 2010]

CHEN, X.M., GONG, X.L., ZHOU, H.X., XU, Z.B., XU, Y.G., y KANG, C.J., An Economical Rapid Control Prototyping System Design with Matlab/Simulink and

TMS320F2812 DSP, The International MultiConference of Engineers and Computer Scientists 2010 Voll IMECS, Hong Kong, 2010

CHEN, Chi-Tsong. Analog and Digital Control System Design: Transfer-Function, State-Space, and Algebraic Methods, 1 ed. Saunders College Publishing , 1993. 600 p.

CHENG, Albert M. K., REAL-TIME SYSTEMS Scheduling, Analysis, and Verification, John Wiley & Sons, Inc., Hoboken, New Jersey, 2002

COHEN, I; HANUS, R; VANBERGEN, J.P. The impact of the control structure on the closed loop efficiency. Journal A, 26(1), p. 18-25, 1985

CONCURRENT Technology Partners - Concurrent Computer Corporation, Concurrent Technology Partners, 2009. [En línea] Disponible en: <http://www.ccur.com/> [Consultado en 7 de Febrero del 2011].

COMEDI, Linux control and measurement device interface, Comedi, 2008 [En línea] Disponible en: <http://www.comedi.org/index.html> [Consultado en 20 de Febrero del 2011].

CREUS, Antonio. Instrumentación Industrial, 6 ed, Alfaomega Grupo Editor, 1997, 741 p.

DE LA PUENTE, Juan Antonio, Introducción a los Sistemas de Tiempo Real, DIP, UPM, 2007.

DEBELLE, J. A control structure based upon process models. Journal A, 20(2): p. 71-81, 1979.

DIRK, A; ULRICH E.; GERD-Ulrich S. Integration von Advanced Control in der Prozessindustrie. WILEY-VCH Verlag GmbH & Co. KGaA, 2008.

DSPACE GmbH, dSPACE International, 2011 [En línea] Disponible en: <http://www.dspaceinc.com/> [Consultado en 7 de Febrero del 2011].

DORF, Richard and BISHOP, Robert. Modern Control Systems, 11 ed. Pearson Prentice Hall, 2008. 1056 p

DUMA, R.; DOBRA, P.; ABRUDEAN, M; DOBRA, M. Rapid Prototyping of Control Systems using Embedded Target for TI C2000 DSP. Mediterranean Conference on Control and Automation, Athens, Greece, 2007.

ERKKINEN, G.; KLUZA, T.; SHENOY, R. Embedded realtime control via MATLAB, Simulink, and xPC Target. In D Hristu-Varsakelis and W.S. Levine, editors, Handbook in Networked and Embedded Systems, Birkhäuser, Boston, MA, 2005.

ETI Tudela, EL PUERTO PARALELO Departamento de Electricidad, Módulo: Comunicaciones Industriales 2011 [En línea] Disponible en: <http://www.etitudela.com/fpm/comind/index.html> [Consultado en 15 de Enero del 2011]

FEEDBACK Instrumenst Limited, Módulo para aplicación a Microordenadores MIC 955 para Control de Temperatura, Manual Sp955, F I Ltd Crowborough Sussex, Inglaterra, 1983.

FEEDBACK plc, The History of Feedback, 2010 [En línea] Disponible en: <http://www.feedback-group.com/articles/feedback-history> [Consultado en 7 de Marzo del 2011]

FUNDACIÓN LUDOVICO RUTTEN, Manual de Armado y Reparación de PC, Curso de Armado de Computadores y redes, Chile, 2010.

GALLEGO León, José. Técnicas de Programación - Ciclo formativo: Grado superior, McGraw-Hill/Interamericana de España, S.A., 1998

GEC PLESSEY Semiconductors, ZN427E8 / ZN427J8 Microprocessor Compatible 8-Bit Successive Approximation A-D Converter, DS3006 - 2.1, 1994.

D. HERCOG y K. JEZERNIK, Rapid Control Prototyping MATLAB/Simulink and a DSP-Based Motor controller, Int. J. Eng. Ed., vol. 21, 2005.

HERRERA, Juan Carlos. Interfaces para el puerto paralelo de la PC, en modo bidireccional. Centro de Innovación y Desarrollo Tecnológico en Cómputo CIDETEC IPN, México, 2009

HÖLTTÄ V.; PALMROTH, L.; ERIKSSON L. Rapid control prototyping tutorial with application examples. Sim-Serv, 2004.

JOHNSON, Michael and MORADI, Mohammad. PID Control - New Identification and Design Methods. Springer-Verlag London Limited, 2005. 539 p.

KAMAL, Raj. Embedded Systems. Architecture, Programming and Design, 2 ed. McGraw-Hill, 2008.

KOPETZ, Hermann. REAL-TIME SYSTEMS Design Principles for Distributed Embedded Applications. Kluwer Academic Publishers, 2002

HONG K; et al. An Integrated Environment for Rapid Prototyping of DSP Algorithms using Texas Instruments TMS320C30, Microprocessors and Microsystems, vol. 24,. 2000.

KOTHARE et al., A Unified Framework for the Study of Anti-windup Designs, Automatica, Vol. 30, No. 12, p 1869-1883, 1994

KUO, Benjamin. Sistemas de Control Automático, 7 ed. México, DF: Prentice-Hall Hispanoamericana SA, 1996. 863 p.

LIPTÁK, Bela. Process Control and Optimization, vol. II, 4 ed. CRC Press - ISA Press, 2006. 2302 p.

MCMILLAN, Gregory and CONSIDINE, Douglas. Process/Industrial Instruments and Control Handbook, 5 ed. New York: McGraw-Hill, 1999.

NATIONAL INSTRUMENTS Corporation. Developing a Rapid Control Prototyping Platform for Implementing Advanced Engine Control Systems 2011. Disponible en: <http://www.ni.com/rcp/esa/> [Consultado en 20 de Febrero del 2011].

NETTER Thomas. Scicos: Block diagram modeler/simulator, INRIA, 2009 [En línea] Disponible en: <http://www-rocq.inria.fr/scicos/> [Consultado en 20 de Febrero del 2011].

O'DWYER, Aidan. Handbook of PI and PID Controller Tuning Rules, 3 ed. Londres: Imperial College Press, 2009. 599 p.

OPAL-RT Corporation. RT- LAB, RT-LAB Professional, 2009 [En línea] Disponible en: <http://www.rt-lab.com/> [Consultado en 7 de Febrero del 2011].

PADILLA, P. et al. Métodos de Identificación dinámica. Facultad de Ingeniería Eléctrica, Universidad de Oriente, 2008.

PRESICIONMBA, LLC, "Rapid Control Prototyping," precisionmba.com, 2005 [En línea]. Disponible en: [http://www.precisionmba.com/rapid\\_control\\_prototyping.htm](http://www.precisionmba.com/rapid_control_prototyping.htm) [Consultado 5 de Febrero del 2011].

RODRÍGUEZ, A. Sistemas SCADA – 2º edición. Aquilino Computación Marcombo Solid Works, 2007

SEBORG, D; EDGAR, T; MELLICHAMP, D. Process Dynamics and Control, 2 ed. New York: Jhon Wiley & Sons, Inc, 2004. 610 p.

SKIBA G.; ŻABIŃSKI, T., BOŹEK A. Rapid Control Prototyping with Scilab/Scicos/RTAI for PC-based and ARM-based Platforms. International Multiconference on Computer Science and Information Technology, Wisla, Poland, 2006.

SMITH, Carlos. Automated Continuous Process Control. New York: Jhon Wiley & Sons, Inc, 2002. 215 p.

SMITH, Carlos and CORRIPIO, Armando. Principles and Practice of Automatic Process Control, 2 ed. John Wiley & Sons, 1997. 763 p.

SVRCEK, W; MAHONEY D; YOUNG, B. A Real-Time Approach to Process Control, 2 ed. West Sussex: Jhon Wiley & Son, 2006. 323 p.

The MATHWORKS, Inc. xPC Target 4.3, Perform real-time rapid control prototyping and hardware-in-the-loop simulation, 2010. [En línea] Disponible en: <http://www.mathworks.com/products/xpctarget/> [Consultado en 20 de Febrero del 2011].

THE RTAI DEVELOPMENT TEAM, DIAPM RTAI-Beginner's Guide, Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, Real Time Application Interface, 2006. [En línea] Disponible en internet en: <https://www.rtai.org/> [Consultado 10 de Enero del 2011].

The SCILAB Consortium, Scilab, Open Source Involvement, 2011 [En línea] Disponible en: <http://www.scilab.org/> [Consultado en 20 de Febrero del 2011].

TIMESYS Corporation. The Concise Handbook Of Linux for Embedded Real-Time Systems. TimeSys Corporation, Pittsburgh, PA, Version 1.1, 2002.

VISIOLI, Antonio. Practical PID Control (Advances in Industrial Control). Londres: Springer-Verlag London Limited, 2006. 309 p.

WADE, Harold. Basic and Advanced Regulatory Control: System Design and Application, 2 ed. ISA, 2004. 365 p.

WOOTAIK L.; MINSUK, S.; MYOUNGHO S. Target-Identical Rapid Control Prototyping Platform for Model-Based Engine Control, J. Automobile Engineering, vol. 218, 2004.

Yun Li et al. PID Control Systems Analysis and Design. IEEE Control Systems Magazine. Feb 2006, p. 32-41.

ZHEN, A; KOTHARE, M; MORARI, M. Anti-windup design for internal mode control", International Journal of Control, Vol. 60, No. 5, p. 1015-1024, 1994.

ZIEGLER, J.G and NICHOLS, N.B. Optimum settings for automatic controllers. Trans. ASME. Vol. 64, no 8, 1942. p. 759 - 768.