APUNTE DE PRACTICAS DE LABORATORIO DE CONTROL

Dr. Yu Tang
Dr. Romeo Ortega
Dra. Cristina Verde
M en I Luis Alvarez
M en I Gerardo Espinosa

D - 95

F-DEPFI
D-95
1991
E.6

I N D I C E

Indice		1
Prólogo		3
Práctica 1	Sistema térmico	5
Práctica 2	Servomotor de Corriente Directa	13
Práctica 3	Servomotor de Corriente Alterna	23
Práctica 4	Computadora Analógica	31
Práctica 5	Tarjeta de adquisición Lab-Master	41
Práctica 6	Controladores PID discretos	55
Práctica 7	Control Digital del Proceso Térmico por Ubicación de Polos	69
Práctica 8	Estabilización de una Planta Inestable con un Control de Estructura Variable	79
Práctica 9	Control de Posición de un Motor de CD a través de un Controlador de Estructura Variable	91
Práctica 10	Diseño Asistido por Computadora	107
Práctica 11	Introducción al robot manipulador	119
Práctica 12	Programación del robot con "Teach Pendant"	133
Práctica 13	Programacion del robot a traves de BASIC	149
Anexo 1	Resumen de comandos de CC	163
Anexo 2	Resumen de comandos de SIMNON	173
Anexo 3	Resumen de comandos de MATLAB	179
Anexo 4	Listado de la unidad LABCOM	183

G(2)

P R O L O G O

La automatización y la robótica ocupan un lugar cada vez más importante en la modernización industrial. El sistema de control por computadora, por su mejor desempeño, bajo costo de operación y mantenimiento, ha ganado una gran aceptación en la industria. La robótica, por otro lado, juega un papel trascendente en la realización de automatización de muchas industrias de procesos y de fabricación.

El objetivo de este apunte es, a través de una serie de prácticas, introducir al lector a la teoría básica de control por computadora, la programación de un robot de laboratorio Rhino, y el manejo de los elementos básicos involucrados en ello. El apunte consta de trece prácticas que consisten cada una, en una breve introducción teórica, el método de diseño y el desarrollo de la práctica. Se diseñaron las prácticas con el fin de apoyar los cursos de control dados en la División de Estudios de Posgrado (DEPFI), la División de Educación Continua (DECFI) y la Facultad de Ingeniería (FI).

Este apunte fue desarrollado en la DEPFI por los profesores Dr. Tang Yu, Dr. Romeo Ortega, Dra. Cristina Verde, M.I. Luis Alvarez Icaza, M.C. Gerardo Espinoza.

Las prácticas que aquí se presentan fueron efectuadas por los alumnos Gerardo Escobar, Rubén Zazueta, Francisco García, Enrique Meza, Job Flores y Alejandro Pérez en las instalaciones del Laboratorio de Control Automático de la DEPFI.

Marzo de 1991
Ciudad Universitaria

PRACTICA 1

SISTEMA TERMICO

I. OBJETIVOS

- Familiarizar al estudiante con el uso y funcionamiento del sistema térmico PT326
- Implementar un sistema de control de temperatura en lazo cerrado

II. MATERIAL Y EQUIPO

- 1 Proceso térmico PT326
- 1 Osciloscopio de dos canales

III. INTRODUCCION

El proceso térmico PT326 fabricado por FEEDBACK es un proceso auto-contenido. En él, el aire tomado de la atmósfera por un ventilador centrífugo es pasado por una malla calentadora y un tubo para regresarlo nuevamente a la atmósfera.

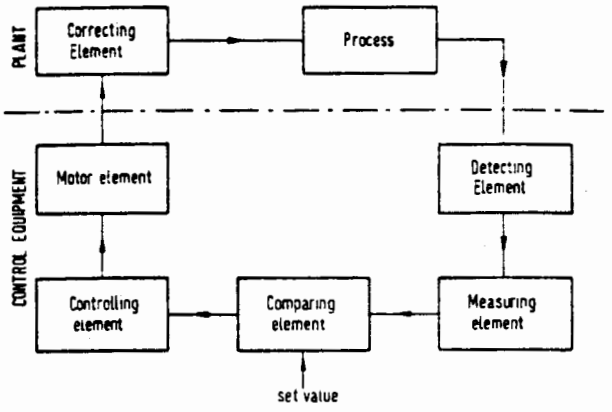


Fig. 1 Diagrama a bloques del sistema PT326.

El proceso consiste en calentar el aire que fluye por el tubo, medir la temperatura del aire y compararla con un valor de referencia dado. El objetivo de control es generar una señal de control la cual determina la

cantidad de energía eléctrica suministrada a un elemento corrector para mantener la temperatura del aire a un nivel deseado. El diagrama a bloques básico de este sistema se muestra en la figura 1.

a. Descripción del equipo

En esta sección se presentan las características principales del sistema PT326.

FUENTES

El equipo opera desde un suministro principal de 220/250V o 100/120V. El interruptor principal, la lámpara indicadora de encendido, los fusibles de 2A y 100mA están montados sobre el lado izquierdo del aparato.

PANEL FRONTAL

Los elementos que conforman al sistema son mostrados en diagrama de bloques sobre este panel. Existe una sobre-tapa que puede ser utilizada para dejar libre solamente el control de ganancia proporcional.

PROCESO

Este término generalmente se emplea para describir un cambio físico o químico, la conversión de energía, velocidad de un fluido, velocidad a la cual sucede una reacción química, etc. En nuestro caso, la temperatura del aire que fluye en el tubo del proceso es elevada a un valor deseado dentro de un rango que abarca la temperatura ambiente y 60°C.

ELEMENTO DETECTOR

Se trata de un termistor dentro de una punta de prueba que puede ser insertado en el flujo de aire en una de tres posibles posiciones a lo largo del tubo que están a distancias de 28mm, 140mm y 279mm del calentador.

ELEMENTO DE MEDICION

El termistor forma parte de un puente de DC que es balanceado a 40°C. La salida del puente es aplicada a un amplificador y produce un voltaje que varía desde 0 hasta +10V para producir una variación de temperatura de 30 hasta 60°C. La salida del elemento de medición puede ser monitoreada desde el conector Y del panel frontal.

VALOR MEDIDO θ_o

Este es el valor de salida del elemento de medición correspondiente al de la señal de control aplicada.

VALOR DE REFERENCIA θ_i

Es el valor requerido al que se desea llevar la temperatura del proceso. Se puede usar un control interno para elevarlo hasta 60°C. También, dicho valor puede ser controlado externamente al emplear un voltaje de 0 a -10V en el conector D del panel frontal.

DESVIACION θ

Es la diferencia entre θ_o y θ_i.

VALOR DE PERTURBACION

Cuando se opera el interruptor marcado como INTERNAL SET VALUE DISTURBANCE, se ejerce internamente un cambio tipo escalón en la señal de referencia.

ELEMENTO COMPARADOR

Es un amplificador sumador usado para comparar el valor medido desde el puente de DC con el valor de referencia. En este equipo las señales son acondicionadas de tal forma que tienen signo opuesto, con lo que la suma, en realidad, es una desviación entre estas señales. El valor de este error se puede medir en el conector B del panel frontal.

ELEMENTO CONTROLADOR

Genera una señal de control que transmite al elemento corrector a partir de una señal proporcional.

En este equipo, el elemento controlador se selecciona para generar dos tipos de control: Continuo y de dos etapas.

La señal de control se mide en el conector C del panel frontal.

TIPOS DE CONTROL

Existen dos tipos de control que se pueden implementar en el proceso:

i) Continuo: Puede ser interno o externo. En el primer caso, se presenta únicamente el control proporcional, mientras que, en el segundo, se puede usar cualquier tipo de control.

ii) De dos etapas: En este caso se tiene un control encendido-apagado o bang-bang.

MOTOR

En el proceso, el motor se utiliza como fuente de poder que genera señales eléctricas entre 15 y 80W, según la señal de control.

ELEMENTO CORRECTOR

Aquí el elemento corrector es una malla de alambre calentada eléctricamente que se alimenta en forma directa por la señal de salida del motor. Su función es la de transferir calor al aire que circula por el tubo del proceso.

IV. DESARROLLO

1) Coloque el sistema en la configuración mostrada en la figura 2 con un nivel de referencia inicial de 20°C y el elemento detector en el punto de prueba de 279mm.

2) Conecte un canal del osciloscopio en la terminal de salida.

3) Con un valor de la banda proporcional de 100%, aplique un cambio de referencia tipo escalón por medio del control de perturbación interna.

4) Observe la forma de la respuesta del sistema a este cambio.

5) Para diferentes valores de la banda proporcional (100-180%), repita el procedimiento anterior, anotando, para cada paso, el valor de la variable de desviación.

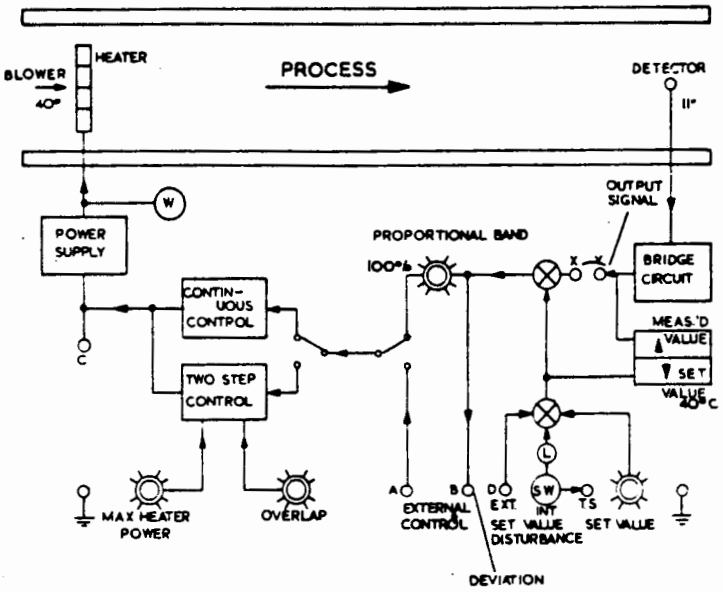


Fig. 2

V. CUESTIONARIO

1. Dibuje en forma de diagrama de bloques el sistema implementado, explicando detalladamente que función realiza cada bloque y cuál es su equivalente en el PT326.

2. ¿Qué tipo de control fue utilizado?.

3. Con base en las observaciones de la respuesta del sistema, ¿cuál sería la función de transferencia del mismo?.

4. Obtenga una gráfica de la desviación en función de la banda proporcional. Dé sus comentarios y explique el comportamiento del mismo.

VI. BIBLIOGRAFIA

1. Feedback LTD. Manual del sistema PT326. Feedback LTD.

2. OGATA, K. Ingeniería de control moderna. Prentice Hall.

PRACTICA 2

SERVO MOTOR DE CORRIENTE DIRECTA (DC)

I. OBJETIVOS

- Familiarizar al estudiante con el uso de los elementos básicos del servo-motor de DC MS150.
- Implementar un control de posición de lazo cerrado.

II. MATERIAL Y EQUIPO

- Amplificador operacional 150A
- Unidad atenuadora 150B
- Unidad de pre-amplificación 150C
- Servo-amplificador 150D
- Fuente de poder 150E
- Servo-motor 150F
- Potenciómetros 150H y 150K
- Osciloscopio o multímetro

III. INTRODUCCION

El servo-modular MS150, fabricado por FEEDBACK, es un equipo que permite implantar una amplia gama de controladores los cuales poseen todas las características de los utilizados a nivel industrial, con la ventaja de mostrar el funcionamiento de cada uno de sus componentes de una manera clara y sencilla.

Las partes fundamentales de las que consta el MS150 son: Un servo-motor de DC y un servo-motor de AC.

Esta primera práctica está diseñada para obtener un conocimiento general del uso y funcionamiento del servo-motor de DC, mientras que la siguiente, se destina al de AC.

A continuación se presenta una descripción de los elementos

principales que conforman al sistema, para posteriormente implantar un control de posición de lazo cerrado.

a. Descripción del equipo

En esta sección se describen las características pricipales de cada uno de los elementos (módulos) del servomecanismo modular MS150.

FUENTE DE ALIMENTACION (PS150E)

Esta unidad provee las alimentaciones necesarias para que funcione toda la unidad MS150. Está diseñada para aceptar a la entrada voltajes de 100/125V o de 200/250V a una frecuencia de 50/60 Hertz. A la salida de la unidad se obtienen 3 niveles de voltaje:

+24 volts de DC sin regular a 2 amperes

+15 volts de DC regulados a 60 miliamperes

22.5 volts de AC a 1 amper

Además de estar provista de un circuito de protección contra sobrecargas, contiene un amperímetro para indicar el consumo de corriente del motor, que es aproximadamente proporcional a la carga y, en la carátula, es posible observar cuándo se alcanza la condición de sobrecarga.

SERVOAMPLIFICADOR (SA150D)

Su función es la de operar al motor, está directamente conectada a éste mediante un conector de 8 vías. En la carátula de esta unidad se tienen marcadas líneas punteadas las cuales indican el modo de conexión para conseguir características aproximadas a un control por campo o por armadura. Tiene, asimismo, un circuito protector que limita la corriente de salida a 2 amperes.

Esta unidad cuenta con dos entradas para controlar el motor, una por cada bobina de campo, que son las que determinan el sentido de giro del motor (ver figura 1).

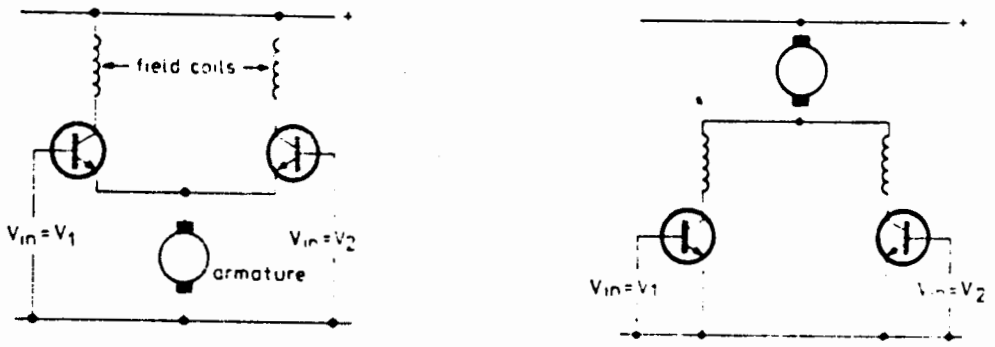


Fig. 1 Tipos de control para el motor de DC.

PREAMPLIFICADOR (PA150C)

Esta unidad provee de voltajes positivos a las entradas del servoamplificador con objeto de facilitar la rotación del motor en ambos sentidos a través de una señal aplicada al preamplificador. Este último está integrado por dos canales de entrada y dos de salida, uno de ellos en push-pull para el manejo directo del servoamplificador. Su ganancia es de aproximadamente 25.

UNIDAD ATENUADORA (AU150B)

Está constituida por dos potenciómetros de 10KΩ cada uno, calibrados y montados en un chasis. Son utilizados como divisores de voltaje y, por lo tanto, como controladores de ganancia.

AMPLIFICADOR OPERACIONAL (OU150A)

Este tipo de amplificador tiene muy alta ganancia y la corriente que consume a la entrada es prácticamente despreciable. Se emplea en los sistemas de control por su fácil manejo y gran versatilidad. Su función de transferencia es:

$$V_s(s)/V_e(s)=-K$$

En el sistema MS150, el amplificador operacional se utiliza,

fundamentalmente, como sumador, además de que se pueden generar tres modos diferentes en su lazo de retroalimentación por medio de:

i) Una resistencia de 10KΩ, lo que provee ganancia unitaria.
ii) Una resistencia y un capacitor, y
iii) Algún elemento definido por el usuario.

MOTOR DE DC (MT150F)

Es un servomotor que está provisto de un eje que permite efectuar el montaje directo de un disco inercial. El motor tiene un tacogenerador interconstruido, que es un pequeño generador de DC que hace la medición de la velocidad del motor a través de la lectura del voltaje entre sus terminales de salida. La relación velocidad-voltaje es aproximadamente de 2.7 a 3V por cada 1000 rpm.

Por otro lado, cuenta, además, con un eje lento implantado con un tren reductor de engranes con una relación de 30 a 1, situado perpendicularmente al eje principal del motor.

POTENCIOMETRO DE ENTRADA Y SALIDA (IP150H Y OP150K)

Son potenciómetros de alambre montados y ajustados con un medidor de aguja indicadora debidamente calibrada. El potenciómetro de entrada entrega un voltaje proporcional al ángulo de giro de la aguja, mientras el de salida posee, también, la característica de que el movimiento de la aguja está controlado por el del eje lento del motor.

UNIDAD ATENUADORA (LU150L)

Está compuesta por un disco metálico montado en el eje del motor que se hace girar entre los polos del imán de un freno magnético de tal forma que se tiene el siguiente funcionamiento:

Al girar el disco se producen corrientes de remolino (Eddy) que, a su vez, generan un campo magnético el cual produce un par que se opone al movimiento del eje del motor. Este par es denominado "par de freno".

El area de interferencia polo/disco puede ajustarse con una escala de 0 a 10 para proporcionar al motor una carga variable.

IV. DESARROLLO

1) Alambre el circuito de la figura 2 conectando inicialmente las dos entradas del preamplificador 150C a tierra.

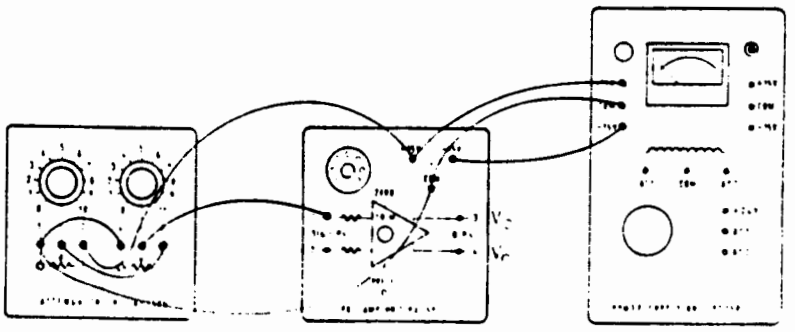


Fig. 2

2) Ajuste el valor entre las salidas 1 y 2 del pre-amplificador 150C a cero volts.

3) Conecte la salida del POT2 a la entrada de la unidad 150C como se muestra, y coloque el cursor de éste en la posición 1.

4)Para diferentes valores de salida de POT2 (variándolo de extremo a extremo) obtenga las lecturas tomadas en las salidas del pre-amplificador.

5) Alambre el circuito de la figura 3 conectando la unidad 150D para conseguir un control por armadura y acoplando el potenciómetro de salida 150K a la flecha de baja velocidad del motor.

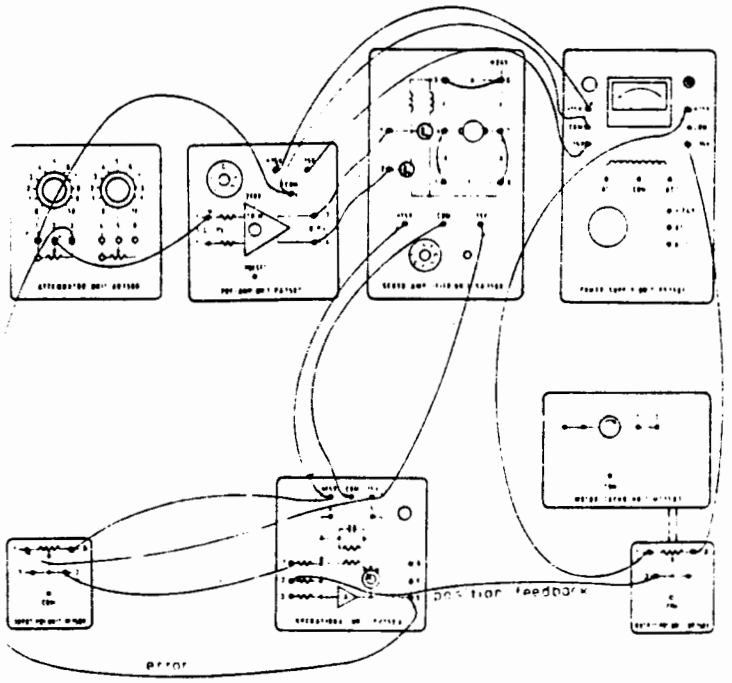


Fig. 3

6) Con el potenciómetro de la unidad atenuadora 150B en 1, coloque el potenciómetro de entrada 150H en una posición arbitraria.

7) Observe el comportamiento del motor y la posición del potenciómetro de salida.

8) Repita el procedimiento para diferentes ganancias en la unidad atenuadora 150B.

V. CUESTIONARIO

1. Realice una gráfica con los valores obtenidos en la etapa 3 del desarrollo y calcule la ganancia del pre-amplificador.

2. Explique el funcionamiento del pre-amplificador.

3. Explique el funcionamiento del sistema de la figura 3 y justifique los resultados conseguidos durante el experimento.

VI. BIBLIOGRAFIA

1. FEEDBACK LTD. Manual de operación del servomecanismo modular MS150.

2. OGATA, K. Ingeniería de control moderna. Prentice Hall.

PRACTICA 3

SERVO MOTOR DE CORRIENTE ALTERNA

I. OBJETIVOS

- Familiarizar al estudiante con el manejo del equipo servo-modular MT 150 en su parte de Corriente Alterna (AC).
- Implantar un sistema de control de velocidad de AC en lazo abierto.

II. MATERIAL Y EQUIPO

- 1 Servo-modular de AC MS 150 A
- 1 Osciloscopio de dos canales

III. INTRODUCCION

El segundo de los elementos principales que componen al servo-modular MT150 fabricado por FEEDBACK es el servo-motor de corriente alterna. Se utiliza este tipo de elemento cuando la señal en un sistema de control es AC, en estos casos un motor de inducción de dos polos se puede usar como servo-motor.

El diagrama básico de este tipo de motores se muestra en la figura 1, en donde un embobinado es energizado desde una fuente de AC (REF), mientras que el otro, lo es por una señal defasada de la referencia.

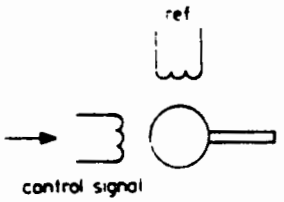


Fig.1 Esquema básico de un motor de dos polos

Se demuestra que, variando el suministro en la señal de control, se varía el par del motor a cualquier velocidad.

Al igual que en su parte de DC, el servo-modular MT150 provee, vía el servo-motor de AC 150U, un medio accesible para implantar este tipo de sistemas.

A continuación se presenta una breve descripción de las características principales del equipo a usar.

a. Descripción del equipo

UNIDAD DE MOTOR-TACOGENERADOR MT150U

La figura 2 muestra la carátula general de este dispositivo. Los devanados de referencia del motor y del tacogenerador están permanentemente conectados a un suministro de referencia. El devanado de control del motor se conecta a los colectores de transistores de potencia cuya corriente es controlada por una alimentación positiva desde el pre-amplificador. Una señal de corriente alterna superimpuesta a esta señal de polarización produce que los transistores conduzcan alternadamente, produciendo campos magnéticos que actúan directamente sobre el motor. La salida del tacogenerador se conecta a un transformador que puede ser unido a una red de defasamiento.

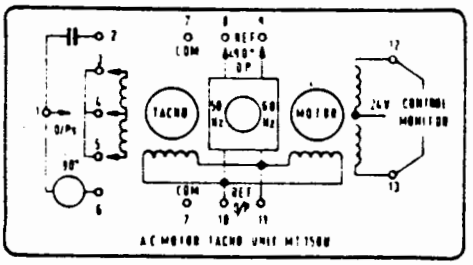


Fig. 2 Motor-Tacogenerador 150U.

PRE-AMPLIFICADOR DE AC PA150V

La unidad de preamplificación de AC es una combinación de un

amplificador operacional configurado como sumador y un desplazador ajustable de fase. La carátula general se representa en la figura 3.

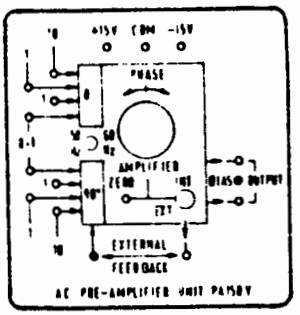


Fig. 3 Pre-amplificador de AC PA150V.

IV. DESARROLLO

1) Alambre el circuito de la figura 4.

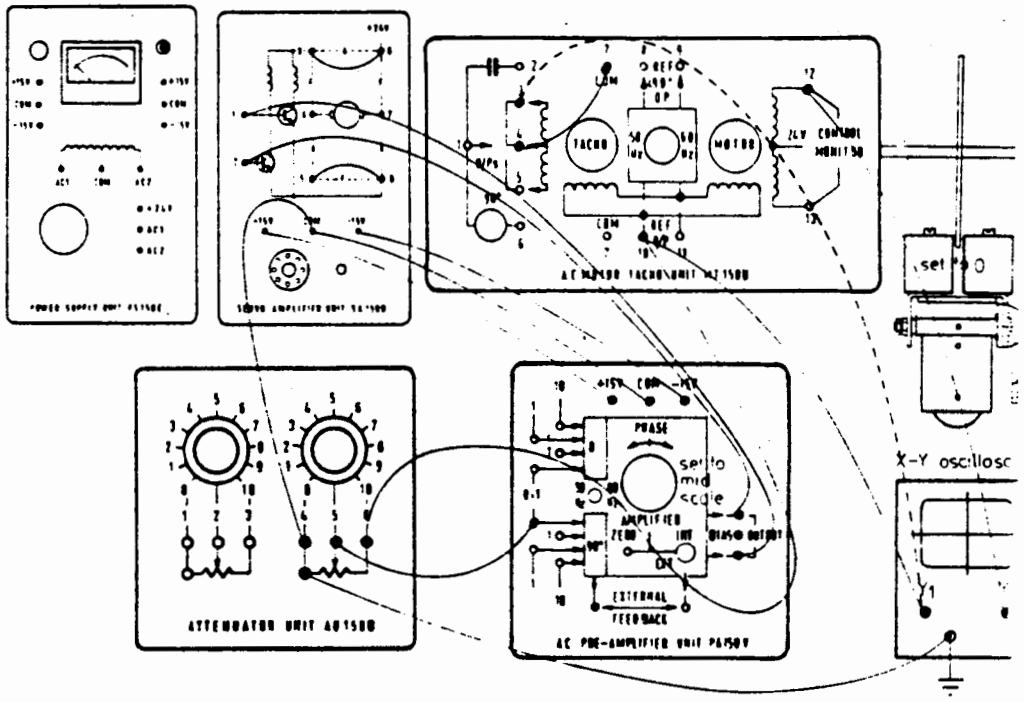


Fig. 4

2) En el módulo AU150B, coloque el cursor del potenciómetro en cero.

3) En el osciloscopio, coloque la entrada Y en 10V/cm y la base de tiempo en 5ms/cm.

4) Para evitar movimiento sin control del motor, aplique el freno magnético de tal forma que no se presente aceleración.

5) Por medio del control de fase, calibre la señal de entrada alrededor de 20V pico a pico.

6) Aplique esta señal a la entrada $0.1\angle 0^{\circ}$ de la unidad PA150V y con el mismo control de fase ajuste hasta que el comportamiento del motor sea estacionario.

7) Utilice nuevamente la señal a la entrada marcada en el circuito.

8) Varíe el potenciómetro de la unidad AU150B hasta un 50% (anote sus comentarios acerca del comportamiento del sistema).

9) Conecte la terminal 3 del transformador de salida del tacogenerador a la terminal Y del osciloscopio en lugar de la referencia e incremente la sensibilidad a 1V/cm.

10) Coloque el potenciómetro de entrada a 2 y el freno a cero.

11) Obtenga las características torque/velocidad con un control de lazo abierto anotando el valor de voltaje pico a pico del tacogenerador para las diferentes posiciones del freno magnético.

12) Apague el sistema.

V. CUESTIONARIO

1. Explique las causas por las que el sistema se comporta como en el punto 8 del desarrollo.

2. Grafique los valores obtenidos en el paso 11 del procedimiento tomando como abcisas las posiciones del freno y, comente la forma de la respuesta conseguida.

VI. BIBLIOGRAFIA

1. FEEDBACK LTD. Manual del servo-sistema modular MS150. Feedback LTD.

PRÁCTICA 4

COMPUTADORA ANALÓGICA

I. OBJETIVO

- Familiarizar al estudiante con la computadora analógica para permitirle su empleo como auxiliar en el estudio de sistemas dinámicos.

II. MATERIAL Y EQUIPO

- Computadora analógica EAI-2000 con accesorios (cables, tableros, etc.)
- Osciloscopio con memoria
- Graficador X-Y analógico
- Multímetro

III. ANTECEDENTES TEÓRICOS

a. Computadora analógica

Una clase muy amplia de sistemas físicos se pueden modelar matemáticamente a través de ecuaciones diferenciales ordinarias. Cuando dos sistemas tienen como modelo matemático la misma ecuación diferencial, se dice que dichos sistemas son análogos [Refs. 1 y 2].

Una computadora analógica es un dispositivo eléctrico que permite implantar con facilidad análogos eléctricos de sistemas físicos cuyo modelo sea de la clase descrita. La computadora analógica se construye alrededor de un conjunto de amplificadores operacionales alambrados internamente para comportarse como sumadores e integradores; a estos dos dispositivos básicos se les agregan potenciómetros, fuentes de alimentación, y, en el caso de algunas computadoras analógicas complejas, dispositivos tales como multiplicadores, generadores de señales, compuertas lógicas, etc.

b. Ecuaciones diferenciales ordinarias

Una ecuación diferencial ordinaria es una expresión matemática que tiene la siguiente forma:

$$y^{(n)}(t) = f\,(y^{(n-1)}(t), y^{(n-2)}(t), \ldots, y(t), u(t), t) \qquad (1)$$

que expresa que la n-ésima derivada de una variable y(t), llamada salida, es una función de sus derivadas de orden menor, de otra variable u(t), denominada entrada, y de un índice t, que normalmente denota el tiempo. Un sistema cuyo modelo matemático adopta la forma (1) se conoce como sistema dinámico.

El modelo matemático que representa la ecuación (1) se identifica también como *descripción externa* para un sistema pues los elementos primordiales en el mismo son la entrada y la salida; todo aquello que ocurra al interior del sistema, entre la entrada y la salida, no es relevante en este caso.

En el estudio del comportamiento de los sistemas dinámicos son de particular interés los casos en que la ecuación (1) es lineal e invariante con el tiempo, es decir, cuando tiene la forma:

$$y^{(n)}(t) + a_1 y^{(n-1)}(t) + \ldots + a_n y(t) = u(t) \qquad (2)$$

con a_i $(i=1, \ldots, n)$ constantes definidas sobre el campo de los reales.

Una vez conocido el modelo matemático de un sistema, es posible pensar en obtener la solución a la ecuación (2), o sea, en hallar una función y(t) que sustituida en (2) convierta la expresión en una identidad.

Existe un número infinito de funciones que satisface la ecuación en cuestión, sin embargo, para el estudio de sistemas dinámicos se requiere encontrar una solución de dicho conjunto infinito que cumpla con un

conjunto de restricciones adicionales que se refieren a especificar el valor de la función y(t), o de sus derivadas, en un instante de tiempo determinado.

Del conjunto de posibilidades para elegir las restricciones adicionales, hay uno que tiene especial importancia pues la solución que se encuentra a partir de él permite describir la evolución del comportamiento del sistema partiendo de un instante de tiempo determinado. Dicho conjunto está dado por las siguientes restricciones:

$$\begin{aligned} y(t_o) &= y_o \\ y^{(1)}(t_o) &= y_o^{(1)} \\ &\vdots \\ y^{(n-1)}(t_o) &= y_o^{(n-1)} \end{aligned} \qquad (3)$$

para un cierto t_o, o tiempo inicial. Para los sistemas descritos por la ecuación (2) este tiempo siempre puede elegirse de tal forma que:

$$t_o = 0 \qquad (4)$$

Cuando se soluciona una ecuación diferencial de la forma (2), sujeta a las restricciones indicadas en (3), se dice que se resuelve un problema de condiciones iniciales.

La solución analítica al problema de condiciones iniciales se puede consultar en textos de ecuaciones diferenciales [Refs. 3 a 5], sin embargo, en esta práctica es de interés su solución a través de una computadora analógica.

c. Sistemas de ecuaciones diferenciales ordinarias descripción en variables de estado

Existen ocasiones en que es conveniente conocer el comportamiento

hacia el interior del sistema dinámico en estudio, en este caso se recurre a una *descripción interna o de estado* del sistema; la estructura del modelo para la misma es de la forma:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\,\mathbf{x}(t) + \mathbf{B}\,\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\,\mathbf{x}(t) + \mathbf{D}\,\mathbf{u}(t) \end{aligned} \qquad (5)$$

con $\mathbf{x}(t)$, $\mathbf{u}(t)$, $\mathbf{y}(t)$, vectores de dimensión n, m y p, que se denominan estado, entrada y salida, respectivamente, **A**, **B**, **C** y **D** matrices con elementos reales, de dimensiones apropiadas. Tanto los elementos de los vectores como los de las matrices serán números reales.

El concepto de estado está fuertemente relacionado con la posibilidad de un sistema de resumir la historia de su comportamiento pasado [Ref. 6]. También en este caso cuando se cuenta con el conjunto de restricciones:

$$x\,(to) = x_o \qquad (6)$$

se habla de un problema de condiciones iniciales, y de hallar una solución única $\mathbf{x}(t)$ para la ecuación (5) con las restricciones (6).

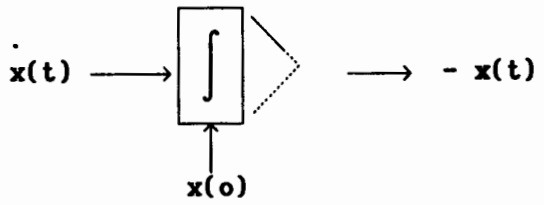
Es posible mostrar que toda ecuación diferencial ordinaria se puede representar mediante un sistema de ecuaciones diferenciales ordinarias (ver, por ejemplo, la Ref. 3).

d. Uso de la computadora analógica para simular sistemas dinámicos

Para implantar en una computadora analógica un sistema equivalente al que describen los modelos matemáticos de las formas mostradas arriba, (2) y (5), se utilizan tres elementos de la computadora analógica.

Integrador

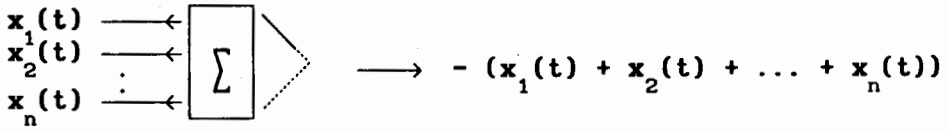
Un integrador es un elemento tal que la salida del mismo es la integral de su entrada, simbólicamente se representan como sigue:



como puede notarse, el integrador cambia el signo de la función que integra, esto se debe a que los amplificadores que implementan el integrador están alambrados en configuración inversora. Asimismo se ha incluido en el diagrama la condición inicial que señala el valor a partir del cual se iniciará la integración indicada y que permite resolver los problemas de condiciones iniciales descritos.

Sumador

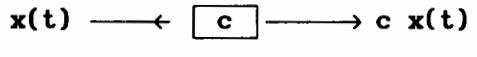
Un sumador es un circuito cuya salida es la suma de sus señales de entrada, simbólicamente es así:



de nueva cuenta, en el sumador se invierte el signo de las señales que se suman. En algunos casos el sumador tiene posibilidades de multiplicar por 10 el valor de alguna entrada antes de sumarla.

Potenciómetros

Estos dispositivos están previstos para escalar el valor de su entrada, simbólicamente su representación es así:



Como los potenciómetros se implementan a través de divisores de voltaje, las constantes de escalamiento aceptables están entre -1 y 1; el valor apropiado se selecciona al mover la perilla de un potenciómetro (de ahí su nombre), o, en el caso de la computadora EAI 2000 en que se realizará la práctica, seleccionando su valor por consola.

Interconexión de elementos

Los tres tipos de elementos descritos se interconectan por medio de cables apropiados, en todos los casos se cuenta con uno o varios bornes conectados a la entrada y uno o dos bornes a la salida.

IV. DESARROLLO

1) Alambrar el circuito que representa la ecuación diferencial de primer orden:

$$\frac{dx}{dt} + a\,x = u(t) \quad ; x(0)=0$$

2) Alambrar el circuito correspondiente a la ecuación diferencial de segundo orden:

$$\frac{d^2x}{dt^2} + b\,\frac{dx}{dt} + c\,x = u(t) \qquad ; \; x(0)=0; \; \frac{dx(0)}{dt} = 0$$

3) Alambrar el circuito que corresponde al siguiente sistema de ecuaciones diferenciales:

$$\dot{\mathbf{x}} = \begin{bmatrix} d & e \\ f & g \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(t) \qquad ; \; \mathbf{x}(0)=0$$

V. RESULTADOS

1. Para la experiencia (1) debe obtener lo siguiente:

   1.a. Gráfica de la respuesta a escalón para tres valores positivos diferentes del parámetro *a*.

   1.b. Igual a 1.a, pero con *a* negativo.

   1.c. Incluya una constante que multiplique al escalón de tal forma que los tres valores del parámetro *a* elegidos en 1.a den como valor final la unidad para la variable x. Obtenga de nueva cuenta las gráficas solicitadas.

2. Para la experiencia (2) considere los valores apropiados de los parámetros *b* y *c* que le permitan realizar lo siguiente:

   1.a. Gráfica de la respuesta a escalón considerando que el sistema es subamortiguado, críticamente amortiguado y sobreamortiguado, respectivamente.

   2.b. Gráfica de la respuesta a escalón para los tres casos de raíces inestables (con parte real positiva): reales diferentes, reales iguales y complejas.

   3.c. Normalice la respuesta a escalón para los tres casos de 1.b, de forma tal que la respuesta a escalón sea la unidad. Obtenga de nuevo las gráficas de 1.b con la respuesta normalizada.

3. Para la experiencia (3) debe obtener:

   3.a. Gráfica del comportamiento de cada una de las componentes del vector de estados (dos en este caso) cuando la matriz formada con los parámetros *d* a *g* tiene valores característicos negativos: diferentes, iguales y complejos.

   3.b. Igual que 3.a, pero para valores característicos positivos.

VI. CONCLUSIONES

Interprete las gráficas obtenidas y concluya sobre ellas. En

particular debe explicar las diferencias entre los resultados 1.a y 2.a, 1.b y 2.b, 1.c y 2.c, respectivamente. Comente, también, sobre las curvas obtenidas en 3.a y 3.b.

VII. BIBLIOGRAFIA

1. BOYCE, W.E. y R.C. Diprima. Ecuaciones diferenciales y problemas con valores en la frontera. Ed. Limusa. México. 1967.

2. CANALES, R. y R. Barrera. Análisis de sistemas dinámicos y control automático. Ed. Limusa. México. 1977.

3. CANNON, R.H. Dynamics of physical systems. McGraw-Hill Book Company. USA. 1967.

4. IMAZ, C. y Z. Vorel. Ecuaciones diferenciales ordinarias. Ed. Limusa México. 1968.

5. WELLSTEAD P.E. Introduction to physical system modelling. Academic Press. USA. 1979.

6. WYLIE, C.R. Differential equations. McGraw-Hill Inc. Japan. 1979.

NOTAS IMPORTANTES

1. Todas las gráficas solicitadas se deben obtener mediante el graficador analógico.

2. Para cada punto de A, B y C presente las gráficas solicitadas en una sola hoja, indique en ella los valores usados en los parámetros y normalizaciones, según proceda.

3. Obtenga la fuente de tiempo para las abcisas de las gráficas a partir de un integrador (integrando un escalón de pequeño valor).

4. El manejo de los elementos de la computadora a través de la consola se detalla en el manual respectivo.

PRACTICA 5

TARJETA DE ADQUISICION LAB-MASTER.

I. OBJETIVOS

- Introducir al alumno en el manejo de la tarjeta de conversión A/D D/A Lab Master.
- Apreciar la utilidad de la tarjeta Lab Master para el procesamiento digital de señales a través de la PC.
- Apreciar la forma en que se lleva a cabo la implementación de un sistema de adquisición de datos.

II. MATERIAL Y EQUIPO

- Un osciloscopio con punta de prueba
- Un generador de señales
- Cables de conexión
- Alambres de calibre pequeño (#20 ó #22)
- Computadora compatible con IBM PC/XT
- Editor de PASCAL

III. INTRODUCCION

Las computadoras digitales se usan cada vez con mayor frecuencia para implementar sistemas de control. Por esta razón es de gran importancia entender bien los sistemas controlados por computadora.

Podemos ver a los sistemas controlados por computadora como una aproximación a los controlados analógicamente, sin embargo, esta es una descripción pobre pues no está mostrando el potencial que en verdad representa al control por computadora.

La trascendencia del uso de una tarjeta con las características del Lab Master es que da a la computadora los recursos para tratar con la información analógica que se presente como una señal eléctrica con cambios

continuos sobre el tiempo y así poder interpetar, alterar y, quizá, almacenar ésta. La computadora también adquiere la capacidad de tener influencias analógicas (en la forma de voltajes controlados) sobre dispositivos o mecanismos externos.

Lab Master es una tarjeta electrónica diseñada específicamente para cubrir las necesidades de laboratorios de procesamiento de señales y de control donde se requiere efectuar pruebas de control en tiempo real. El Lab Master convierte a la PC en una herramienta multi-función para análisis de laboratorio. Vemos, así, que el Lab Master permite la interacción entre el mundo real y una computadora personal; de aquí que ésta adquiera la capacidad de llevar a cabo el control de ciertos procesos físicos.

a. Características del equipo

Algunos rasgos estándares de la tarjeta Lab Master son los siguientes:

ANALOGICO A DIGITAL

- 16 entradas sencillas u 8 diferenciales (expandible a 256 y 128 respectivamente)
- Resolución de 12 bits
- 30 KHz de rango estandar de conversión
- Rangos seleccionables de entrada: ±10V, 0 a +10V
- Formatos de salida: complemento a dos y binario
- Disparo externo controlado por la computadora
- Incremento automático repetitivo de canal sobre cualquier bloque de canales consecutivos
- Utiliza interrupciones por vector o por prueba de estatus de A/D
- Provisión para sincronización de A/D's de múltiples Lab Master's
- Incluye Muestreador-Retén de alta velocidad y un multiplexor de precisión

DIGITAL A ANALOGICO

- 2 convertidores D/A independientes
- Resolución de 12 bits
- Tiempo de conversión de 5 μs
- Rangos de salida seleccionables por alambrado interno independientes: ±2.5V, ±5V, ±10V, 0 a +5V, 0 a +10V
- Sólo 2 instrucciones de software son necesarias para la conversión
- Entrada en 2's complemento
- Cada DAC sostiene el valor previo hasta que una nueva palabra entera le sea presentada

TIMER

- 5 contadores independientes de 16 bits
- 16 líneas disponibles para uso externo
- Tiempo del día con resolución de 1/100 segundos
- Contador de eventos (velocidad hasta de 5 MHz)
- Comparador con alarma para 2 contadores
- Salidas en disparo simple "One Shot" o en frecuencia continua
- Disparo programable y selección de fuente de conteo
- Interrupción por vector
- Reloj interno con 1μs de precisión
- Puede iniciar la conversión A/D y ser usado independientemente para el tiempo del día, contar eventos, corrimientos en la frecuencia

ENTRADA DIGITAL / SALIDA DIGITAL

- 24 líneas de I/O paralela programables en grupos de 8 ó 12
- Pueden ser alambradas para interrumpir al CPU cuando esté listo para más datos o cuando el buffer de entrada esté lleno
- Usa el Intel 8255 PPI
- 3 modos de operación: 3 puertos de entrada o de salida; 2 puertos de entrada o salida con protocolo; puerto bidireccional de entrada y salida con protocolo
- Puede disparar en eventos externos
- Proporciona información de estatus para un dispositivo externo o recibir datos numéricos o alfanuméricos

b. Configuración general

El Lab Master está formado por dos tarjetas elaboradas en circuito impreso, una denominada "tarjeta madre" (Fig 1) y, la otra, "tarjeta hija" (Fig 3). La tarjeta madre contiene la lógica digital y se introduce directamente en el bus de la PC. La tarjeta hija contiene toda la circuitería de adquisición analógica, está conectada a la tarjeta madre mediante un cable de listón y viene empacada en una caja externa a la PC.

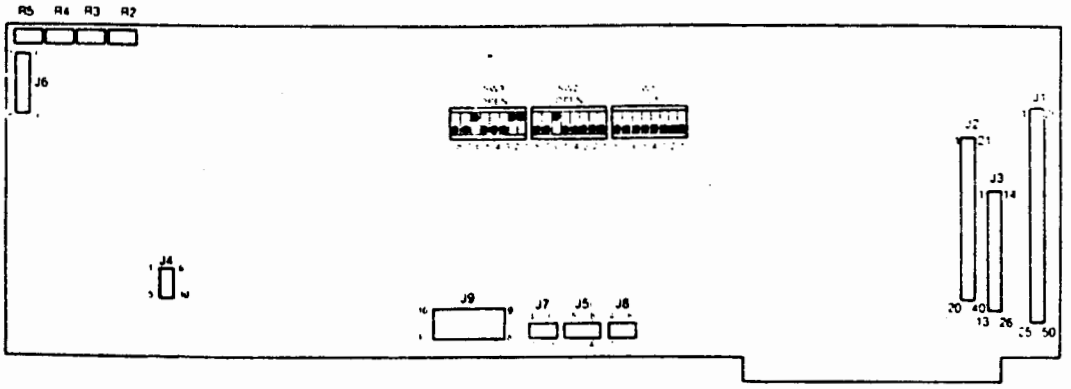


Fig. 1 Tarjeta madre

c. Descripción de la tarjeta madre

Como se mencionó anteriormente, es factible programar en forma de alambrado interno algunas características que nosotros necesitemos de la tarjeta:

DIRECCION DE LA TARJETA

Por medio del juego de interruptores denominados SW1, SW2 y SW3 ya ha sido configurada de fabrica la tarjeta para responder a una dirección inicial 1808_{dec} ó 710_{hex}; sin embargo, es factible cambiar dicha dirección cambiando el estado de los interruptores que se localizan en la parte superior de la tarjeta madre.

A través del puente J5 que se halla en la parte baja de la tarjeta, se puede controlar la opción de que ésta sea mapeada como memoria o como puerto.

CONVERTIDORES D/A

La salida de estos convertidores ya ha sido configurada para proporcionarnos un rango de salida de -10 a +10 V de los puentes J6.

CONEXION CON LA TARJETA HIJA

El conector J1 de 50 pins es la conexión a la tarjeta hija.

PIN	INPUT OR OUTPUT	SIGNAL NAME
21-40		GROUND
1,2		+5v
	A TO D SIGNALS	
16	OUTPUT	Done
3	INPUT	External Start Conversion
	9513 TIMER SIGNALS	
4	OUTPUT	OUT 5
9	INPUT	GATE 5
14	INPUT	SOURCE 5
5	OUTPUT	OUT 4
10	INPUT	GATE 4
15	INPUT	SOURCE 4
6	OUTPUT	OUT 3
11	INPUT	GATE 3
17	INPUT	SOURCE 3
7	OUTPUT	OUT 2
12	INPUT	GATE 2
18	INPUT	SOURCE 2
8	OUTPUT	OUT 1
13	INPUT	GATE 1
19	INPUT	SOURCE 1
20	OUTPUT	FOUT

Fig. 2 Distribución de pines en el conector J2

CONECTOR J2

Todas las entradas y salidas para el timer 9513 junto con las señales de DONE (conversión terminada) y EXTERNAL START CONVERSION del convertidor

A/D están en este conector. El cual tiene ya acoplado un cable junto con su correspondiente conector hembra. Los pins y sus correspondientes nombres se presentan en la figura 2.

. PUERTOS PARALELOS

Las 24 líneas de entrada/salida del puerto paralelo 8255 están ubicadas en el conector de 26 pins J3.

b. Descripción de la tarjeta hija

CONEXION CON LA TARJETA MADRE

El conector P2D es la conexión a la tarjeta madre.

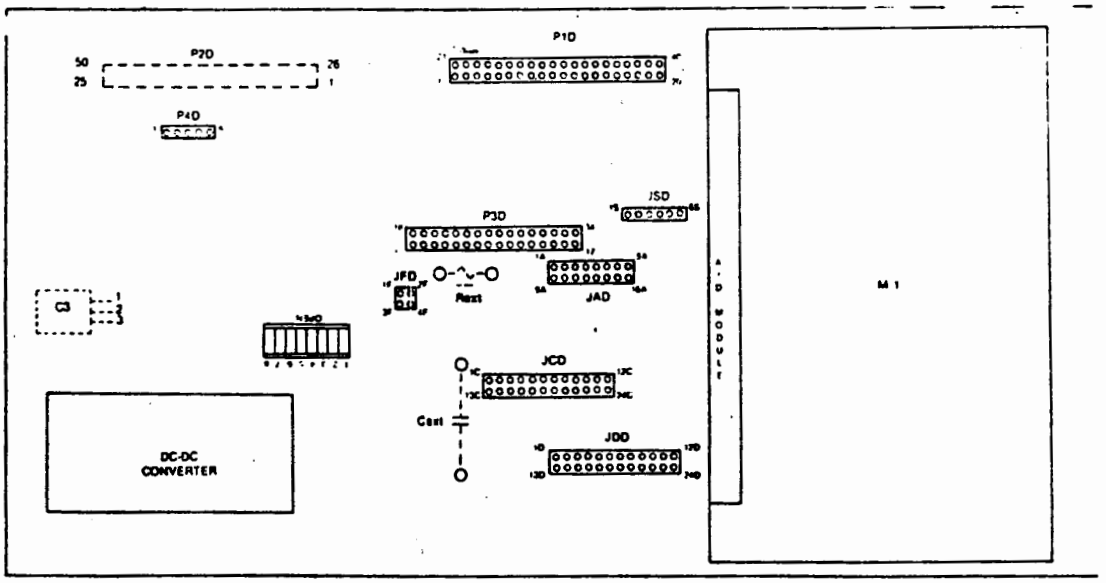


Fig. 3 Tarjeta hija

ENTRADAS ANALOGICAS

El conector P1D es la conexión a los voltajes de entrada. Los pins de entrada son como sigue:

PIN	CANAL ENTRADA (SENCILLO)	CANAL DE ENTRADA (DIFERENCIAL)
20	0	0
19	8	0
18	1	1
17	9	1
16	2	2
15	10	2
14	3	3
13	11	3
12	4	4
11	12	4
10	5	5
9	13	5
8	6	6
7	14	6
6	7	7
5	15	7
1-4,21-40	tierras para terminales sencillas y diferenciales entrada negativa para modo pseudodiferencial.	

IV. DESARROLLO

1) Identifique los siguientes elementos del sistema:
   a)Tarjeta hija.
   b)Cable de conexión entre tarjeta madre y tarjeta hija.
   c)Cable de salida de la tarjeta hija.
   d)Terminales de salida de la tarjeta madre (juego de tres cables solos).

e)Cable proveniente del conector J2 de la tarjeta madre.

Las terminales de salida de la tarjeta madre corresponden a los canales 0,1 de conversión D/A y a la tierra digital.

2) Remueva la cubierta (café) de la tarjeta hija, ubique el conector PID e identifique cuáles son las tierras y cuáles, los canales de entrada y etiquételos en el conector externo correspondiente.

3) Encienda la PC , introdúzcase al editor del lenguaje Turbopascal 5 y elabore un programa que tenga la capacidad de llevar a cabo conversiones de tipo D/A.

El programa estará formado por diversos procedimientos que se encargarán de generar los datos digitales, convertirlos a complemento a dos y finalmente mandarlos al convertidor D/A. Estos dos últimos procedimientos ya se encuentran implementados en unidades (ver anexo 4)a las que se puede accesar desde cualquier programa escrito en Turbopascal 5. A continuación se propone la estructura que deberá tener el programa y la manera en que son accesadas las rutinas de conversión.

```
PROGRAM DA_MASTER;
USES
     CRT,LABCOM;
VAR
     Y:REAL;
     CANAL:INTEGER;
. . .
BEGIN {Inicia Prog Principal}
     . . .
     LEE_DATO(Y);{Lee el valor del dato y}
     DA_OUT(Y,CANAL);{Manda dato al convertidor D/A}
     . . .
END.{Fin del Prog Principal}
```

Nótese que se hace referencia a la unidad LABCOM, en esta unidad se encuentran contenidas las rutinas que permiten manejar al Lab-Master. Por ejemplo, allí se halla la rutina DA_OUT a la que se hace referencia en este programa.

Si revisa el listado de la rutina que se muestra en el anexo 4, observará que para enviar un cierto valor hacia el convertidor D/A se requiere mandar dos paquetes de bits uno de 4 y el otro de 8, esto se debe a que el bus de datos de la computadora maneja tan sólo 8 bits a la vez; otra observación es que el procedimiento manda primero los 4 bits más significativos y en seguida los 8 bits restantes, esto a causa de que el convertidor acepta primero los bits más significativos sin iniciar la conversión hasta que los restantes 8 bits le sean mandados.

4) Una vez editado y compilado el programa, proceda a probar su funcionamiento. Tome la terminal del Canal 0 ó 1 del convertidor D/A e introdúzcala en uno de los canales del osciloscopio, lo mismo haga con la tierra. Encienda el osciloscopio y ajústelo a una escala de 5 V/div. Ejecute el programa y procure introducir valores en el rango $-10<y<10$. Elabore comentarios y anote sus observaciones.

5) Estando en el editor de Turbopascal 5, proceda a elaborar un programa que permita apreciar la capacidad que proporciona el Lab Master para efectuar la conversión de señales analógicas a señales digitales bajo un cierto periodo de muestreo.

El programa deberá estar formado de varias subrutinas que ejecutarán las tareas de programación del convertidor (número de canal e inicialización), programación del contador que va a proporcionar una serie de pulsos para disparar al convertidor con una cierta frecuencia de muestreo, de adquisición de datos provenientes del convertidor (estas rutinas ya están implementadas en la unidad LABCOM) y, finalmente, de graficación. A continuación se propone la estructura que deberá tener el programa.

```
PROGRAM AD_MASTER;
USES
     CRT, LABCOM, GRAPH;
VAR
     T, CANAL: INTEGER; {Dirección inicial de la tarjeta}
     Y, FREC: REAL;
. . .
BEGIN {Inicia Prog Principal}
     . . .
     PROG_CANAL(CANAL, FREC); {Programa al convertidor y al contador}
     . . .
     AD_IN(Y);
     TRAZAR(T, Y); {Grafica Y vs T}
     . . .
END. {Termina Prog Principal}
```

Nótese que en este programa se hace referencia a las rutinas PROG_CANAL y AD_IN las cuales se encuentran dentro de la unidad LABCOM. Si se revisa el listado de la rutina AD_IN notará que para tomar un dato del convertidor, aquél se tiene que tomar en dos paquetes pues, el convertidor es de 12 bits y el bus de la PC es de 8, tomando primero el paquete de los 8 bits menos significativos y luego, el paquete de los 4 bits más significativos. Obsérvese, también, que en este procedimiento se reúnen ambos juegos de bits en una sola variable y después se hace una manipulación a fin de escalar a la variable a un rango de ±10.

Elabore comentarios y explique lo observado.

6) Para ejecutar este programa habrá que hacer antes algunos ajustes. En el conector J2 se encuentran las terminales del Timer, A/D DONE (conversión terminada) y disparo externo del convertidor A/D, hay que unir con un puente las terminales 4 y 3 que corresponden a la salida del contador 5 y a la entrada del disparo externo del convertidor, esto se hace ya que se va a muestrear una señal analógica y el contador 5 es el que va a proporcionar el periodo de cada muestra.

7) Proceda a encender el generador de señales y a ajustarlo en una frecuencia baja, por ejemplo de 5 Hz, y en la forma de onda que desee, con una amplitud que no sobrepase los ±10 V; se recomienda quitar el nivel de offset de la señal. Conecte la salida del generador de señales a uno de los 16 canales (0 a 15) de conversión A/D. Elabore sus comentarios y explique lo observado.

V. BIBLIOGRAFIA

1. Scientific Solutions Incorporated. Instalation manual & users guide. Scientific Solutions Presses. Solon Ohio 1985.

PRACTICA 6

DISEÑO ASISTIDO POR COMPUTADORA

I. OBJETIVO.

- Dominar los paquetes interactivos comerciales más comúnmente empleados para el análisis y diseño de sistemas de control.

II. MATERIAL Y EQUIPO

- Computadora IBM PC, PC/XT, PC/AT o compatible con al menos 320K de memoria
- Impresora compatible con EPSONMX-80 o con IBM5152 gráfica
- Coprocesador 8087, 80287 ó 80387
- Paquete "Program CC"
- Paquete "Program PC-MATLAB
- Paquete "Program SIMNON"
- Al menos un manejador de disco flexible
- Sistema operativo MS-DOS o PC-DOS con versión mayor a dos

III. ANTECEDENTES TEÓRICOS

a. Compensadores de adelanto y de atraso

Considere la configuración de un sistema retroalimentado con un grado de libertad de la figura 1. El problema de control consiste en diseñar el compensador en serie $KG_c(s)$ de tal manera que se satisfagan las especificaciones más frecuentes del sistema en lazo cerrado como son velocidad de respuesta, estabilidad relativa, constantes de error, rechazo o atenuación de perturbaciones, etc.

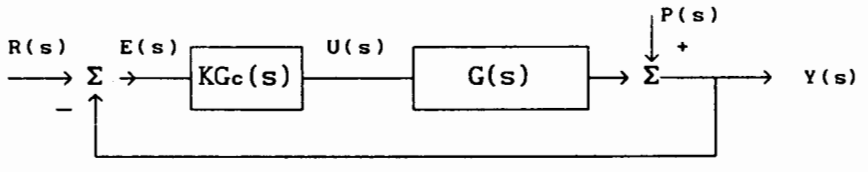


Fig. 1 Sistema retroalimentado con un grado de libertad

Por razones de simplicidad se limita la función de transferencia del compensador $G_c(s)$ a una función propia con un polo y cero dada por:

$$G_c(s)= \frac{Ts + 1}{\beta Ts + 1} \qquad (1)$$

con $\beta>1$ para un compensador de atraso y $\beta \in (0,1)$ para uno de adelanto. Por tanto el problema se reduce a encontrar los valores K, T y β que permitan satisfacer las especificaciones en lazo cerrado.

Dadas las especificaciones del sistema, no se conoce a priori, en general, si existe al menos un juego de valores (T, β, K) con los cuales se logre el desempeño requerido. Existen casos donde es posible emplear tanto un adelanto como un atraso; sin embargo, puede suceder que sólo un tipo de compensador deba ser usado o una combinación de ambos (adelanto-atraso). Cuando se tienen requerimientos muy estrictos puede ocurrir que no exista un compensador que los satisfaga.

En general, se dice que un compensador de adelanto permite obtener una respuesta transitoria más rápida, o sea, aumenta el ancho de banda a costa de incrementar el ruido de altas frecuencias y disminuir en algunos casos, los márgenes de estabilidad. Por el contrario, un atraso, brinda la posibilidad de mejorar los regímenes estacionarios y los márgenes de estabilidad a expensas de incrementar el tiempo de respuesta.

Por otro lado, existe un gran número de procedimientos para ajustar los compensadores empleando tanto el lugar de las raíces como la respuesta en frecuencia (ver [1] -[3]).

A continuación se bosqueja un procedimiento basado en la respuesta frecuencial tomado de J. Melsa & D. Schulz [1].

1.- Determinar la ganancia K que satisfaga los requisitos de errores en estado permanente.

2.- Obtener las especificaciones que son violadas usando el valor de K

previamente obtenido y, con base en ellas, proponer el tipo de compensador.

3.- Determinar la contribución máxima en fase que se requiere para satisfacer el margen deseado ϕ_m y, con base en ellas, proponer el tipo de compensador a usar.

4.- En caso de elegir un compensador de adelanto, ajustar β y T empleando:

$$\phi_m = \text{arc sen } \frac{1 - \beta}{1 + \beta} \qquad (2)$$

$$\omega_{max} = \frac{1}{T\sqrt{\beta}} \qquad (3)$$

donde ω_{max} es la frecuencia en la cual $|KG(j\omega_{max})|db = -20 \log(1/\sqrt{\beta})$.

5.- En caso de tratarse de un compensador de atraso se propone:
a) encontrar la frecuencia ω_c' donde la fase es $(-\pi + \phi_m)$ donde ϕ_m es el margen de fase requerido y elegir ésta como la frecuencia de corte del sistema compensado.
b) elegir $\omega_1=1/T$ por lo menos una octava por debajo de ω_c' y ajustar el valor de β de tal manera que se cumplan las siguientes condiciones:

$$|KG(j\omega_c')G_c(j\omega_c')|. = 1 \qquad (4)$$

$$\arg[KG(j\omega_c')G_c(j\omega_c')] = -\pi + \phi_m \qquad (5)$$

6) Comprobar que el sistema compensado cumple con el desempeño deseado.

b. Solución de ecuaciones diferenciales ordinarias

Existen básicamente dos filosofías para determinar el comportamiento

de un modelo descrito por ecuaciones diferenciales ordinarias usando una computadora digital de propósito general. Una de ellas consiste en discretizar la ecuación diferencial, obteniendo ecuaciones en diferencias que se implementan fácilmente en la computadora; su desventaja es la complejidad para discretizar ecuaciones no lineales.

La otra, reside en transformar la ecuación en una de estado y resolverla numéricamente. La idea básica consiste en obtener a partir de:

$$\dot{x} = f(x,u,t); \quad x(0)= x_o \tag{6}$$

una solución aproximada $\hat{x}(t)$ de (6) utilizando su expansión en serie de Taylor:

$$x(\tau) = x(\tau-\Delta t) + \Delta\tau\frac{dx}{dt}\Big|_{\Delta\tau} + \frac{\Delta\tau^2}{2!}\cdot\frac{d^2x}{dt^2}\Big|_{\Delta\tau} + \frac{\Delta\tau^3}{3!}\cdot\frac{d^3x}{dt^3}\Big|_{\Delta\tau} + \ldots \tag{7}$$

Existen muchos algoritmos para aproximar (7). Los más empleados son el de Runge Kutta de segundo y cuarto orden y el predictor corrector (ver [4] para más detalles). La precisión de la solución depende en todos los casos de la elección del incremento Δt y el tipo de algoritmo empleado.

En particular, el paquete de simulación SIMNON resuelve numéricamente ecuaciones diferenciales del tipo (6) usando un Runge-Kutta de cuarto orden. Además resuelve ecuaciones en diferencia dadas por:

$$x(k+1)= f(x(k),k); \quad x(0)=x_o \tag{8}$$

Asimismo el paquete está dotado de un conjunto de comandos que permite acoplar subsistemas y obtener los resultados gráficamente en el monitor e impresora.

A continuación se indican los pasos básicos para la solución de ecuaciones diferenciales cuyos resultados se grafican en la pantalla. El texto subrayado representa comandos de SIMNON y no deben omitirse. Por simplicidad, el texto en letras itálicas corresponde a comentarios. En el anexo A2 se encuentran los comandos básicos del paquete tomados del manual [6].

Paso 1. Definir con un editor de texto (por ejemplo TURBO) el sistema a simular y guardarlo en un archivo con extensión ARCHI.T.

```
CONTINUOUS SYSTEM EJEMPLO
INPUT U                 definición de la variable de entrada u
OUTPUT Y                nombre de la variable de salida
STATE X1 X2             definición de las variables de estado
DER DX1 DX2             nombre de las derivadas
TIME T                  definición de la variable independiente t
U = T                   entrada rampa
DX1=X2                  ecuaciones del sistema
DX2=-A0*x1-A1*x2+B*U
Y=X1
A0:2                    valores de las constantes
A1:1
B:.1
END                     fin de datos
```

Paso 2. Dentro de SIMNON, teclear el comando SYST

```
> SYST ARCHI
```

Si hay error en la definición del sistema salir con

```
> LEAVE
```

en caso contrario, ir al paso 3

Paso 3. Definir los ejes para la gráficas que se desean obtener

```
>AXES  H 0 20 V -.5 2
```

Paso 4. Definir las variables que se quieren graficar

```
>PLOT Y
```

Paso 5. Definir las condiciones iniciales de las variables de estado

```
>INIT X1:0 X2:0
```

Paso 6. Definir el intervalo de simulación

```
>SIMU 0 2
```

c. Cálculo numérico y procesamiento de señales

El paquete MATLAB es un programa matemático interactivo potente cuyo principio básico es que no necesita dimensionamientos en las matrices al realizar operaciones numéricas y lógicas. Además, permite agregar comandos y funciones programadas en lenguaje Fortran y C. A continuación se presentan brevemente algunos de los problemas y conceptos que maneja el paquete y que sirven como base para realizar la práctica. En el anexo A3 se halla la lista completa de comandos.

Hecho 1: Considere un sistema lineal de ecuaciones:

$$Ax = b \tag{8}$$

con $A \in \mathbb{R}^{mxn}$, $b \in \mathbb{R}^m$ y $x \in \mathbb{R}^n$. Existe una solución x de (8) si y sólo si el vector b es un elemento del rango {A} o equivalentemente:

$$\text{rango}[A\,|\,b] = \text{rango}[A] \tag{9}$$

Hecho 2: Sean A, B, X y Q matrices $\in \mathbb{R}^{nxn}$. Existe una única solución de la ecuación de Lyapunov:

$$A\,P + P\,B = -Q \tag{10}$$

si y sólo si A y -B no tienen valores propios comunes.

Hecho 3: La solución del sistema lineal autónomo:

$$\dot{x}(t) = A\,x(t) \quad ; \; x(0)=x_o \tag{11}$$

con $x \in \mathbb{R}^n$ está dada por $x(t) = \phi(x_o, t) = \exp(At)\,x_o$.

Hecho 4:

Sean $a(s) = a_n s^n + a_{n-1} s^{n-1} + \ldots + a_o$ y $b(s) = b_m s^m + b_{m-1} s^{m-1} + \ldots + b_o$

polinomios con coeficientes reales, entonces los coeficientes c_i del producto

$$c(s) = a(s)b(s) \tag{12}$$

están dados por:

$$c_i = \sum_{k=0}^{i} a_k . b_{i-k} \qquad (13)$$

Hecho 5: Teorema de Parseval. Considere las funciones complejas $G_1(\omega)$ y $G_2(j\omega)$ como tranformadas de Fourier de las señales $g_1(t)$ y $g_2(t)$ respectivamente, esto es:

$$G_i(\omega) = \int_{-\infty}^{\infty} g_i(t)\, e^{-j\omega t} dt \quad \text{para } i = 1,\ 2 \qquad (14)$$

Entonces, el producto de ambas señales en el tiempo y en la frecuencia están relacionados por:

$$W = \int_{-\infty}^{\infty} g_1(t)\ g_2(t)\ dt = \int_{-\infty}^{\infty} G_1(\omega)\ G_2(\omega)\ d\omega \qquad (15)$$

donde W se puede interpretar como la energía disipada del sistema [9]. En particular $|G_i(\omega)|^2$ se conoce como la densidad o espectro de energía.

IV. DESARROLLO DE LA PRÁCTICA

Parte 1.

Considere una planta cuyo modelo está dado por la función de transferencia:

$$G(s)=\frac{14.1}{s(.084s+1)} \qquad (16)$$

Diseñar un compensador G_c usando el CC (ver anexo A1), de tal manera que se tenga un error en estado permanente cero ante perturbaciones rampa, un margen de fase mayor a 40° y una ganancia en lazo cerrado menor a 1.4 para toda las frecuencias (esto es $\max|G_g(j\omega)| < 1.4$ para toda ω con G_g la función de transferencia en lazo cerrado). En caso de no poder satisfacer todas las especificaciones con un compensador G_c proponga alternativas.

Parte 2.

Considere la figura 2 donde se muestra un regulador de temperatura cuyo elemento de medición tiene una histéresis [7].

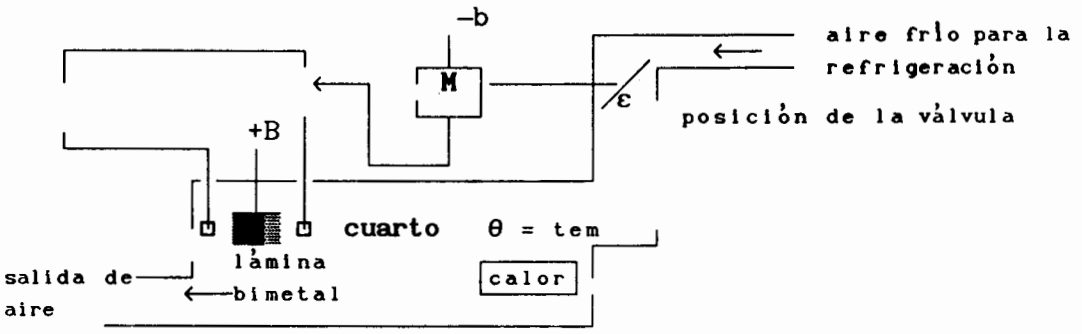


Fig. 2 Esquema del regulador de temperatura

El comportamiento del sistema después de varias simplificaciones se puede describir por medio de la ecuación:

$$MC\frac{d^2\theta}{dt^2} + S\alpha\frac{d\theta}{dt} = - K\,\dot{\varepsilon}(\theta) + \dot{f}(t) \qquad (17)$$

donde M es la masa del cuarto; C, el coeficiente de calor específico; S, la superficie del cuarto; α, el coeficiente de conducción de calor; θ, la temperatura del cuarto expresada como una desviación de alrededor de un punto de funcionamiento; $\varepsilon(\theta)$, la posición del elemento de acción y $f(t)$, la fuente de calor que se considera constante en el estudio. El actuador es del tipo todo o nada con histéresis y su comportamiento se representa en la figura 3.

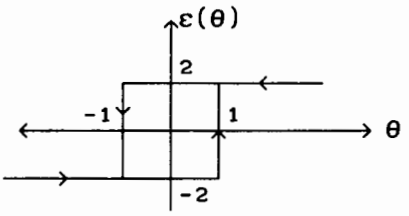


Fig. 3 Histéresis

Suponiendo los valores $MC/S\alpha$ = 1 min, $K/S\alpha$ = 1°C/rad y $f(t)/S\alpha$ = 0, obtenga la evolución en el plano de fase de (θ y $\dot{\theta}$) empleando el paquete de simulación SIMNON para 5 juegos de condiciones iniciales $\theta(0)$ y $\dot{\theta}(0)$.

Parte 3.

Usando el paquete MATLAB, realizar las tareas que se indican a continuación:

1) Proponer un juego de matrices (A,b), al menos de dimensión tres, que satisfaga el hecho 1 y verificar numéricamente que éste se cumple. Reemplazar b=0 y encontrar una solución; en caso de no hacerlo, justifique porqué.

2) Proponer dos matrices A, una con valores propios negativos y la otra, con al menos uno positivo, suponer $B = A^T$ y resolver la ecuación de Lyapunov, en ambos casos, con la matriz Q igual a la identidad. Finalmente, verificar si las matrices solución son definidas positivas.

3) Seleccionar un sistema estable autónomo cuyo modelo (ec. 11) sea de orden tres, y determinar la evolución del estado x(t) para dos juegos de condiciones iniciales usando la matriz de transición exp(At).

4) Escoger dos polinomios a(s) y b(s) con n >3 y m >4 y calcular el producto c(s) = a(s)b(s) utilizando el hecho 4. Verificar que las raíces de c(s) son iguales a las de a(s) y b(s).

5) Encontrar la Transformada de Fourier de la señal

$$f(t) = \begin{cases} A & \text{para } 0 < t < T_p \\ 0 & \text{para } T_p < t < \infty \end{cases} \qquad (18)$$

y graficar $|F(\omega)|^2$. Variar el valor de T_p y analizar su efecto en el espectro en frecuencia.

V. REPORTE DE RESULTADOS

Para dar por aprobada la práctica se requiere entregar:

Parte 1.

1.1 Las gráficas de Bode y de Nyquist del sistema compensado en lazo abierto indicando sobre las gráficas el margen de fase del sistema.

1.2 Las respuesta en frecuencia en lazo cerrado donde se observe que no se viola la restricción de máxima ganancia.

1.3 La justificación del tipo de compensador empleado.

1.4 Las limitaciones del método de diseño y alternativas para mejorarlo.

Parte 2.

2.1 La evolución de la temperatura y su derivada en función del tiempo.

2.2 La evolución de temperatura θ y su derivada en el plano de fase.

2.3 El error en estado permanente de la temperatura del cuarto.

2.4 Una proposición para reducir el error obtenido en el punto 2.3.

Parte 3.

En todos los incisos de la parte 3 se necesita la secuencia de comandos efectuados para obtener los resultados .Además,

3.1 Los valores propuestos de A, b y la solución x; en caso de no tener solución el sistema justificar porqué.

3.2 Las matrices elegidas A, sus correspondientes valores propios y las soluciones de la ec. de Lyapunov P mencionando si son definidas positivas.

3.3 La matriz A propuesta y las gráficas de x(t).

3.4 Los polinomios seleccionados y el producto.

3.5 La gráfica de $|F(\omega)|^2$ y el análisis del efecto de T_p en su espectro.

VI. BIBLIOGRAFÍA

1. MELSA J.& D. Schultz. Linear control systems. Mc. Graw Hill, 1969.

2. KUO B. Automatic control systems. Prentice Hall, 1975.

3. FRANKLIN G., J Powell & A. Emami-naeini. Feedback control dynamic systems. Addison Wesley, 1986.

4. STOER J. & R Bulirsch. Introduction to numerical analysis. Springer.

5. Program CC user's guide. System Technologies, INC.

6. ELMQVIST H. SIMNON user's guide. Dep. of Automatic Control Lund Institute of Technology, LUTFD2/TFRT-3091)/1975, Suecia.

7. GIBSON J..Nonlinear automatic control. Mc. Graw Hill 1963.

8. MATLAB user's guide. The Math Works Inc. 1987.

9. DESOER C. & E. Kuh. Basic circuit theory. Mc. Graw Hill 1969.

PRACTICA 7

CONTROLADORES PID DISCRETOS

I. OBJETIVOS

- Introducir al estudiante al estudio de los controladores PID discretos.
- Simular digitalmente diferentes variantes de los controladores PID discretos.
- Implementar en forma real un sistema de control discreto para un proceso térmico.

II. INTRODUCCION

Aunque en la actualidad (debido al gran desarrollo en la tecnología de construcción de circuitos integrados) existen una gran variedad de controladores a nivel industrial, es innegable que los controladores PID siguen siendo los más utilizados para resolver la mayoría de los problemas de control existentes.

En esta práctica se presenta la estructura, uso y funcionamiento de este tipo de controladores en su versión discreta, esto es, implantados a través de un programa almacenado en una computadora digital.

El formato que se sigue es: Primero se estudia en un nivel teórico el equivalente discreto de los controladores P, PI y PID analógicos. Posteriormente se realizan simulaciones digitales de algunas variantes de éstos para, finalmente, implantar un sistema de control real. Este último, compuesto por una computadora digital, un sistema de adquisición de datos y un proceso térmico, en donde el objetivo de control es el de mantener la temperatura del aire circulando por medio del proceso en un valor determinado.

III. PRELIMINARES

Sea

G(z)	la función de transferencia discreta del controlador
e(kh)	la secuencia de error
u(kh)	la secuencia de control
s	la variable de Laplace
U(s)	transformada de Laplace de la señal de control
E(s)	transformada de Laplace de la señal de error
h	periodo de muestreo

a. Control proporcional (P)

La función de transferencia de un controlador proporcional en el dominio de Laplace está dada por:

$$G(s)=\frac{U(s)}{E(s)}=K \qquad (4.1)$$

donde K es la ganancia proporcional y de donde se obtiene:

$$G(z)=\frac{U(z)}{E(z)}=K \qquad (4.2)$$

Obteniendo la ecuación en diferencias de expresión anterior se tiene que:

$$u(kh)=Ke(kh) \qquad (4.3)$$

b. Control proporcional-integral (PI)

Un control PI en el dominio de Laplace está dado por:

$$G(s)=\frac{U(s)}{E(s)}=K\left(1+\frac{1}{T_i s}\right) \qquad (4.4)$$

donde T_i es una constante denominada tiempo de integración.

Utilizando el método de "diferenciación hacia adelante" se tiene que:

$$G(z)=\frac{U(z)}{E(z)}=\frac{KT_I(1-z^{-1})+Khz^{-1}}{T_I(1-z^{-1})} \qquad (4.7)$$

a partir de la cual se obtiene que:

$$u(kh)=u(kh-h)+Ke(kh)+K((h/T_I)-1)e(kh-h) \qquad (4.8)$$

c. Control proporcional-integral-derivativo (PID)

Este tipo de control agrupa las ventajas de los tres tipos de control básicos. Su función de transferencia en el dominio de Laplace es:

$$G(s)=\frac{U(s)}{E(s)}=K\left(1+\frac{1}{T_I s}+\frac{T_D s}{1+(T_D/N)s}\right) \qquad (4.9)$$

donde T_D es la constante derivativa y:

$$\frac{1}{1+(T_D/N)s}$$

es un filtro estable.

Empleando el método de "diferenciación hacia atrás" se obtiene que:

$$G(z)_P=K \qquad (4.10a)$$

$$G(z)_I=\alpha/(1-z^{-1}) \qquad (4.10b)$$

$$G(z)_D=(1-z^{-1})/\beta \qquad (4.10c)$$

$$F(z)=\gamma+1/1+\gamma z^{-1} \qquad (4.10d)$$

donde:

$$\alpha=Kh/T_I \qquad (4.11a)$$

$$1/\beta=KT_D/h \qquad (4.11b)$$

$$\gamma=(Nh/T_D)-1 \qquad (4.11c)$$

ESTRUCTURAS ALTERNATIVAS

Existen diferentes formas de implantar un controlador PID además de la convencional mostrada en la figura 1a. Dos de ellas se presentan en las figuras 1b y 1c.

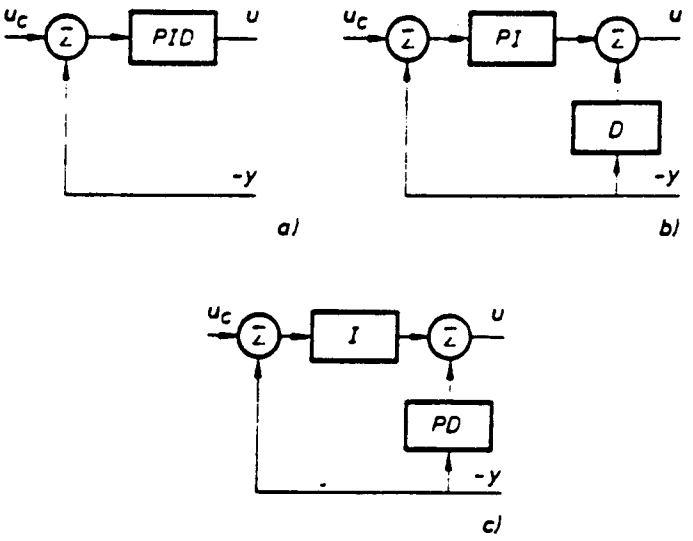


Fig. 1 Estructuras alternativas para PID.

En estas estructuras, la ley de control tiene la siguiente forma general:

$$U(z)=\frac{T(z)}{S(z)}R(z)-\frac{Q(z)}{S(z)}Y(z) \qquad (4.12)$$

donde:

$S(z)=(1-z^{-1})(1+\gamma z^{-1})$

$Q(z)=q_0+q_1z^{-1}+q_2z^{-2}$

$T(z)$, depende de la estructura empleada

Ejercicio. Encuentre el valor de $T(z)$, $Q(z)$ *y* $S(z)$ *para las figuras mostradas.*

Antitransformando las ecuaciones encontradas en el ejercicio anterior se obtiene que:

• Para la figura 1a:

$$u(kh)=\gamma u(kh-2h)-(\gamma-1)u(kh-h)+q_2[r(kh-2h)-y(kh-2h)]+$$
$$+q_1[r(kh-h)-y(kh-h)]+q_0[r(kh)-y(kh)] \qquad (4.13a)$$

•Para la figura 1b:

$$u(kh)=\gamma u(kh-2h)-(\gamma-1)u(kh-h)+\alpha(\gamma+1)r(kh)-$$
$$-q_2y(kh-2h)-q_1y(kh-h)-q_0y(kh) \qquad (4.13b)$$

Ejercicio. Obtenga la ecuación en diferencias u(kh) *para la figura* 1c.

d. Sintonización

METODO DE RESPUESTA TRANSITORIA

Este método propuesto por Ziegler y Nichols (1942) establece que tomando la respuesta transitoria de la planta en lazo abierto a una función escalón unitario, los parámetros para el PID se obtienen como:

	K	T_i	T_d
P	1/RL		
PI	0.9/RL	3L	
PID	1.2/RL	2L	0.5L

donde R y L son como se muestran en al figura 2:

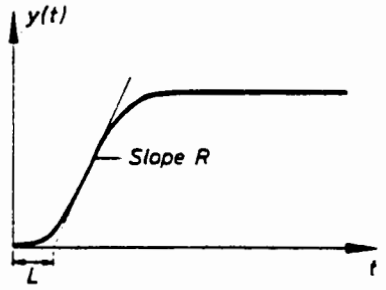


Fig. 2 Sintonización de PID

IV. SIMULACION DIGITAL

IV.a Objetivo

- Evaluar el comportamiento de las diferentes estructuras de controladores estudiados en los preliminares con base en simulaciones digitales.

IV.b Equipo

- Una microcomputadora PC-compatible
- Un paquete de simulación SIMNON

IV.c Desarrollo

1) Implemente en SIMNON un bloque analógico el cual describa la siguiente planta de 2do. orden:

$$G(s)=\frac{1}{s^2+s+1}$$

y obtenga de su respuesta a un escalón unitario los parametros R y L.

2) Con los parámetros sintonizados y basados en el paso anterior, implante los bloques digitales para conseguir los controladores P, PI y PID estudiados en los preliminares.

3) Forme un sistema en lazo cerrado y observe la salida del sistema para cada uno de los controladores considerando como secuencia de referencia un escalón unitario.

V.CONTROL DE TEMPERATURA

V.a Objetivo

- Implantar un control de temperatura sobre un proceso térmico real con una computadora digital.

V.b Equipo

- 1 microcomputadora PC-compatible
- 1 proceso térmico PT326

- 1 equipo de adquisición de datos LABMASTER
- 1 paquete de control CON_PACK

V.c Desarrollo

1) Ejecute el programa CON_PACK y conecte el sistema térmico al adquisidor de datos LABMASTER.

2) Del menú principal, escoja "SELECCION DE CANALES DE CONVERSIÓN" y seleccione el canal 0 para conversión A/D y el canal 0 para conversión D/A.

3) Del menú principal, elija "SELECCIÓN DE PANTALLAS DE GRAFICACIÓN" y seleccione grafica contra tiempo.

4) Del menú principal elija "SELECCIÓN DE CONTROLADORES".

5) Con el controlador P seleccionado, obtenga la respuesta del sistema para: $1<K<5$ y $0.5<f<2$ con una señal cuadrada como referencia.

6) Con el controlador PI seleccionado y en la opción de proporcionar los parámetros K y T_i, obtenga la respuesta del sistema a una señal cuadrada como referencia con $u(0)=e(0)=0$. Los valores de K y T_i deben variarse de tal forma que:

$$0.5<K<1$$
$$0.1<T_i<0.5$$

a una frecuencia de muestreo de 10 Hz.

NOTA. OBTENGA IMPRESIONES DE CADA UNA DE LAS GRÁFICAS GENERADAS.

VI. CUESTIONARIO

VI.a Simulación

1. ¿Cuál es la diferencia observada en la salida del sistema para los diferentes controladores?. Justifique estas diferencias.

VI.b Control de temperatura

1. Explique las diferencias en la respuesta del sistema para cada uno de los controladores implementados.

2. ¿Por qué no se usó un PID en el control de este proceso?

VII. BIBLIOGRAFIA

1. ASTRÖM & Wittenmark. Computer controlled systems: Theory and design. (1984). Prentice Hall.

2. FRANKLIN & Powell. Digital control of dynamics systems. (1980).

3. SCIENTIFIC Solutions Inc. LABMASTER. Installation manual and user's guide. Scientific Solutions Inc.

4. FEEDBACK Instruments. Process trainer PT326 instructional manual D326. Feedback Instruments Ltd.

5. PINET y Pinet. Diseño e implementación de un paquete de controladores digitales en una microcomputadora. Tesis de Licenciatura, Facultad de Ingeniería, UNAM. 1989.

PRACTICA 8

CONTROL DIGITAL DEL PROCESO TERMICO POR UBICACION DE POLOS

BIBLIOTECA DE LA UNIVERSIDAD NACIONAL DE MEXICO
DEPFI

I. OBJETIVOS

- Diseñar controladores digitales por el método de ubicación de polos.
- Controlar el proceso térmico PT326 mediante una PC y la tarjeta LabMaster.

II. MATERIAL Y EQUIPO

- Un proceso térmico PT326
- Una PC compatible con IBM
- Una tarjeta de conversión LabMaster
- Un osciloscopio
- Cables de conexión

III. ANTECEDENTES TEORICOS

El comportamiento dinámico de un sistema lineal invariante en el tiempo está determinado por los polos de la función de transferencia que describe el sistema. El método de diseño de controladores por ubicación de polos, como su nombre indica, ubica los polos del sistema en malla cerrada a posiciones deseadas que estén, a su vez, determinadas de acuerdo con una aplicación específica.

a. Preliminares sobre polinomios

Sea:

$$A(z) = a_o z^n + a_1 z^{n-1} + \ldots + a_n \qquad a_o \neq 0$$

un polinomio en z, con z un número complejo. El *orden* de A(z), denotado por degA, es n.

Si $a_o=1$, se dice que el polinomio A(z) es mónico.

Si los ceros de A(z) (las raíces de A(z)=0) se encuentran dentro del círculo unitario, se dice que A(z) es estable.

Se dice que dos polinomios A(z), B(z) son coprimos, si A(z) y B(z) no tienen factores en común.

Hecho: (Solución de la ecuación Bezout)

Dados A(z), B(z), C(z) polinomios en z, la ecuación Bezout

$$A(z)X(z) + B(z)Y(z) = C(z)$$

con degX<degB, o degY<degA, tiene solución única en polinomios X(z) y Y(z) si A(z) y B(z) son coprimos.

b. Diseño del controlador digital por ubicación de polos

1.Objetivo

Diseñar un controlador tal que el sistema en malla cerrada se comporte igual que un modelo de referencia.

2.Planta

Considérese una planta lineal invariante en el tiempo (LIT) (Fig.1) dada por:

$$Y(z) = G_p(z)U(z) + D(z)$$

$$G_p(z) = \frac{B(z)}{A(z)}$$

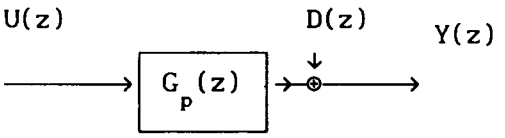


Fig. 1

Los polinomios A(z) y B(z) son coprimos con degA ≥deg B, y:

U(z) - señal de entrada a la planta

Y(z) - señal de salida de la planta

D(z) - ruidos en la salida

3. Modelo de Referencia

El comportamiento deseado del sistema en malla cerrada está especificado a través de un modelo de referencia,

$$G_m(z) = \frac{B_m(z)}{A_m(z)}$$

donde $A_m(z)$ y $B_m(z)$ son polinomios coprimos con el orden relativo:

$$\deg A_m - \deg B_m \geq \deg A - \deg B$$

4. Controlador

Para cumplir los objetivos de diseño, se considera el siguiente esquema de control (Fig. 2):

$$U(z) = \frac{T(z)}{R(z)} U_c(z) - \frac{S(z)}{R(z)} Y(z)$$

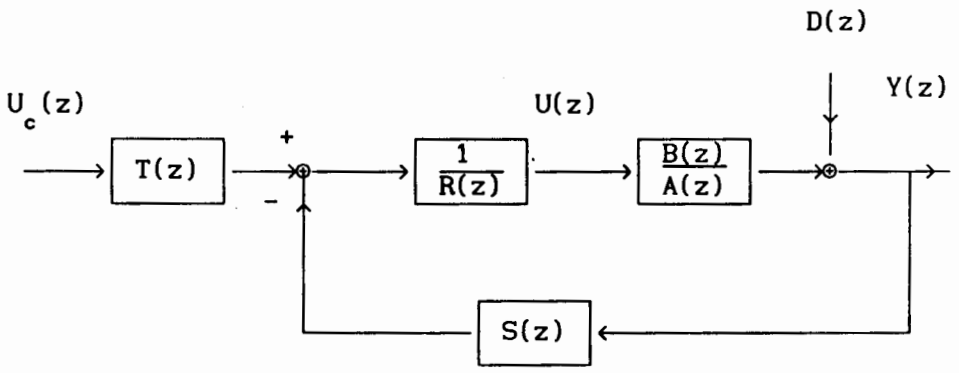


Fig. 2

donde $U_c(z)$ es la señal de comando (referencia), y T(z), R(z) S(z) son polinomios a determinar.

5. Observador

Se efectúa el rechazo de ruidos en la salida D(z) de forma implícita al especificar el modelo y un observador cuya dinámica está dada por el polinomio $A_o(z)$.

6. Procedimiento de diseño

Datos: La planta: A(z), B(z) coprimos, con A(z) mónico.

Factorización: Factorizar $B(z) = B^-(z)B^+(z)$ con $B^+(z)$ mónico y estable.

Especificaciones: El modelo de referencia: $B_m(z)$, $A_m(z)$ con $B^-(z)$ como factor de $B_m(z)$ y $A_m(z)$ mónico y estable.
El observador: $A_o(z)$ mónico y estable.

Condición de compatibilidad:

$$degA_m - degB_m \geq degA - degB$$
$$degA_o \geq 2degA - degA_m - degB^+ - 1.$$

Paso 1

Encontrar $R_1(z)$, S(z) resolviendo la ec. Bezout.

$$A(z)R_1(z) + B^-(z)S(z) = A_o(z)A_m(z)$$

con $R_1(z)$ mónico, $degR_1 = degA_o + degA_m - degA$, y $degS = degA - 1$.

Paso 2

Formar $R(z) = R_1(z)B^+(z)$, $T(z) = A_o(z)B_m(z)/B^-(z)$.

Paso 3

La ley de control está dada por:

$$U(z) = \frac{T(z)}{R(z)} U_c(z) - \frac{S(z)}{R(z)} Y(z)$$

7. Aspectos Prácticos

7.1 Selección del modelo de referencia

Se puede seleccionar el modelo de referencia como:

$$G_m(z) = K_m \frac{B^-(z)}{z^m P(z)}$$

donde:

$$K_m = \frac{P(z)}{B^-(z)}\Big|_{z=1}$$

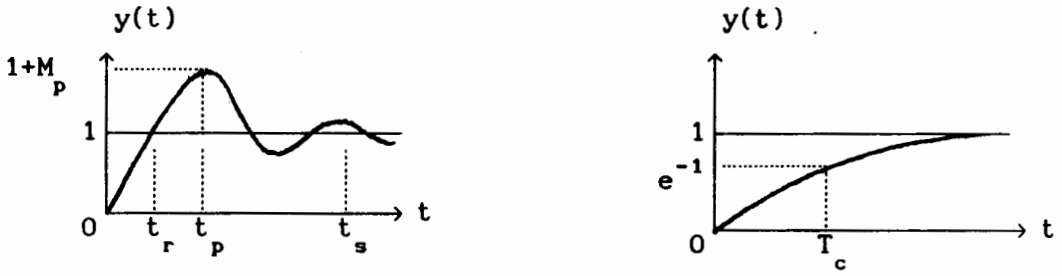
$$m \geq \deg A - \deg B^+ - \deg P.$$

Observaciones:

(1). La ganancia estática de $G_m(z)$ es unitaria, i.e.,

$$G_m(z)\Big|_{z=1} = 1$$

(2). Los polos de $G_m(z)$ están dados por $P(z)$ y los m restantes en $z=0$. Se puede escoger $P(z)$ de la siguiente manera:



a. segundo orden

b. primer orden

Fig. 3

caso a

(segundo orden) se escoge el polinomio P(z) como:

$$P(z) = z^2 + p_1 z + p_2$$

con coeficientes dados por:

$$p_1 = -2e^{-\xi\omega_n T}\cos(\omega_n T\sqrt{1-\xi^2})$$

$$p_2 = e^{-2\xi\omega_n T}$$

donde:

T = tiempo de muestreo

ω_n = frecuencia natural deseada

ξ = coeficiente de amortiguamiento deseado

Los parámetros ω_n y ξ y el comportamiento dinámico de un sistema están relacionados por las siguientes expresiones (Fig.3a):

sobrepaso:

$$M_p = e^{-\frac{\xi\pi}{\sqrt{1-\xi^2}}}$$

tiempo de asentamiento:

$$t_s = \frac{4.6}{\xi\omega_n}$$

Tiempo de levantamiento:

$$t_r = \frac{\pi - \cos^{-1}\xi}{\omega_n\sqrt{1-\xi^2}}$$

caso b

(primer orden) se escoge el polinomio P(z) como:

$$P(z) = z-p$$

con el coeficiente dado por:

$$p = e^{-\frac{T}{T_c}}$$

donde:

T = tiempo de muestreo

T_c = constante de tiempo deseada (Fig. 3b)

7.2 Selección del tiempo de muestreo.

Una regla heurística es:

$T = t_r/N$ para sistemas en malla cerrada cuya respuesta al escalón unitario dada en la Fig. 3a

$T = T_c/N$ para sistemas en malla cerrada cuyo respuesta al escalón unitario dada en la Fig. 3b

con N entre 2-4.

III. DESARROLLO

1) Obtener el modelo del proceso térmico

Se puede modelar el proceso térmico por un sistema de primer orden con retardo,

$$Y(s) = G(s)U(s), \qquad G_p(s) = \frac{K\, e^{-ds}}{T_c s+1}$$

donde: Y(s) = Temperatura (°C) de salida

U(s) = Voltaje (Volt) de entrada

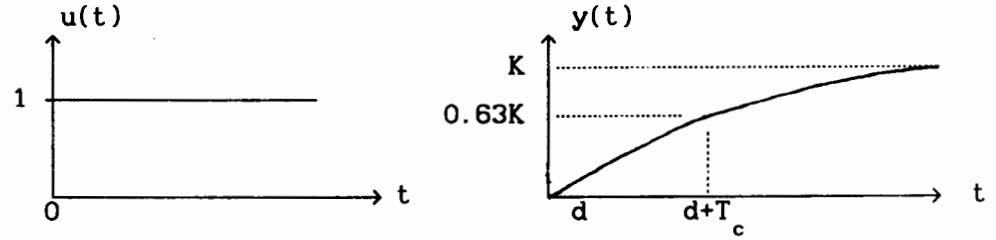
K = ganancia estática del proceso (°C/Volt)

T_c = constante de tiempo (seg)

d = retardo de entrada (seg.)

Se consiguen los valores de K, T_c, d mediante experimentos en malla abierta como se muestra en las siguientes gráficas.

Fig. 4



(a) entrada escalón (b) respuesta al escalón

Fig. 5

2) Obtener el modelo discreto de la planta.

Se obtiene el modelo discreto (Fig. 6) de acuerdo con los siguientes pasos:

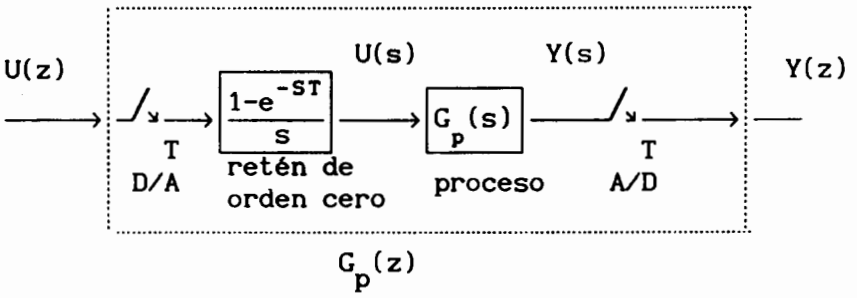


Fig. 6

a) Seleccionar el tiempo de muestreo T.

b) Supone que $\alpha = \ell T - mT$, con $0 \leq m < 1$, entonces:

$$G_p(z) = \begin{cases} \dfrac{K\left[1 - e^{-\frac{T}{T_c}}\right]}{z^{\ell}\left[z - e^{-\frac{T}{T_c}}\right]} & \text{para } m=0 \\ \dfrac{K_1\ (z+\alpha)}{z^{\ell}\left[z - e^{-\frac{T}{T_c}}\right]} & \text{para } 0<m<1 \end{cases}$$

con:

$$\alpha = \frac{e^{-m\frac{T}{T_c}} - e^{-\frac{T}{T_c}}}{1 - e^{-m\frac{T}{T_c}}}$$

$$K_1 = K\left[1 - e^{-m\frac{T}{T_c}}\right]$$

3) Diseñar un controlador digital por ubicación de polos.

4) Implantar el sistema de control mediante una PC, una tarjeta de conversión LabMaster.

El sistema de control digital se representa en la siguiente figura,

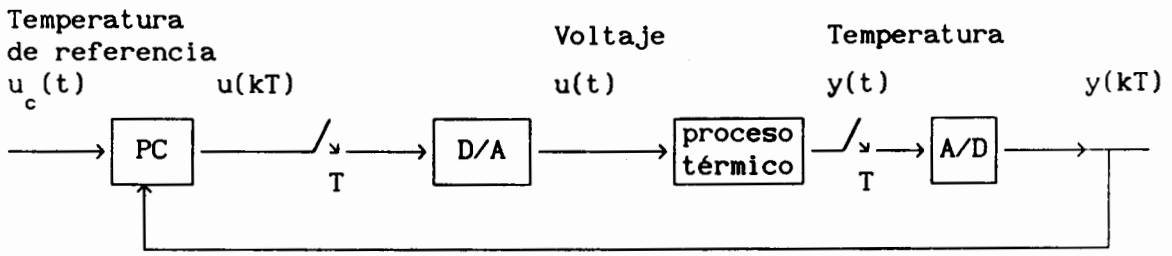


Fig. 7

5) Para diferentes señales de comando $u_c(t)$, observar el comportamiento de la temperatura de salida.

IV. RESULTADOS

Se debe incluir en el reporte:

1. Todos los desarrollos teóricos.

2. Gráficas de los resultados de los experimentos en malla abierta así como en mallas cerrada.

3. Observaciones y conclusiones.

V. BIBLIOGRAFIA

1. ASTRÖM K.J. y B. Wittenmark. Computer controlled systems, theory and design. Prentice Hall, 1984. Cap. 10.

2. FRANKLIN G. and J.D. Powell. Digital control of dynamic systems. Addison Wesley, 1986. Apéndice A.

PRACTICA 9

ESTABILIZACION DE UNA PLANTA INESTABLE CON UN CONTROL DE ESTRUCTURA VARIABLE

I. OBJETIVOS

- Entender el principio en el que se basan los Algoritmos de Control de Estructura Variable (ACEV).
- Ver la forma en que se implementa un ACEV para estabilizar una planta.
- Probar que es posible hacer estables sistemas mediante conmutaciones lógicas de sus estructuras constituyentes aunque éstas no sean estables.
- Verificar que es posible imponer al vector de estado una trayectoria predeterminada denominada Modo Deslizante (MD), por medio de modificaciones en la estructura de la planta.
- Probar la robustez de los ACEV para variaciones en los parámetros del sistema.

II. MATERIAL Y EQUIPO

- Una computadora analógica EAI-2000
- Un panel de programación o alambrado de circuitos para la EAI-2000
- Cables de alambrado de la EAI-2000
- Una unidad de monitoreo de señales para la EAI-200 o un osciloscopio de 2 canales
- Una unidad de graficación EAI 1135
- Una plumilla para la graficadora EAI 1135 o plumín
- Hojas de papel para la graficadora

III. ANTECEDENTES TEORICOS

Dentro del campo de estudio de los procesos industriales se encuentran una diversidad enorme de sistemas de control que, con el paso del tiempo, se vuelven cada vez más complejos. Esto ocurre como consecuencia de los continuos avances tecnológicos y de las crecientes

necesidades de las distintas industrias que luchan por economizar, aprovechar al máximo sus materias primas, automatizar sus procesos de producción y los problemas técnicos que ello implica. De esta manera, dentro del estudio de los sistemas de control podemos hablar de diseño de controladores lineales lo cual, sin embargo, en la práctica resulta absurdo pues en la realidad los procesos industriales presentan distintos tipos de alinealidades típicas entre las que podemos citar: la histéresis, bandas muertas, saturación, etc.

Los procesos industriales son operados dentro de amplios regímenes para los que se espera satisfacer un buen desempeño de los controladores. A pesar de esto, el diseño de un controlador convencional para una planta de tipo no lineal no siempre asegura el buen funcionamiento del sistema para los amplios rangos de funcionamiento a los que se hacía referencia. Por esta razón, se reconoce la necesidad de utilizar algoritmos de control más confiables que puedan trascender dificultades, como podrían ser los efectos no lineales, no estacionarios[1] y de ruido en el proceso o planta.

En la actualidad existen una gran variedad de algoritmos de control entre los que podemos mencionar los óptimos, adaptables, frecuenciales, etc. Sin embargo, haremos mención a un tipo de algoritmo que ha sido investigado principalmente en la Unión Soviética y que tiene un control adecuado sobre plantas no estacionarias donde el ambiente de operación es ruidoso. A este tipo de algoritmo se le conoce como : Algoritmo de Estructura Variable y se deriva de lo que se conoce como Teoría de los Sistemas de Estructura Variable que, en forma abreviada, llamaremos SEV.

La eficiencia de controladores de estructura variable radica en la sencillez de su implantación pues se trata de un simple conmutador que

1 Se refiere al hecho de que una planta no sea estacionaria, es decir, que no tiende a un punto de operacion estacionario, o bien, que cambia con el tiempo.

conecta a dos controladores proporcionales con distintas ganancias y donde la conmutación depende de un criterio de diseño (esto se explicará posteriormente). Además, el controlador puede ser desarrollado tanto analógica como digitalmente. En forma analógica se construye a partir de atenuadores de señales y relevadores u otro tipo de conmutador electrónico; y en forma digital, el algoritmo se implementa partiendo de un programa de computadora digital ya sea en lenguaje ensamblador, en lenguaje de alto nivel, o algún paquete de programación especializado, y de convertidores analógico-digitales y digital-analógicos.

Para comprender mejor un controlador de estructura variable se explicará qué se entiende por estructura y su relación con una planta o proceso. De esta manera, primero se partirá del hecho de que una planta o proceso puede modelarse, y es común que en la Teoría de Control se emplee la *transformada de Laplace* como herramienta matemática para dicho propósito. Al modelo de la planta en el dominio de la transformada de Laplace se le conoce como *Función de transferencia* y representa una relación funcional matemática entre la salida y la entrada de un proceso. Esta representación matemática es un cociente de polinomios en una variable compleja denominada 's'.

Otro par de términos muy usado en la Teoría de Control es : *polos*, el cual indica las raíces del polinomio denominador de la función de transferencia; y, *ceros*, que alude a las raíces del polinomio numerador de la función de transferencia.

Ahora, procederemos a mostrar la función de transferencia con la que trabajaremos en el desarrollo de la práctica. Se trata de un sistema de segundo orden cuya ecuación o función de transferencia se muestra a continuación:

$$\frac{Y(s)}{U(s)} = \frac{\omega^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad (3.1)$$

en donde ζ, significa el coeficiente relativo de amortiguamiento; ω_n, la frecuencia natural de oscilación; Y(s), la señal de salida, mientras que, U(s), la señal de control o la entrada al sistema. Además, este sistema se conoce como de segundo orden porque su polinomio denominador es de grado dos. Así pues, el sistema de segundo orden tendrá dos raíces en su polinomio denominador y cero raíces en su polinomio numerador, por lo cual se dice que la planta tiene dos polos y ningún cero.

Por otra parte, si se grafica en un plano en el eje de las abcisas una cierta variable y en el eje de las ordenadas la derivada de esa misma variable, decimos que tal plano es un *plano de fase*. El plano de fase y la función de transferencia se relacionan íntimamente con la localización de los polos y ceros de una planta en el plano complejo 's'. De esta forma, podemos decir, que a distintos patrones de polos y ceros corresponden también, distintos planos de fase.

Con lo anterior, se dice entonces, que una misma función de transferencia puede tener distintos patrones de polos y ceros en el plano complejo 's', y, consecuentemente, distintos planos de fase dependiendo de los valores que las raíces tomen en la función de transferencia.

Con lo dicho, por lo tanto, se entiende que una estructura no es más que un patrón de polos y ceros en el plano complejo 's',o un plano de fase determinado por las raíces de la función de transferencia. De la observación anterior, se infiere que un controlador de estructura variable es un conmutador que proporciona dos patrones diferentes de polos y ceros que, de acuerdo a un criterio de diseño, establecerá en qué momento conmutará una estructura y en qué otro, conmutará a la otra.

El criterio de diseño consiste en establecer que existe una recta en el plano de fase que define una zona de convergencia la cual representa la tendencia hacia la estabilidad del sistema, es decir, la dirección en la que el error se hace más pequeño.

Antes de continuar con la descripción del CEV, a continuación se mostrarán algunas representaciones de distintas estructuras en el plano complejo 's', sus respectivos planos de fase, y el nombre común con el que se le conoce a la estructura.

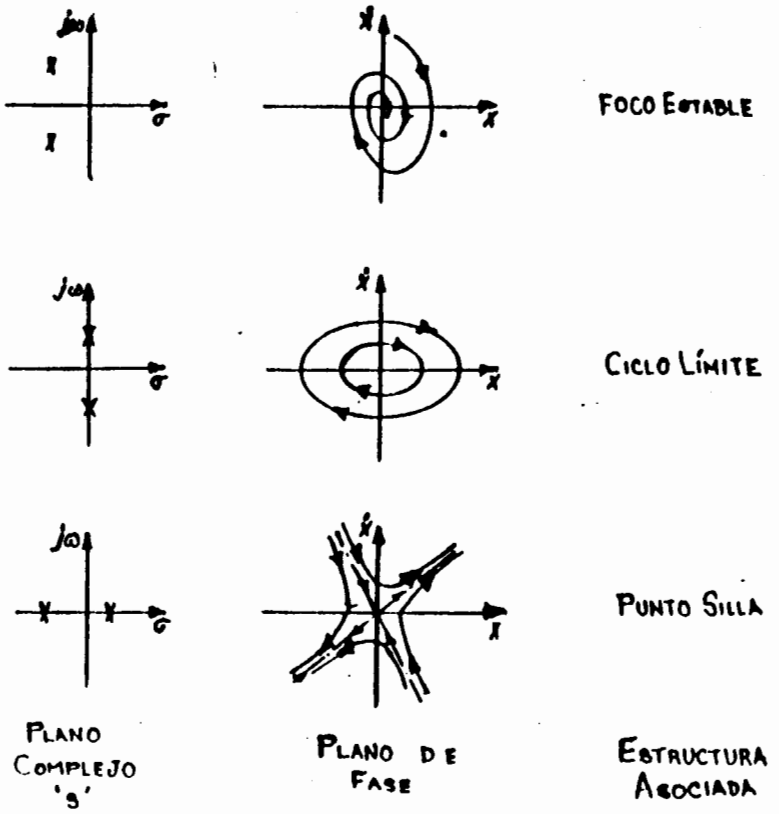


Fig. 1 Diferentes distribuciones de las singularidades en el plano complejo y su trayectoria asociada en el plano de fase.

Para el desarrollo del experimento se tomará en cuenta que las estructuras a utilizar serán: el ciclo límite, puesto que establece la condición de estabilidad marginal, y el punto silla, que es la condición de inestabilidad necesaria para que las trayectorias que se encuentran por encima de la recta de diseño, al ser sus derivadas negativas, converjan a la recta de diseño; ésta también es conocida como modo deslizante (MD). La recta de diseño es una ecuación del plano de fase y se expresa de la siguiente manera :

$$s = cx + \dot{x} = 0 \qquad (3.2)$$

en donde s, es la ecuación de la recta y no debe ser confundida con la variable compleja 's'; c, es la pendiente de la recta en el plano de fase que viene dada por la siguiente expresión al despejarla de la ecuación anterior :

$$- \quad c = \frac{\dot{x}}{x} \tag{3.3}$$

además $\dot{x}$ es la derivada de la variable x.

De esta forma, el problema del control se reduce a encontrar la pendiente c de la recta que cumpla la siguiente desigualdad :

$$\alpha > \psi_0 > \beta \tag{3.4}$$

en donde α es una de las ganancias del CEV y está asociada con la estructura de tipo "ciclo límite", mientras que, β, representa la otra ganancia que se asocia con la estructura "punto silla"; asimismo ψ_0 se calcula como sigue :

$$\psi_0 \equiv \frac{2 \zeta \omega_n c - c^2 - \omega_n^2}{\omega_n^2} \tag{3.5}$$

Para mayores detalles relativos al diseño del CEV consultar la bibliografía referente a los Sistemas de Estructura Variable (SEV), pues allí se hacen explicaciones paso a paso que en este espacio resulta imposible hacer.

Finalmente, se muestra un diagrama generalizado en el plano de fase (ver fig. 2) de las trayectorias de ambas estructuras cuando se tiene un CEV. Nótese la robustez que presenta dicho sistema que, en términos más comprensibles, se trata de un controlador sumamente inmune a las variaciones en los parámetros o condiciones de operación de la planta o proceso, así como también del ruido que pudiese presentarse.

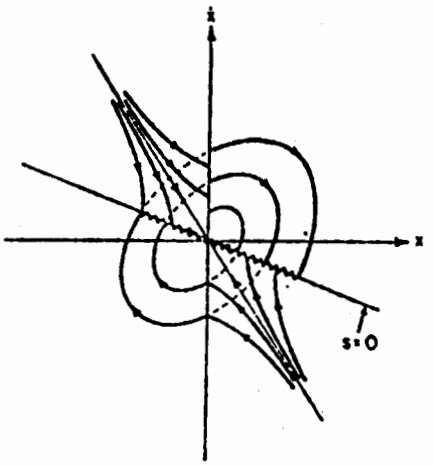


Fig. 2 Definición de las áreas de conmutación en el plano de fase

III. DESARROLLO

1) Para mostrar la ejecución del ACEV, se hará la simulación en computadora analógica de una planta inestable a la que se pretende estabilizar, realimentando la salida mediante un ACEV, y probar la robustez de este último para variaciones en los parámetros.

La planta inestable de segundo orden está descrita por las siguientes ecuaciones de estado :

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -4 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \qquad (3.6)$$

Podemos, entonces, añadir una realimentación negativa de la salida de la forma :

$$u = -\psi y \qquad (3.7)$$

Por otra parte, el MD se lleva a cabo a lo largo del hiperplano s=0, cuya ecuación resulta ser, para el caso de dos dimensiones :

$$\dot{x} = -c\, x \qquad (3.8)$$

Debe tenerse presente que las condiciones de existencia del MD para este caso particular serán :

$$\alpha > -c^2-4 > \beta \qquad (3.9)$$

donde c es la magnitud de la pendiente de la curva s=0 (1.3) y α y β son los parámetros requeridos de ψ que serán conmutados para obtener las trayectorias de ciclo límite y punto silla respectivamente.

Primeramente, se probarán los siguientes valores para los parámetros c= 2.333, α=1, β=-10. Puede evaluar la desigualdad 1.4 con estos valores para comprobar que las condiciones de existencia del MD se siguen cumpliendo.

2) Observe el circuito de la siguiente figura, diga si responde a las características que se necesitan para la simulación del sistema.

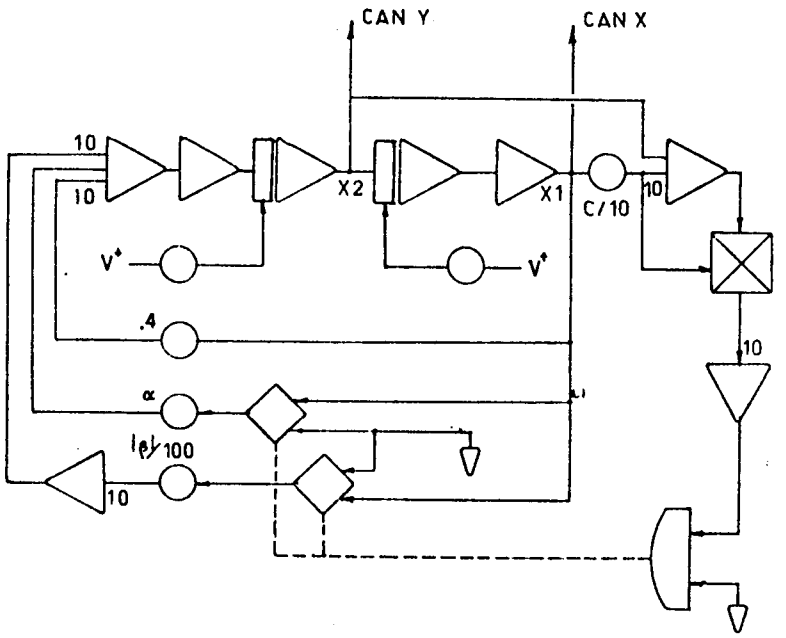


Fig. 3 Diagrama analógico de la planta y el ACEV.

Adviértase que el circuito está formado por dos bloques fundamentales, uno de ellos corresponde a la simulación de la planta de segundo orden inestable, y, el otro, al desarrollo del ACEV. Identifique ambos bloques en el diagrama y explique la manera en que llevan a cabo su función.

3) Alambre el circuito de la figura anterior en la computadora analógica. Revise bien las trayectorias alambradas. Se sugiere ir marcando las trayectorias en el diagrama del papel conforme se van efectuando las conexiones; además procure colocar identificadores en los símbolos sobre el diagrama que respondan a cada uno de los componentes usados en la computadora con el fin de localizarlos con mayor facilidad.

4) Coloque la señal $\dot{x} = x_2$(salida del primer integrador) en el canal 1 del osciloscopio (0 en el panel de la computadora) y la señal $x = x_1$ (salida invertida del segundo integrador) en el canal x del osciloscopio (canal de barrido).

5) Ajuste la perilla CHANNELy FUNCTION del osciloscopio en las posiciones 1 y SWEEP respectivamente, con esto logrará desplegar el plano de fase $\dot{x}$ - x del control del proceso que está simulando.

6) Proceda a encender la computadora y ajuste los coeficientes de los potenciómetros empleados, tome en cuenta los valores de los parámetros sugeridos en el punto 1 y auxíliese del diagrama del circuito.

El 0.4 asignado al coeficiente del potenciómetro de la trayectoria de realimentación directa es el parámetro de la planta a controlar, por lo que deberá permanecer con ese valor a lo largo de la práctica. Se sugiere utilizar uno de los DCA (C10-C37) disponibles en la computadora.

Los potenciómetros que se hallan controlando los valores de las condiciones iniciales realmente no requieren ser ajustados a un valor específico pues se estarán modificando a lo largo de toda la práctica. Se

recomienda emplear, en este caso, los potenciómetros de ajuste manual (C0-C7) del panel de control de la computadora, a fin de variar las condiciones iniciales con mayor facilidad ya que, de otra manera, habría que estar asignándoselos cada vez mediante comandos por la terminal.

7) Ajuste las constantes de tiempo de los dos integradores a una escala relativa de E - 1.

La escala maestra de integración podrá ser ajustada de tal forma que caiga en el rango de E0 a E3, aunque se sugiere utilizar la escala E2.

Se recomienda operar la computadora en modo repetitivo (comando AR <RETURN>) a fin de visualizar los cambios que sufre la trayectoria tras modificaciones en los parámetros, sin tener que estar enviando comandos de inicialización y operación cada vez que se realice alguna modificación.

8) Estando en el modo repetitivo, varíe arbitrariamente los coeficientes de los potenciómetros asignados a las condiciones iniciales sin perder de vista su ubicación en el plano de fase y haga lo siguiente:

a) Verifique si el sistema tiende a la estabilidad para los diversos valores de sus condiciones iniciales (la trayectoria debe tender al origen en este caso, pues nuestra señal de referencia es nula).
b) Identifique la recta $s = c\dot{x} + x = 0$ que representa al MD.
c) También, identifique las áreas de conmutación en el plano de fase.
d) Verifique que la conmutación se efectúa entre trayectorias de punto de silla y de ciclo límite.

9) Ahora, varíe el coeficiente del potenciómetro que controla el valor del parámetro c; ¿qué variaciones ocurren en la gráfica del plano de fase?

Mueva el potenciómetro y encuentre otra posición en la que se siga manteniendo el MD, extraiga los valores de los coeficientes de los potenciómetros, deduzca a partir de estos valores los parámetros c, α y β, y con ellos compruebe la condición de existencia del MD dada por la desigualdad 1.4.

Mueva nuevamente el potenciómetro y encuentre otra posición para la cual ya no se presente el MD y realice lo mismo que en el caso anterior.

10) A continuación trace las curvas del plano de fase para diversas condiciones iniciales en el graficador de la computadora. Para hacer esto, se conectan las señales $\dot{x} = x_2$ y $x = x_1$ a las terminales Y y X respectivamente del graficador que se halla en el panel de la computadora.

Seleccione las escalas adecuadas en el graficador y coloque éste en el modo local para tener control sobre el ascenso y descenso de la plumilla.

Use escalas maestras de integración pequeñas para el trazado, por ejemplo E1.

Opere la computadora en modo no repetitivo, es decir, pulse IC<RETURN> y OP<RETURN> independientemente.

Cambie las condiciones iniciales de tal forma que se puedan trazar trayectorias provenientes de diversas direcciones.

Una vez terminando de trazar un número conveniente de trayectorias, proceda a ejecutar lo siguiente :

a) Dibuje sobre los trazos obtenidos los ejes $\dot{x}$ - x que definen el plano de fase.
b) Dibuje la recta $s = c\dot{x} + x = 0$, que representa al MD.
c) Indique las áreas de conmutación en el plano de fase.

Se deberá conseguir una gráfica similar a la siguiente:

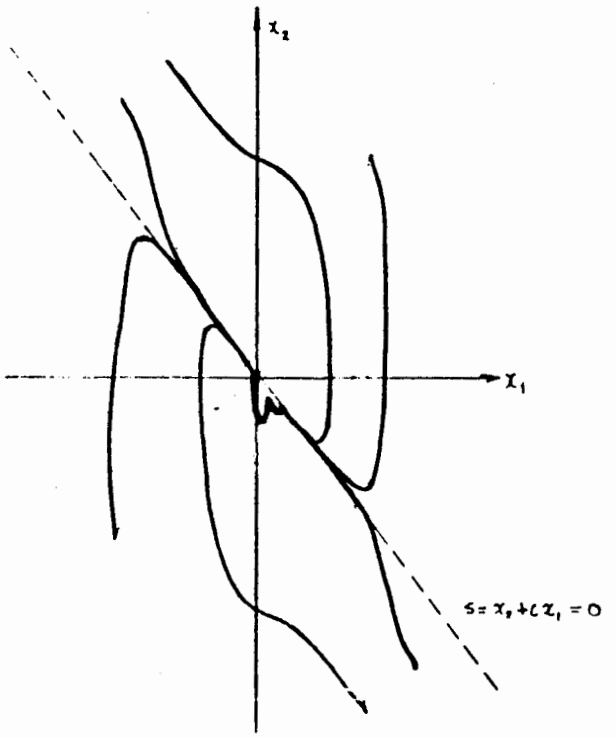


Fig. 4 Plano de fase de la planta con el ACEV para α=1, β=-10 y c=2.333

IV. BIBLIOGRAFIA

1. ESPINO Gutiérrez, Juan J. Realización de algoritmos de control de estructura variable en microprocesadores. Tesis C.I.E.A.-I.P.N. México, 1981.

2. KUO, Benjamín C. Sistemas automáticos de control. C.E.C.S.A., México, 1987.

3. OGATA, Katsuhiko. Ingeniería de control moderna. Prentice Hall. México, 1986.

4. ORTEGA Martínez, Romeo S. Sistemas de estructura variable, modos deslizantes y control. Apuntes D.E.P.F.I.- U.N.A.M. México, 1980.

5. UTKIN, Vadim I. Variable structure systems with sliding modes. Transactions on Automatic Control. Vol.AC-22, No.2, April 1977.

6. EAI 2000 analog reference handbook. Electronics Associates. New Jersey, 1980.

PRACTICA 10

CONTROL DE POSICION DE UN MOTOR DE CORRIENTE DIRECTA A TRAVES DE UN CONTROLADOR DE ESTRUCTURA VARIABLE

I. OBJETIVOS

- Implementar un control de posición convencional de un motor de corriente directa.
- Utilizar los conceptos manejados en la práctica 8 para identificar los elementos necesarios para diseñar de un Controlador de Estructura Variable (C.E.V.).
- Implementar un Controlador de Estructura Variable a través de la computadora analógica para realizar el control de posición del motor de corriente directa del punto 1, haciendo los ajustes convenientes.
- Observar el comportamiento del C.E.V. del motor de C.D. ante la variación de parámetros, como son, ganancia y carga, y, con base en ello, determinar la robustez del algoritmo.

II. MATERIAL

- Una computadora analógica EAI-2000
- Un panel de programación o alambrado de circuitos para la EAI-2000
- Cables de alambrado de la EAI-2000
- Una unidad de monitoreo de señales para la EAI-200 o un osciloscopio de dos canales
- Un servomecanismo o equipo marca Feedback MS150 con los siguientes elementos:

NOMBRE	CLAVE
a. Una Unidad Operacional	OU150A
b. Una Unidad Atenuadora	AU150B
c. Una Unidad Preamplificadora	PA150C
d. Un Servoamplificador	SA150D
e. Una Fuente de Alimentación	PS150E
f. Un Motor de C.D.	MT150F
g. Un Potenciómetro de Entrada	IP150H
h. Un Potenciómetro de Salida	OP150K

i. Una Unidad de Carga o Freno Magnético LU150L

j. Una base para colocar todos los dispositivos

k. Cables de doble banana para alambrado del MS150

- Caimanes para interconectar el circuito de la computadora analógica con el servomecanismo.

III. ANTECEDENTES TEORICOS

En la práctica 8 se estudiaron de manera elemental los principios básicos de los Sistemas de Estructura Variable (S.E.V.) que, como se vio, son controladores muy sofisticados en cuanto a su concepción, no así, en la manera de implantarlos. Los S.E.V. resultan ser controladores de tipo no lineal lo cual complica el análisis de estabilidad del sistema en conjunto, es decir, controlador y planta. Sin embargo, pese a este inconveniente, los C.E.V. resultan ser muy buenos controladores para plantas de segundo orden cuando se pueden implementar físicamente las dos estructuras de conmutación, o sea, su patrón de polos y ceros.

Por otra parte, un servomecanismo o servomotor puede ser modelado como una planta de segundo orden. Para detalles sobre modelado consultar [1],[2] y [3] de la bibliografía. Así pues, un sistema de segundo orden se modela en el dominio de la *transformada de Laplace* como :

$$\frac{Y(s)}{U(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad (4.1)$$

en donde ζ representa el coeficiente relativo de amortiguamiento, ω_n es la frecuencia natural de oscilación, Y(s) es la señal de salida, mientras que, U(s), es la señal de control o la entrada al sistema.

De esta manera, cuando se modela el sistema basado en un motor de CD., se encuentran a través de experimentación los parámetros ω_n y ζ para poder elaborar un diseño de la recta del Modo Deslizante (M.D.). Así, el servomecanismo del Sistema de Feedback MS150 se modela por medio de la

siguiente función de transferencia :

$$\frac{\theta}{e} = \frac{k\ 80}{s\ (s + 4.76)} \qquad (4.2)$$

en donde θ es la posición de entrada o señal de referencia; e, la señal de error producida como diferencia de la señal de entrada y la señal de salida del sistema; k, la ganancia que se le proporciona al servomecanismo y depende, básicamente, de la posición en la que coloquemos el potenciómetro de la unidad atenuadora más la señal de control que implementemos ya sea α, o bien, β. Identificando los parámetros de la ecuación (4.1) e igualándolos con los de la ecuación (4.2) se obtiene que: ω_n= 8.944 rad/seg. y ζ= 0.266, siempre y cuando k la consideremos aparte de la planta.

IV. DESARROLLO

1) Antes de comenzar a alambrar el circuito de control de posición, es primordial hacer la calibración adecuada de los dispositivos que se mencionan a continuación:

a. Unidad operacional. Localice la unidad operacional en el servomecanismo MS150, cuya clave es OU150A, o bien identifíquela con el nombre de OPERATIONAL UNIT. Una vez localizada esta unidad, se procederá a calibrarla. Para ello es necesario que la unidad sea debidamente polarizada, es decir, se debe conectar a la fuente del servomecanismo, con las terminales +15V, -15V y COM, una a una.

Ahora, para poder garantizar que la señal de salida del amplificador sea cero, hay que cortocircuitar las terminales numeradas 1,2 y 3 a tierra, o sea, a la terminal identificada como COM, ya sea de la fuente o del mismo amplificador. Para observar la señal de salida nula del amplificador se toma un cable de alguna de las salidas del amplificador, identificadas con el número 6, hacia el canal A del osciloscopio. Cabe hacer mención de que la tierra del osciloscopio debe coincidir con la

tierra del amplificador por lo cual la terminal COM de este último se conectará con la terminal de tierra del canal A del osciloscopio. Verifique que el osciloscopio esté seleccionando debidamente en el canal A y que se halle en la posición de medición de señales continuas y no alternas.

Asimismo, deberá verificar que la señal de tierra coincida con el cero que presenta el osciloscopio; para ello, simplemente presione el botón que está etiquetado como GND; si el haz de luz no coincide con el cero del osciloscopio en pantalla, entonces mueva la perilla de ajuste de vertical hasta lograrlo; si coincide el haz con el cero de la pantalla, no será necesario hacer ajustes.

Regresando, nuevamente, a la calibración de la unidad operacional, resta, simplemente, monitorear la señal de salida en el osciloscopio. Como esta señal deberá ser cero, por lo tanto debemos mover la perilla que se encuentra en la parte superior derecha del amplificador hasta hacerla coincidir con el cero de la pantalla del osciloscopio. Deberá tener cuidado de quitar la señal de GND del osciloscopio mientras realiza el monitoreo de la señal de salida de la unidad operacional.

b. Unidad preamplificadora. Localice entre los módulos del servomecanismo MS150, aquella unidad etiquetada como PA150C, cuyo nombre también aparece como PRE-AMP. UNIT. Para calibrarla se efectuará un procedimiento similar al llevado a cabo con la unidad operacional. La unidad preamplificadora también debe estar polarizada adecuadamente, como lo mencionamos en el caso anterior, e igualmente las entradas 1 y 2 de la unidad deben estar cortocircuitadas a tierra del sistema. El monitoreo de la señal de salida es un poco diferente con respecto al caso descrito en el punto anterior puesto que se trata de un amplificador diferencial.

Para monitorear adecuadamente el cero en el osciloscopio deberá tomar un par de cables de las salidas de la unidad, identificadas con los números 3 y 4, hacia las dos terminales del canal A del osciloscopio.

Para calibrar la unidad, ésta también cuenta con una perilla de ajuste. De esta manera, el cero de la señal de salida deberá coincidir con el de la pantalla del osciloscopio. Debemos reiterar que se deben tener ciertas precauciones al realizar la calibración, por lo que deberá atender a las recomendaciones que se dan en el punto A.

c. Potenciómetro de entrada. Localice entre los módulos del servomecanismo MS150, la unidad identificada con la clave IP150H, o bien, aquella que viene etiquetada como INPUT POT. UNIT. Para calibrar esta unidad es necesario colocar en sus dos terminales, identificadas con el número 1, +15V. y -15V. respectivamente, que provenienen de la fuente de alimentación del sistema.

Lo que resta, es observar en el osciloscopio el punto en el cual la señal es cero, moviendo la perilla del mismo potenciómetro y colocando, previamente, la señal de salida identificada en la unidad con el número 3 con el receptáculo de la señal viva del canal A del osciloscopio así como la tierra del sistema con la tierra del canal A del osciloscopio. Una vez que encuentre este punto, es necesario que observe si el cero de la señal coincide con el cero de la escala del potenciómetro. Si coinciden, no es necesario efectuar ajuste alguno, pero si no es así, entonces, será necesario que una persona mantenga sin mover y firmemente el eje de giro del potenciómetro mientras que, otra, realiza cuidadosamente el giro del disco que contiene la escala hasta que el cero de este último coincida con el que aparece en el osciloscopio.

Las recomendaciones para manejar el osciloscopio durante el proceso de calibración de las unidades mencionadas con anterioridad, se deben incluir en este caso.

d. Potenciómetro de salida. Ubique la unidad que tiene el nombre OUTPUT POTENTIOMETER UNIT o la que está identificada con la clave OP150K. El procedimiento de calibrado de esta unidad es el mismo que el llevado a cabo para el potenciómetro de entrada. Deberá tomar en consideración las

recomendaciones hechas con respecto al manejo del osciloscopio.

2) Alambrar el circuito de control de posición que se muestra en la figura 1. Cabe mencionar que en el dibujo no aparecen las conecciones de alimentación desde la fuente hacia el servoamplificador así como, tampoco, la alimentación desde el servoamplificador hacia el motor de C.D. Estas conexiones se efectúan a partir de un par de cables de ocho conductores que llevan los voltajes necesarios a los devanados del motor, excepto los de control. Una recomendación muy importante es que el sistema debe estar apagado para lo cual deberá verificar que el botón de encendido de la fuente de alimentación se encuentre en una posición diferente de ON, o, para garantizar la seguridad del equipo, desconecte la clavija de alimentación del sistema mientras efectúa su cableado.

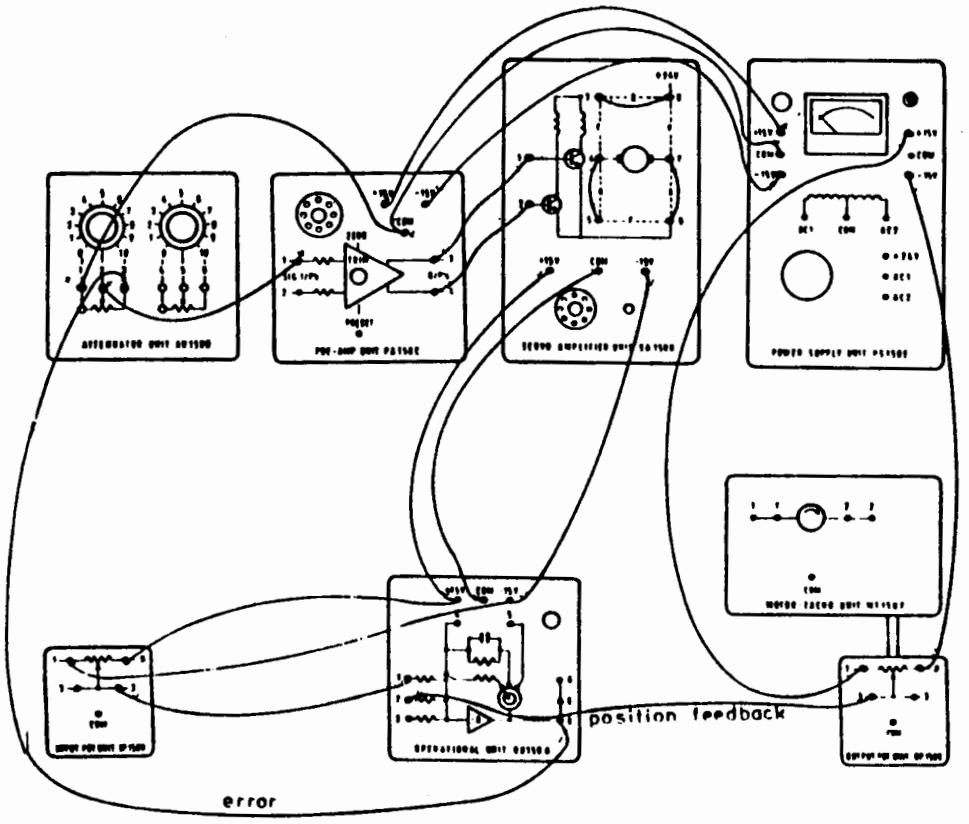


Fig. 1 Control de posición de lazo cerrado.

3) Revise cuidadosamente cada una de las conexiones que hizo asegurándose de que su instructor lo supervise.

4) Conecte la clavija de alimentación y encienda el sistema mediante el botón que se halla en la fuente de alimentación.

5) Comience a mover la perilla del potenciómetro de entrada a la posición que usted desee. Observe cómo el potenciómetro de salida trata de alcanzar el mismo punto. Es importante mencionar que si el potenciómetro que se encuentra en la unidad atenuadora está en la posición 0 usted no observará movimiento alguno del motor por lo que le debe dar algo de ganancia al sistema. Observe cómo es la respuesta del sistema cuando usted aumenta gradualmente la ganancia de éste.

6) Como opción, usted puede colocarle al disco de la carga, que está montado en el eje principal del motor, el freno magnético que forma parte del servomecanismo MS150 con el propósito de analizar el comportamiento del sistema cuando varía la carga en el motor.

7) Usted puede combinar en el experimento, también, la variación de ganancia con la variación de la carga del sistema. Describa sus observaciones.

8) Apague por unos instantes el sistema de control de posición y proceda a armar el circuito de la figura 2 en un panel de programación de la computadora analógica EAI-2000.

9) Encienda la computadora analógica. Recuerde que todos los comandos que ejecuta la computadora se le proporcionan a través de la terminal de control conocida como M.A.C.S. Realice la prueba de balance de los componentes de la computadora tecleando en la terminal BA <RETURN>. En la terminal se deberá observar que ningún indicador de alarma esté encendido, especialmente el identificado como BA. Si resultara algún problema de este tipo vuelva a ejecutar la prueba o, de persistir la anomalía, consulte a

su instructor para que él le indique de qué manera proceder.

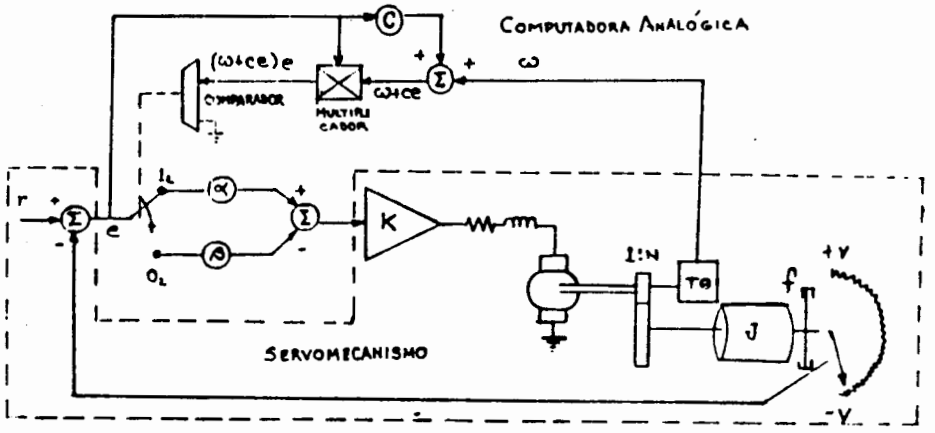


Fig. 2 Control de posición con ACEV

10) Una vez que se ha verificado el correcto funcionamiento de la computadora, se procede a inicializar los valores de los potenciómetros los cuales proporcionarán tanto las ganancias de las dos estructuras a conmutar como la pendiente de la recta de diseño del modo deslizante (M.D.). Para poder realizar ajustes fácilmente se recomienda el uso de los potenciómetros C00-C07. Cada uno de ellos deberá identificarse con el parámetro que representa : uno para C, que es la pendiente de la recta de diseño en el plano de fase; α, que es la ganancia que representa a la estructura de ciclo límite en el plano de fase; y β, que es la ganancia de la estructura de punto silla.

Cabe mencionar que primero debemos tener a la computadora en el modo inicial (esto no quiere decir que estamos en condiciones iniciales, sino en modo de reposo para la computadora) para lo que se introduce el comando QT <RETURN>. Para seleccionar el potenciómetro que se desea, se teclea Cnn <RETURN> donde 00 ≤ nn ≤ 07 y luego, únicamente se mueve el potenciómetro deseado hasta que se observe en la terminal el valor requerido.

Otra nota muy importante a recordar es el hecho de que cuando se resuelve un sistema no se debe estar en el modo Cnn. Para mayor

información consulte a su instructor, o bien, consulte el manual de usuario de la computadora.

11) En el circuito del dibujo, identifique las señales que serán interconectadas con el servomecanismo tales como la señal de retroalimentación tacométrica que proviene del servomecanismo; la señal de error, que viene de la unidad atenuadora del mismo servomecanismo; y la señal de control que proporcionará la computadora analógica hacia la entrada de la unidad preamplificadora del servomecanismo.

12) Proceda a hacer los cambios necesarios en el servomecanismo para implementar el C.E.V. Por una parte, una de las terminales del tacómetro se conecta a tierra del sistema, es decir, la terminal etiquetada como COM, mientras que, la otra, se conecta a un sumador de la computadora analógica. Por otro lado, la señal de error que sale de la unidad atenuadora se enviará al potenciómetro de la computadora analógica que representa la pendiente de la recta de diseño. Finalmente, la entrada de la unidad preamplificadora ahora estará conectada a la salida de la señal de control que proporciona un sumador de la computadora analógica.

Otro punto no menos importante es el hecho de que debemos hacer coincidir las señales de referencia de ambos sistemas, o sea, debemos cortocircuitar la tierra de la computadora analógica (que es cualquier punto donde se observe el color negro) con la tierra del servomecanismo (identificado con la etiqueta COM).

13) Antes de encender el servomecanismo, revise bien el alambrado del circuito y asegúrese de que su instructor lo supervise. Después proceda a encender el servomecanismo; coloque a la computadora en condiciones iniciales, a través del comando IC <RETURN>; mueva el potenciómetro de entrada del servomecanismo a la posición que usted desee e indique a la computadora que resuelva el sistema mediante el comando OP <RETURN>. También se le puede señalar a la computadora que ejecute en modo

repetitivo de tal manera que primero se ponga en estado de condiciones iniciales y que luego resuelva el sistema. Este comando se introduce como : AR <RETURN>.

14) Observe el comportamiento del sistema en condiciones normales. Puede repetir las condiciones del experimento mencionadas en los puntos 6. y 7. Observe este comportamiento y compárelo con el del servomecanismo sin C.E.V.

V. BIBLIOGRAFIA

1. ESPINO Gutiérrez, Juan J. Realización de algoritmos de control de estructura variable en microprocesadores. Tesis C.I.E.A.-I.P.N. México, 1981.

2. KUO, Benjamín C. Sistemas automáticos de control. C.E.C.S.A. México, 1987.

3. OGATA, Katsuhiko. Ingeniería de control moderna. Prentice Hall. México.

4. ORTEGA Martínez, Romeo. Sistemas de estructura variable. Modos deslizantes y control. Apuntes D.E.P.F.I.- U.N.A.M. México, 1980.

5. UTKIN, Vadim I. Variable structure systems with sliding modes. Transactions on Automatic Control. Vol.AC-22, No.2, April 1977.

6. EAI 2000 analog reference handbook. Electronics Associates. New Jersey, 1980.

7. FEEDBACK Instruments Ltd. Modular servo system MS 150. Book 1: DC, syncro, & basic experiments. Feedback Instruments Limited. England, 1980.

PRACTICA 11

INTRODUCCION AL ROBOT MANIPULADOR

I. OBJETIVOS

- Identificar por nombre los elementos más importantes de un sistema de robot.
- Inspeccionar las conexiones mecánicas y eléctricas de un sistema de robot.
- Identificar por nombre todas las partes del brazo de robot XR-3.
- Identificar los motores, articulaciones y especificaciones para cada eje de movimiento del brazo.
- Conocer la función de los switches, conectores, terminales y luces sobre el controlador Mark III del robot.
- Ejecutar tareas sencillas con el brazo del robot usando el "teach pendant" o botonera de enseñanza.

II. MATERIAL Y EQUIPO

- Brazo de robot XR3
- Pinza estandar
- Controlador Mark III
- "Teach pendant" del XR3
- Herramientas manuales básicas
- IBM PC/XT
- Base de aluminio

III. INTRODUCCION

Un sistema de robot industrial está formado por los siguientes elementos: Un brazo mecánico, un controlador con base en una computadora, una herramienta en el extremo del brazo, un "teach pendant" (botonera de enseñanza) y/o una terminal de computadora, un dispositivo de almacenamiento para archivar programas y tareas, y una fuente de poder externa. Ver fig. 1 donde se muestra un diagrama a bloques de estos

elementos.

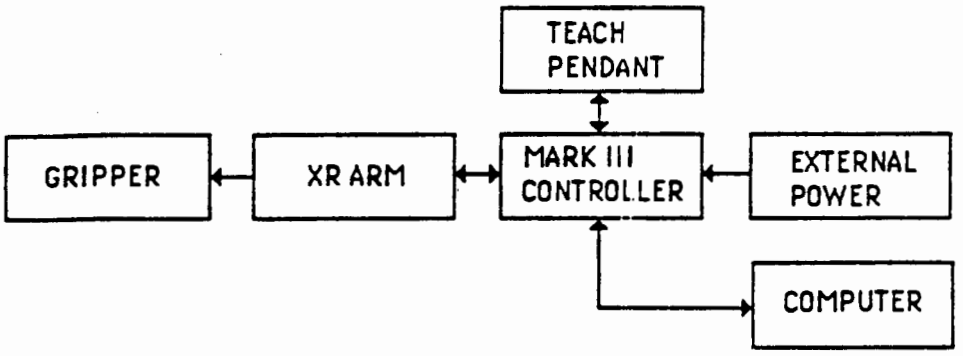


Fig. 1 Diagrama a bloques del sistema de robot Rhino.

El robot Rhino que se tiene en el laboratorio cuenta con todos los elementos de un robot industrial, a saber:

SISTEMA DE ROBOT INDUSTRIAL	SISTEMA DE ROBOT RHINO
∘Brazo mecánico	∘Brazo de robot XR-3
∘Controlador con base en computadora	∘Controlador Mark III
∘Herramienta en el extremo del brazo eléctricamente	∘Pinza controlada
∘Terminal de enseñanza	∘Computadora
∘"Teach pendant"	∘"Teach pendant"
∘Dispositivo de almacén de programas	∘Floppys grabados en computadora
∘Fuente de poder externa	

a. Brazo del robot XR-3

El robot tiene una geometría esférica articulada con cinco grados de libertad en el brazo y muñeca (fig 2). Los cinco ejes incluyen movimiento

de rotación en cintura, hombros, codo para posicionar la mano y movimientos de rotación y flexión en la muñeca para orientación de la pinza.

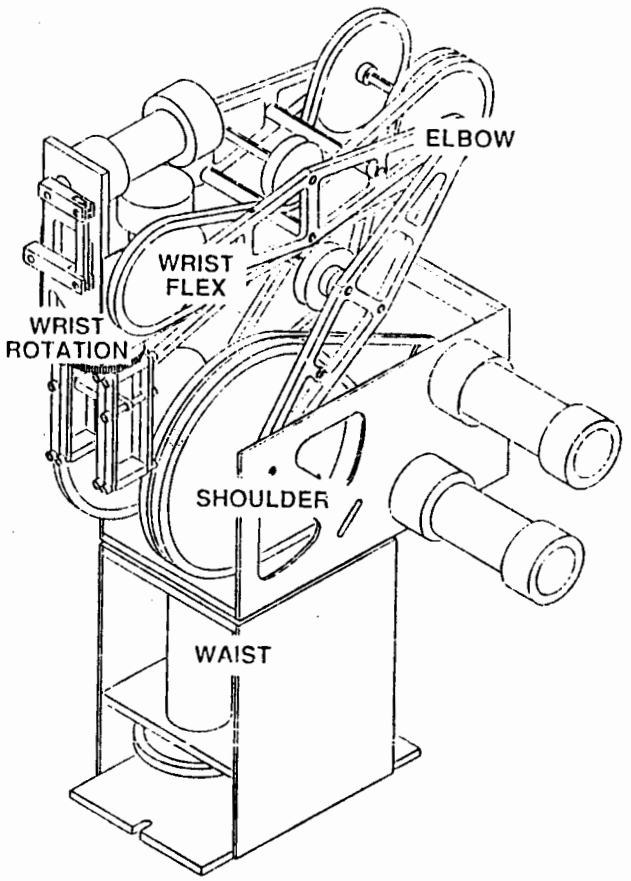


Fig. 2 Brazo del robot XR-3

El movimiento es transferido desde los motores hasta su respectiva articulación por medio de cadenas y conectores de eslabón. La razón de la rotación de los motores es reducida por engranes dentro del cuerpo del motor y en los eslabones.

Los motores son servomecanismos de DC con codificadores ópticos ajustados a un determinado ángulo de la articulación y posición del brazo.

Los codificadores ópticos no son visibles, pero se pueden accesar removiendo las cubiertas de los motores; se conectan al controlador por medio de un cable plano con 10 conductores a través de los cuales se suministra potencia al motor y se carga la posición del codificador.

Un sexto motor se monta en la estructura de la muñeca, y es usado para mover la pinza sin causar movimiento en ningún eje del brazo.

b. Controlador Mark III

Es un controlador completo de una celda de trabajo con una microcomputadora interna, software de sistema operativo e interface. La electrónica proporciona la interface para motores de 8 VDC, un "teach pendant", y un conector RS-232C para conectar el puerto serie de una computadora externa. Ver figura 3.

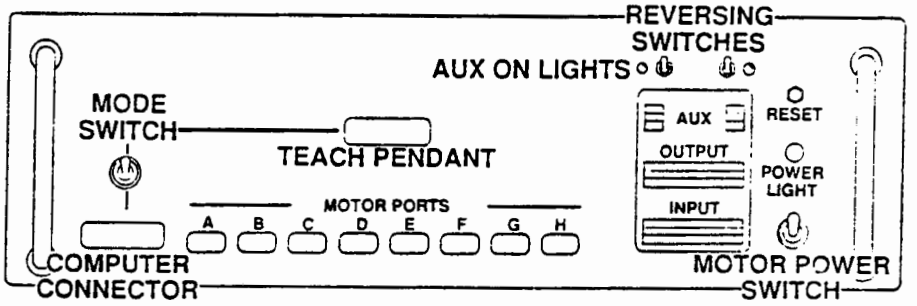


Fig. 3 Controlador Mark III.

El controlador tiene, además, 8 puertos de entrada discreta (0-5V), 8 puertos de salida discreta (0-5V) compatibles con TTL y 2 puertos de salida auxiliares de alta potencia (0 ó 20V, 2A).

Los puertos de los motores son puertos DC servo, lo que significa que proporcionan pulsos de 20 VDC para manejar los 6 motores. Para cambiar la dirección de rotación del motor, se varía la polaridad de los pulsos. El ciclo de trabajo de los pulsos se modifica para desacelerar los motores al final del movimiento. Asimismo, este control de ancho de pulso utilizado en Rhino se emplea en otros robots para los procesos de aceleración y desaceleración.

Los componentes más importantes dentro del controlador son un microprocesador 6502, una fuente de voltaje y una memoria de estado sólido. El sistema operativo, un programa permanentemente almacenado dentro de la memoria del controlador, permite a éste enviar y recibir información a través de las interfaces, cambiar posiciones del brazo y operar el equipo de la celda de trabajo.

El controlador Mark III se comunica con el "teach pendant" o con una computadora para manejar la información necesaria para el control del brazo, si es con computadora, se emplea como medio el puerto RS-232C serie. Apple IIe e IBM PC/XT dan soporte al robot Rhino. El controlador tiene un interruptor de selección de modo (**MODE SELECT**) para la comunicación con "teach pendant" o con computadora. Existen dos interruptores que permiten controlar separadamente la fuente de voltaje principal y la fuente de alimentacion para los motores. Los interruptores de los puertos auxiliares AUX1 y AUX2 permiten invertir la polaridad de estos puertos de salida. Un botón normalmente abierto, etiquetado con **RESET**, restabece los valores iniciales en la memoria del controlador, concede operar al sistema, y es un botón de detención de emergencia.

c. Herramienta de trabajo

La herramienta que se usa generalmente es una pinza tipo paralelo actuada por un servomotor de DC, esto brinda un sinnúmero de posiciones programadas para los dedos de la pinza.

d. Computadora y "Teach Pendant"

El sistema de robot Rhino posee ambas ventajas para programar al robot y manejar el equipo de la celda de trabajo. En la figura 4 se muestra un esquema de la conexión del sistema y, en la figura 5, se señala la cara principal del "teach pendant".

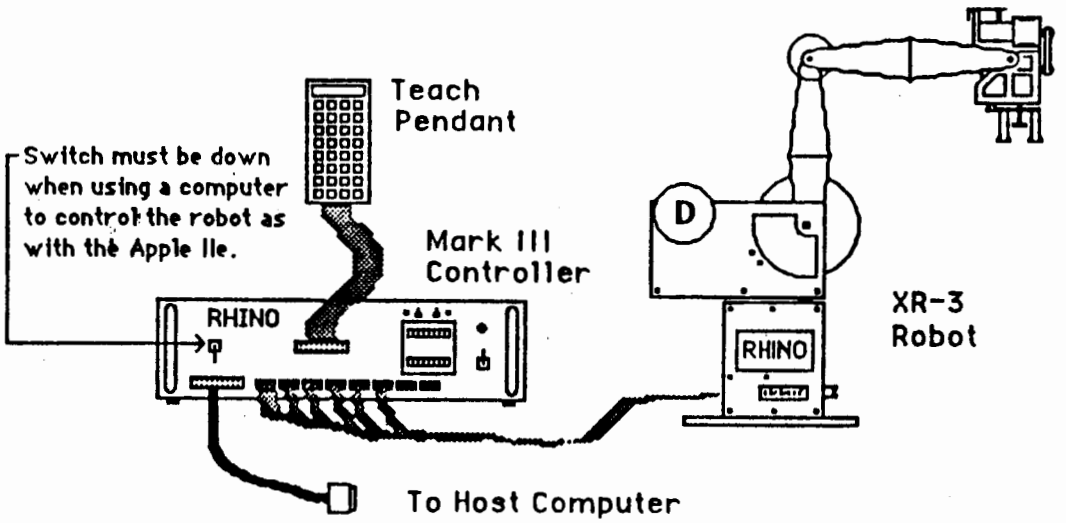


Fig. 4 Conexión típica del robot.

El "teach pendant" se conecta al panel frontal del Mark III. Este dispositivo es una unidad programable controlada por microprocesador, contiene 32 teclas y un desplegue digital de 7 digitos, 15 de las teclas son usadas para programar y 16 de ellas permiten el control de movimiento para los servos de los 5 ejes de unión, el servo de la grapa y dos servos auxiliares.

El "teach pendant" soporta programas para la celda de trabajo desarrollados en dos formas:

- Usando sólo el "teach pendant".
- Empleando la computadora para escribirlos y el "teach pendant" para ejecutarlos.

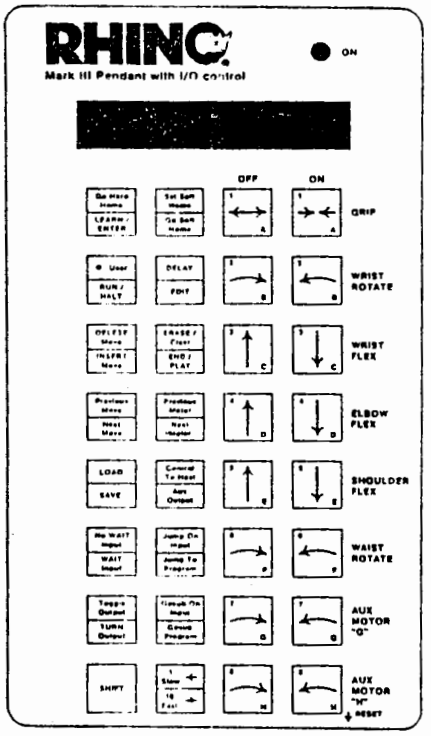


Fig. 5 "Teach Pendant".

La computadora se conecta al controlador Mark III por medio del puerto serie RS-232C. Programas elaborados utilizando sólo el "Teach pendant" pueden ser transferidos a la computadora para ser editados y almacenados en floppys. También, se pueden editar programas en lenguajes de alto nivel como BASIC, o bien, en el lenguaje propio del robot ROBO-TALK, que se corre desde la PC.

e. Dispositivos de almacenamiento de programas

Los manejadores de discos de la microcomputadora son los mecanismos de almacenamiento permanente para programas de celdas de trabajo producidas sobre el sistema del robot Rhino. Es posible salvar en floppys programas desde el controlador Mark III a través del puerto serie al

drive para, después, ser cargados en el controlador para ejecución.

Un segundo tipo de almacenamiento permanente es el empleo de dispositivos de memoria de estado sólido EPROM's. El sistema operativo del robot está almacenado en uno de estos dispositivos. Este programa permite al robot ser programado en BASIC, con el "teach pendant" o bien con el lenguaje ROBO-TALK. Además, controla el envío de datos entre la computadora y el "teach pendant".

e. Fuente externa de alimentación

El sistema estándar usa 120 VAC, 60 Hz y sistemas como el de grapa de vacío requieren, además, una fuente neumática.

IV. DESARROLLO

1. Asegúrese de que el cable del controlador no esté conectado a la línea.

2) Remueva la cubierta de los codificadores (uno a la vez) ubicados en la parte posterior de cada motor. El disco con superficies reflejantes alternantes es un codificador óptico que se usa para medir el desplazamento del eje del motor. El codificador óptico es el mecanismo de realimentación que dice al controlador el cambio incremental en la rotación del eje de unión.

3) Cada eje de unión tiene asignado un motor que se identifica con las letras **A, B, C, D, E** y **F**. Ver figura 2. Llene la columna etiquetada con **Letra de identificación** de la tabla 1 con la letra correspondiente del motor actuador.

4) Manualmente, rote el disco de cada codificador en ambas

direcciones y escriba en la segunda columna de la tabla 1 la dirección en que el eje de unión correspondiente se mueve.

Tabla 1

	WHEEL ROTATION	IDENTIFICATION LETTER	DRIVE LINKAGE	MOTOR GEAR RATIO	FINAL REDUCTION
WAIST					
SHOULDER					
ELBOW					
WRIST FLEX					
WRIST ROTATION					
GRIPPER					

5) El brazo del robot utiliza cadenas, palancas y engranes para hacer la transmisión mecánica entre los motores y su unión correspondiente. Determine el tipo de elementos empleados para cada movimiento del brazo y coloque esta información en la tabla 1 en la columna etiquetada **Enlace mecánico**.

6) Cada motor tiene una razón de engranaje entre el eje del motor mismo y el eje que habrá de usarse para manejar el movimiento de la unión correspondiente, esto se hace con el fin de amplificar el par a la salida del motor. Encuentre en el manual de usuario la razón de engranaje para cada motor actuador y grave esta información en la columna **Razón de engranaje** de la tabla 1.

7) Al efectuar el enlace entre motor y eje de unión se da una amplificación extra al par del motor debido a que los engranes empleados

poseen diferente diámetro. A esta ventaja mecánica obtenida por la razón de engranaje en el motor y el enlace mecánico en el brazo se le llama **Reducción final**. Obtenga el valor de la reducción final para cada eje de unión del manual de usuario y llene la última columna de la tabla 1.

8) Dibuje un esquema de los elementos usados en la transmisión mecánica para manejar los ejes de unión de hombro y codo. Identifique el tamaño de cada engrane como grande, mediano o pequeño.

9) El "teach pendant" es una caja portátil de control presente en casi todos los robots industriales. Su función es programar al robot a través de comandos de un lenguaje para enseñarle de manera sencilla posiciones deseadas. Ejecute las siguientes indicaciones a fin de operar al sistema con el "teach pendant":

- Coloque el interruptor de selección de modo para operar con "teach pendant".

- Encienda el controlador con el interruptor principal situado en la parte posterior de aquél. Verifique que encienda la lámpara indicadora localizada en la parte del frente.

- Active el interruptor de la fuente de los motores, con el fin de hacer llegar energía a éstos sin que pierdan su posición original.

- Presione el botón de reset, esto limpiará todas las localidades de memoria y restablecerá todas las salidas a la condición de apagado. En el desplegado del "teach pendant" se observará la palabra **init** lo que significa que el sistema se está inicializando.

- Espere a que en el desplegado del "teach pendant" aparezca la letra P lo que indicará que el controlador se halla en el modo **PLAY** o de operación normal.

∘ Pruebe presionar cualquiera de las teclas identificadas con las letras de la **A** a la **F** en sus dos sentidos. Si desea regresar a la posición original oprima la tecla identificada con **Go Soft Home**.

V. CUESTIONARIO

1. Mencione los elementos básicos que conforman un sistema de robot.

2. ¿Cuál es la geometría y el número de grados de libertad en el brazo de robot Rhino?.

3. ¿Cuáles son los nombres de los ejes de unión?. Clasifíquelos en dos grupos: los que proporcionan posicionamiento y los que dan orientación al elemento terminal.

4. Mencione las características de los siguientes elementos ubicados en el panel de control:

Puertos de motores actuadores.
Puertos de entrada salida.
Puertos auxiliares.
Botón de reset.
Interruptor de selección de modo.

5. ¿Cuál es la función del codificador óptico?.

6. BIBLIOGRAFIA

1. RHINO Robots, Inc. XR owner's manual. Champaign Illinois.

PRACTICA 12

PROGRAMACION DEL ROBOT CON "TEACH PENDANT" O PROGRAMADOR

I. OBJETIVOS

- Identificar los cuatro modos de operación del "teach pendant" o programador.
- Preparar al brazo del robot Rhino y al controlador Mark III para operarlos en modo **play**.
- Usar correctamente las teclas de movimiento básico del programador: **shift**, **go hard home**, movimientos de motores **A-F**, **fast** y **slow**.
- Preparar al brazo del robot Rhino y al controlador Mark III para operarlo en modo **learn**.
- Enseñar al robot programas básicos con el programador para movimientos simples y movimientos multiejes o coordinados.
- Emplear correctamente las teclas de programación siguientes: **learn/enter**, **delay**, **set soft home**, **go soft home**, **jump to program** y ***user**.
- Esquematizar el espacio de trabajo del brazo.
- Aplicar correctamente los términos exactiud y repetibilidad.

II. EQUIPO REQUERIDO

- Brazo de robot XR3
- Pinza estandar
- Controlador Mark III
- Programador del XR-3
- Base de aluminio

III. INTRODUCCION

El "Teach Pendant" o programador, una caja manual portátil, es un dispositivo presente en todos los sistemas de robots industriales. La función del programador es de programación del brazo del robot la cual

incluye enseñanza de posiciones y manejo de comandos. Muchos robots industriales permiten que el programa provenga del programador o de una terminal. El programador es ampliamente utilizado para la programación de "primera intención" debido a que ofrece muchas ventajas como lo es su facilidad de uso respecto a otras formas de programación. Todos los programadores son similares y usan un lenguaje pictórico entendible tal que es fácil aprender su manejo en poco tiempo. Ver fig. 1.

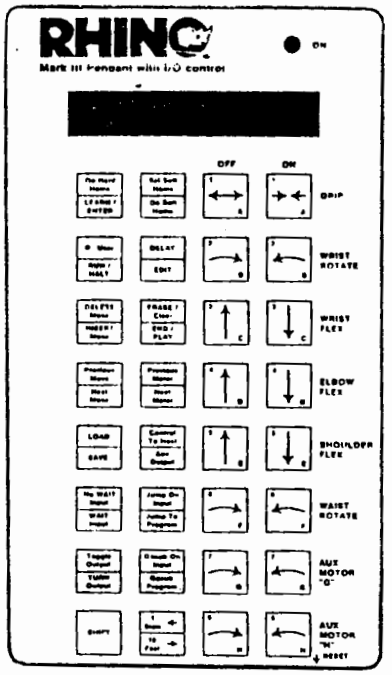


Fig. 1 Programador del robot Rhino

Con un programador, un operador puede controlar un brazo de robot sin tener conocimiento de un lenguaje formal de programación. Es la herramienta ideal para una rápida introducción a los sistemas de robots.

El robot Rhino cuenta con un programador con la mayoría de las capacidades que tienen los programadores de robots industriales tal como el Cincinnati Milacron, ASEA o GMF.

El programador del robot Rhino posee las siguientes características:

1. Una botonera manual de 32 teclas en un casco de metal.

2. Una tarjeta basada en microprocesador què constituye el corazón del sistema del programador.

3. Un disco que contiene el software necesario para llamar o salvar programas desde o hacia la computadora.

4. Este manual.

a. Procedimiento de encendido

El sistema de robot Rhino incluye 3 microcomputadoras. La más obvia es la IBM PC/XT o Apple IIe conectada al controlador Mark III vía puerto serie. Las otras dos se encuentran en el interior del controlador Mark III y en el programador. Se hace necesario seguir una cierta secuencia de encendido con la finalidad de lograr una buena comunicación entre las computadoras. Este procedimiento se muestra a continuación:

1. Encender la computadora (IBM PC/XT o Apple IIe).

2. Con el switch **mode select** colocado en la posición **up** para el **teach pendant** aplicar la energía al controlador usando el interruptor principal localizado en la parte trasera del controlador.

3. Colocar el interruptor **motor power** ubicado en el controlador, en la posición **on**.

4. Presionar el botón de **reset**.

El despliegue digital porporciona información al operador acerca de lo que está pasando en el sistema una vez que se ha aplicado energía al sistema.

Las 32 teclas del teclado del programador pueden ser divididas en dos partes. Las dos columnas de la izquierda son las teclas de función. Las dos columnas de la derecha son las teclas de movimiento de motores. Ver fig 3. Nóte, además, que todas las teclas excepto la tecla **shift** sirven para más de una función.

Las teclas corresponden a cada uno de los motores, designados **A** a **H**, sobre el controlador, esto es:

Motor	**Servicio**
A	Pinza
B	Rotación de muñeca
C	Flexión de muñeca
D	Flexión de codo
E	Flexión de hombro
F	Rotación de cintura

b. Modos de operación del teach pendant

El "tech pendant" tiene cuatro modos de operación:

PLAY Aquí, se pueden llevar a cabo varios movimientos de motores.

LEARN Enseña al robot una determinada secuencia de movimientos.

EDIT Se utiliza para realizar modificaciones y afinar la secuencia de movimientos enseñada.

RUN Ejecuta la secuencia de movimientos previamente enseñada.

Debe notarse que siempre se pasa por el modo **play** cuando se va de un modo a otro. Ver figura 2.

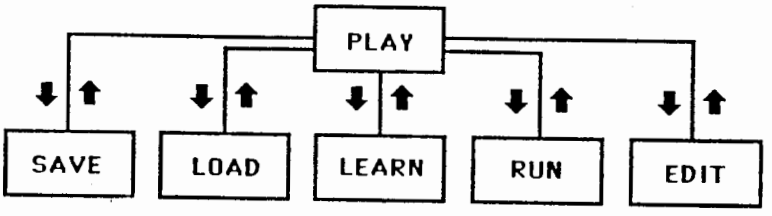


Fig. 2 Protocolo de transferencia entre modos

c. Modo PLAY

Al encender el sistema se despliega el mensaje **"init"**, en seguida, aparece la letra **P**. Esto indica que el sistema esta en modo **PLAY**. Este modo es útil para practicar movimientos y posiciones del robot en forma sencilla.

En el modo **PLAY** todas las teclas de movimiento de motores (ver figura 3) están activas. Presionando alguna de esas teclas se puede mover el correspondiente motor en el robot. Si uno de los motores no estuviera conectado se desplegaría el mensaje **"OFF"**.

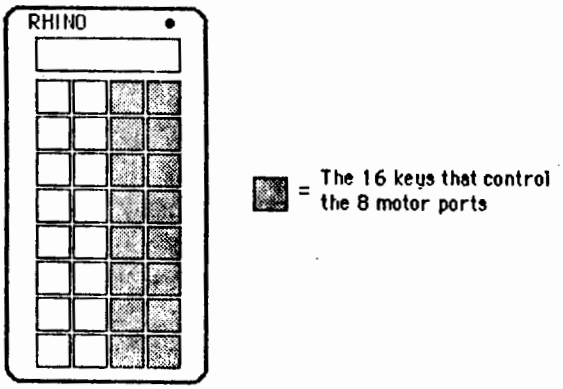


Fig. 3 Las 16 teclas de movimiento de motores.

Conforme se presiona alguna tecla de movimiento, éste es registrado en el display. El display siempre registra posiciones **absolutas** no relativas. Por ejemplo, si rotáramos la muñeca (motor B) 120 pasos en dirección contraria a la del reloj, el display mostraría **"Pb 120"**.

TECLA DE DISTANCIA

La tecla etiquetada con los números **1 y 10(slow fast)** se emplea para

fijar la distancia en que se moverá el motor cada vez que se presione una tecla. Con **10**, indicamos que el motor se moverá 10 cuentas del codificador cada vez que se presione una tecla de movimiento. Este es el valor por default.

El **1** es activado presionando esta tecla conjuntamente con la tecla **shift**.

RUTINA DE SEGURIDAD

Si durante un movimiento de algún motor el robot encuentra un obstáculo que le impide seguir su movimiento, el programador desplegará el mensaje "**Stall**". El sistema automáticamente regresa de esta posición y limpiará el mensaje en cuanto se deje de presionar la tecla.

d. Modo LEARN

En el modo **LEARN** es posible enseñar al robot una secuencia de movimientos que serán retenidos en su memoria. Se deben hecer dos cosas antes de que se entre al modo **LEARN**: Limpiar (**clear**) la memoria y enviar al robot a su posición **Hard home**.

Para limpiar la memoria, presione las teclas **shift** y **erase/clear**.

Hard home significa hogar "físico" (ver figura 4). Es usado para inicializar al robot en una posición conocida. Después de haber dado el comando **go hard home**, el robot cerrará las pinzas y entonces se moverá para encontrar los centros exactos de los microswitches de los motores B, C, D, E, y F y finalmente, abrirá la mano. Se debe ejecutar el comando **go hard home** una vez después de cada **reset** antes de que el sistema le permita entrar al modo **LEARN**, en caso de olvidarlo, el sistema desplegará "**do Hard**".

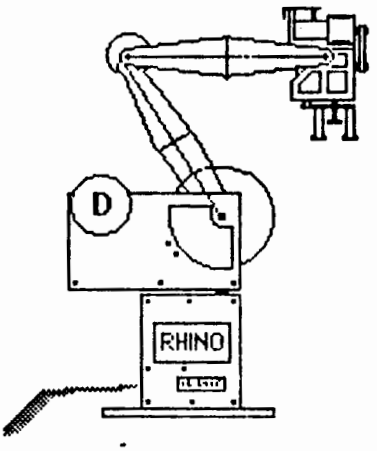


Fig. 4 Posición "Hard Home" del robot

UBICANDO LA POSICION "SOFT HOME"

La posición "**soft home**" es una posición inicial del robot conveniente para crear una secuencia de movimientos.

Para establecer una posición "**soft home**" (software home) se requiere mover al robot en la posición necesitada y presionar **set soft home**. De allí en adelante, el robot regresará a esta posición siempre que se presione la tecla **go soft home**.

Esta posición es retenida mientras èl sistema permanece encendido o hasta que se presione el botón de reset y no puede ser establecida a menos que se halla ejecutado la rutina "**go hard home**" primero. Si no se ha definido una posición **soft home**, el robot asume por default la posición de **hard home** como la misma.

APRENDIENDO MOVIMIENTOS SIMPLES DE MOTOR

Para este punto se debe estar en el modo **LEARN**. Si no aparece una letra "**L**" en el desplegado, se deberá presionar la tecla **learn** que lo pondrá en este modo.

Para un movimiento simple del motor únicamente presione la tecla que corresponde al motor que se desea mover hasta alcanzar la posición deseada. Ahora, presione la tecla **enter** para grabar el movimiento. El desplegado mostrará **LStP 1** lo cual significará que estamos en el modo **LEARN** y que hemos grabado un movimiento.

Se pueden grabar tantos movimientos como se desee, y, también, se podría incluir una rutina de **go soft home** al final la cual enviaría al robot a su posición original; el despliegue mostrará **LAtSOFt.**

Para completar el programa, presione la tecla **END.** La **L** del display desaparecerá y, en su lugar, aparecerá una letra **P** para indicar que el robot está nuevamente en modo **PLAY.**

PROGRAMANDO UN RETRASO "DELAY"

El comando **delay** permite programar retrasos de 1,2 ó 3 segundos en cualquier punto del programa a ser ejecutado por el robot. El retraso se introduce presionando las teclas **shift** y **delay** simultáneamente seguidas por una tecla **1, 2** ó **3** para indicar la longitud del retraso en segundos. Cada retraso está limitado a un máximo de 3 segundos. Cabe mencionar que mayores retrasos se alcanzan introduciendo varios comandos **delay**. Los retrasos pueden estar incluidos en un movimiento programado, o puede ser el único dato de ese paso.

e. Modo RUN

Después que se ha enseñado una secuencia de movimientos al robot, se está en la disponibilidad de correr el programa bajo el control del programador. Presionando la tecla **RUN** se podrá observar cómo el robot ejecuta una a una las instrucciones que se programaron.

El despliegue indicará la condición del robot mientras se ejecuta el programa. Durante el primer movimiento de la secuencia, el desplegado

mostrará **rStP 1** indicando que estamos en el modo **RUN** y que se trata del paso 1.

Cada vez que se presiona la tecla **run** el robot ejecuta el programa una vez. Si se presiona la tecla **run/halt** durante la ejecución de un programa, el robot se detiene inmediatamente y se va a modo **HALT**. El programador desplegará ahora **rHLt**. Para reanudar la operación bastará con presionar nuevamente la tecla **run**.

Si se presiona la tecla **end** durante la ejecución del programa, el robot se detiene inmediatamente. Posteriormente el display mostrará una P que señalará que el sistema ha regresado al modo **PLAY**. Cuando **run** se presiona después de haber presionado **end**, el robot empezará el programa iniciando por la ejecución del primer punto desde la posición presente del robot. Si la tecla **end** es presionada durante el modo **HALT**, el sistema regresa al modo **PLAY**.

Si ya se corrió una secuencia de movimientos, se puede regresar al modo **LEARN** y añadir otros al final de la secuencia previa de movimientos. Esta técnica permite hacer series complejas de movimientos simplemente encadenando pequeñas secuencias de ellos.

f. Movimiento multiejes o coordinado

La diferencia esencial entre el movimiento **multiejes** y el **simple** radica en el uso del comando **enter**. En el movimiento de ejes simple, para llevar al brazo a la posición requerida se presiona **enter** después de que un eje ha sido movido . En el movimiento multiejes, se presiona **enter** después de que se ha movido una combinación de ejes para alcanzar un punto deseado en el espacio. Los pasos preliminares son los mismos, i.e. hacer un "**go hard home**", **clear** (limpiar la memoria, opcional), colocar un "**soft home**", e introducirse al modo **LEARN**.

Siempre que planee un movimiento multiejes, tenga en mente los siguientes principios:

PRINCIPIO 1

Inicie el movimiento del robot empezando con la cintura y finalizando con la grapa.

Un punto importante a recordar es que el robot se mueve de punto a punto, no en una trayectoria específica.

PRINCIPIO 2

Se recomienda que el movimiento de la grapa sea un movimiento separado.

PRINCIPIO 3

Cada movimiento puede mover a alguno de los motores en una sola dirección.

PRINCIPIO 4

Observe cómo se mueve el robot de un punto a otro para familiarizase con su sistema coordenado.

PRINCIPIO 5

Evite movimientos multiejes complicados en el principio.

PRINCIPIO 6

Cuando se le indica al robot un **go soft home** al final de la secuencia, se irá a su posición "**soft home**" en un movimiento simple y multiejes. Si esto causa problemas, se sugiere llevar al robot a una posición intermedia y grabar el movimiento con el comando **enter** antes de dar el comando **go soft home**.

g. Ejecutando un movimiento multiejes

Ya que se ha terminado de introducir el programa en el sistema, presione la tecla **end** y, enseguida, la tecla **run**.

IV. DESARROLLO

1) Use el procedimiento de encendido descrito en este experimento y coloque al sistema en el modo **play**. Anote las respuestas del sistema a cada paso de la secuencia de encendido.

2) Observe el movimiento de las uniones conforme son presionadas las teclas de cada motor. Pruebe presionar **shift-slow**, y repita la misma operación.

3) Ejecute la rutina **go hard home** y describa la respuesta del sistema, incluya la secuencia de movimientos, es decir, el orden en que el controlador localiza los interruptores de límite para cada eje.

4) Usando las teclas de movimiento de motores **A-F** trate, experimentalmente, de definir el espacio de trabajo del brazo del robot. Dibuje un esquema.

5) Redefina una posición **soft home**, lleve el brazo a otra posición y, finalmente, compruebe que regresa a la posición definida presionando la tecla **go soft home**.

6) Utilice el procedimiento descrito en la sección de teoría para operar al sistema del robot en el modo **learn**.

7) Use las teclas **learn/enter** y **end/play** para cambiar de modo play a **learn** y regresar. Observe la lectura que aparece en el desplegado del programador en cada modo de operación.

8) Coloque al sistema en modo **learn**, programe al robot para que realice una tarea con base en movimientos simples, regrese al modo **play** y ejecútela. La tarea consiste en que el brazo deberá tomar un objeto de un cierto lugar, lo llevará a otra posición, lo dejará allí por tres segundos y en seguida lo tomará nuevamente y lo regresará a su posición original. Este proceso es de aplicación muy usual para los robots en una línea de

producción y se refiere a él como procedimiento de **"pick up and place"**.

9) Repita el ejercicio anterior pero, ahora, empleando movimiento multiejes; para esto, deberá tener presentes los seis principios de programación expuestos en la sección teórica de esta misma práctica.

10) Utilizando los comandos de **jump** y **label**, desarrolle una tarea similar a la anterior pero, en este caso deberá ser repetitiva partiendo siempre de una posición definida con **soft home**.

V. CUESTIONARIO

1. ¿Cuál es la diferencia entre **hard home** y **soft home**?

2. ¿Cómo encuentra el robot la posición de **hard home** y cómo la de **soft home**?

3. ¿Qué teclas deben ser presionadas antes de entrar al modo **learn**?

4. ¿Qué tecla se presiona para sacar al sistema de modo **learn**?

5. ¿Por qué es necesario que el brazo se ubique en la posición **hard home** antes de que se ejecute un programa?

6. Cuando un programa está corriendo, ¿en qué difieren las operaciones de **run/halt** y **end/play**?

7. ¿Qué factores afectan la repetibilidad del robot?

8. ¿Qué factores afectan la exactitud del robot?

9. Mencione algunas otras aplicaciones del robot dentro de un proceso industrial.

VI. BIBLIOGRAFIA

1. RHINO Robots Inc. Owner's manual. Champaign Illinois.

2. REHG, James A. Instructor's guide I: Introduction to robotics. Rhino Robots Inc. Champaign Illinois.

PRACTICA 13

PROGRAMACION DEL ROBOT A TRAVES DE BASIC

I. OBJETIVOS

- Conocer la modalidad de programación de robots empleando un lenguaje de alto nivel por medio de la computadora.
- Notar las ventajas y desventajas que ofrece controlar al robot a través de una computadora.
- Distinguir la manera en que se lleva a cabo el intercambio de información entre la computadora y el controlador Mark III del robot.
- Conocer el conjunto de comandos "Kernel" que reconoce el controlador, su signficado y la forma en que se usan para operar al robot.
- Utilizar los comandos "Kernel", adecuadamente, en programas que ilustren su efectividad.

II. MATERIAL Y EQUIPO

- Brazo del robot XR-3
- Controlador Mark III
- "Teach Pendant" o programador del Rhino
- Computadora compatible con IBM PC/XT
- Base de aluminio
- Disco con lenguaje BASIC

III. INTRODUCCION

El controlador Mark III reconoce ciertos comandos de tipo alfanumérico en forma de cadenas introducidas por medio de su entrada para puerto serie los cuales son denominados comandos **kernel** y tienen la forma "F+50", "F?", "AX"; dichos comandos son utilizados para manejar los puertos de los motores, los puertos de entrada y salida, etc.

Cualquier lenguaje de programación que permita el acceso al puerto de comunicaciones serie se puede emplear aunque, en esta práctica, se usará

BASIC por su facilidad de manejo.

Para poder controlar los movimientos del robot desde la computadora, se interconecta el puerto **RS232C** de ésta con la entrada de el controlador Mark III provista para tal fin, además, se coloca el switch de modo en la posición de **computer** (ver figura 1).

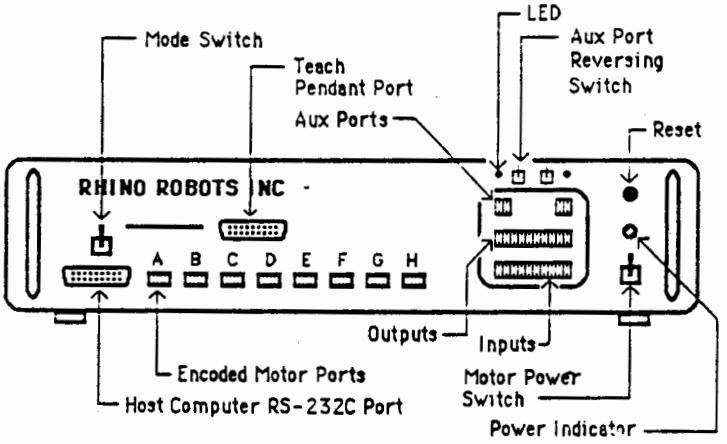


Fig. 1 Frente del controlador Mark III.

El controlador Mark III reconoce los siguientes comandos:

COMANDO	DESCRIPCION
<RETURN>	Retorno de carro (inicia un movimiento)
?	Regresa la distancia restante
A-H	Coloca el valor del movimiento del motor
I	Investiga el estado de los switches de límite C-H
J	Investiga el estado de los switches de límite A-B y entradas
K	Investiga el estado de las entradas
L	Enciende el puerto AUX #1
M	Apaga el puerto AUX #1
N	Enciende el puerto AUX #2
O	Apaga el puerto AUX #2

P	Coloca los bits de salida en alto
Q	Reinicializa al controlador
R	Coloca los bits de salida en bajo
X	Detiene a alguno de los motores A-H

El conjunto de comandos del sistema ha sido diseñado para tener el control completo sobre el controlador Mark III. A través del uso de los 14 comandos básicos, el usuario puede controlar la posición de los 8 motores, leer cualquiera de los 16 bits de entrada, colocar cualquiera de los 8 bits de salida y controlar los puertos auxiliares (**AUX**).

Como se puede prever, para poder enviar estos comandos bastará con escribirlos hacia el archivo que identifica al puerto serie, en este caso el **#1**. Esta operación se efectúa con la ayuda de un programa en el que habrá que considerar las características de la comunicación asíncrona entre computadora y controlador.

a. Habilitando la comunicación

La idea general es abrir el puerto de comunicaciones **COM1** como un archivo **#1**. Entonces, por lectura y escritura de varios comandos kernel a este archivo, se podrán leer los valores de los codificadores, manejar motores, etc.

Para que se establecezca una buena comunicación entre la computadora y el controlador, se recomienda la siguiente secuencia de encendido:

1. Inserte el sistema operativo en la computadora.
2. Encienda el interruptor principal (**main power**) en el controlador.
3. Encienda el interruptor que alimenta a los motores (**motor power**).
4. Una vez que se ha cargado el sistema en la computadora, entre al editor de BASIC.

Para activar la comunicación entre la computadora y el controlador se deberá teclear al inicio del programa la siguiente línea:

```
10   OPEN "COM1:9600,E,7,2,CS,DS,CD" AS #1
```

Esta línea abrirá un archivo de comunicación serie identificándolo como #1 y colocará la velocidad de comunicación y el formato de los datos.

b. El registro de error

Para el manejo de los motores actuadores del brazo, el controlador mantiene un registro denominado **registro de error**, cuyo tamaño es de 8 bits para cada uno de los 8 motores. Cuando alguno de los registros de error está en cero, el motor correspondiente permanece parado, mientras que, si el registro de error es diferente de cero, provocará que el controlador conecte en el motor una señal de voltaje con lo que éste tenderá a girar en uno u otro sentido dependiendo del signo de ésta, decrementándose en cada paso el valor del registro de error hasta alcanzar, otra vez, el valor de cero.

c. Inicio de un movimiento <return>

Siempre que un retorno de carro se recibe desde la computadora, el controlador toma la cuenta del movimiento en el buffer de conteo de movimiento del motor y lo añade (o sustrae) al valor del registro de error del motor al que se está direccionando por medio del buffer del motor.

d. Comando start (A-H)

El comando **start** añade el valor del movimiento deseado del motor al registro de error correspondiente y tan pronto como el valor es agregado a

este registro, el controlador mueve al motor, el codificador ubicado en la parte posterior del motor es leído y el contador en el registro es decrementado; este proceso continúa hasta que el registro de error alcanza el valor cero.

Recuerde lo siguiente:

1. Un comando **start** agrega un valor a un determinado registro de error de un motor.
2. El valor añadido al registro de error representa el número de pasos que el motor requerido deberá moverse para regresar al registro a cero.
3. Los motores tienden a la condición de error cero.
4. El signo del comando **start** designa la dirección del movimiento.

Algunos ejemplos de uso de este comando son:

```
10   OPEN "COM1:9600,E,7,2,CS,DS,CD" AS #1
20   PRINT #1,"A+55"
30   END
```

Corriendo el programa anterior provocará que la grapa (motor A) se cierre totalmente y, para abrirla otra vez, sólo hay que cambiar el signo en la línea 20 y correr de nuevo el programa. Si ahora sustituimos la línea 20 por:

```
20   PRINT #1,"F+50"
```

Esta línea provocará que la cintura (motor F) gire 50 pasos en dirección positiva.

Enviando únicamente un retorno de carro sin movimiento de motor, después de un comando de movimiento, provoca que se repita el último comando de movimiento. Aun los retornos de carro que forman parte de otros

comandos hacen que el comando previo se vuelva a ejecutar. Para cancelar un comando de movimiento que pudiera ser re-ejecutado por un retorno de carro, todos los comandos deberán ser precedidos por un identificador de motor. Cualquier identificador de motor coloca el buffer de movimiento en cero si no lleva consigo algún valor. Una vez que el buffer de movimiento ha sido limpiado, otros comandos con retorno de carro no activarán la instrucción de movimiento. Por ejemplo, comandos similares a los siguientes limpiarán el buffer de movimiento.

```
nnn    PRINT #1,"A" <return>
nnn    PRINT #1,"CI" <return>
nnn    PRINT #1,"EJ" <return>
nnn    PRINT #1,"BL" <return>
```

Una vez que el buffer ha sido limpiado, otros comandos pueden ser emitidos sin el carácter de identificador del motor.

e. Comando question (?)

Instruye al controlador para que éste regrese el valor actual del registro de error de un motor especificado. Es decir, pregunta cuánto más debe rotar el motor para que el registro de error alcance la condición cero. Para utilizarlo bastará con colocar este comando (?) seguido de la letra que identifica al motor, por ejemplo "A?", "B?", etc.

Para permitir a la computadora recibir la respuesta, se emplea la subrutina siguiente escrita en BASIC:

```
100  IF LOC(1)=0 THEN 100 ELSE W$=INPUT$(LOC(1),#1)
110  W=ASC(W$)-32
120  RETURN
```

Esta rutina deberá ser accesada inmediatamente después de haber

enviado un comando **question (?)**, por ejemplo:

```
30   PRINT #1,"F?";:GOSUB nnn
```

Algunas consideraciones que se deben tener en mente son las siguientes:

- Con 8 bits, el número decimal más grande que se puede representar es el 127.

- Cuando el controlador recibe un comando **question (?)** añade 32_{10} a la respuesta antes de enviarla a la computadora, por lo que será necesario restar esa misma cantidad al recibirla para poder identificar su valor en forma correcta.

- Añadiendo 32_{10} a los valores transmitidos por el controlador se asegura que a la computadora nunca llegue un valor inferior al 32_{10} que se podría interpretar como un código de control.

- No se debe dejar que ningún registro de error exceda el valor de 95_{10} ya que provocaría saturación y, en consecuencia, resultados erróneos.

- Para poder llevar a cabo movimientos cuya magnitud sea superior a 95 unidades, se envía un primer paquete de 95 usando el comando **start** y, luego, se monitorea el estado del registro de error con el comando **question**. Se continúan enviando paquetes quizá, más pequeños, a fin de ir añadiéndolos al registro de error sin permitir que éste llegue a cero sólo hasta que se halla completado el movimiento.

f. Comando stop (X)

Este comando es utilizado para detener el movimiento de un motor determinado, esto lo realiza llevando a cero al registro de error

correspondiente al motor especificado. Es de gran utilidad cuando se desea proteger al robot contra colisiones. Por ejemplo:

```
mmm     PRINT #1,"FX"
```

Esta última línea causará que la cintura (motor **F**) se detenga. El comando **X** es seguido por un retorno de carro. La parte que todavía restaba por mover se pierde después de enviar este comando.

g. Comando Inquiry (I)

El commando **inquiry (I)** permite al usuario preguntar por el estado de los 6 microswitches asignados a los motores **C, D, E, F, G y H.**

El comando regresa el estado de los 6 microswitches en un sólo byte. El byte leído es interpretado (después de sustraerle 32) como sigue:

BIT	MOTOR
0 LSB	C
1	D
2	E
3	F
4	G
5	H
6	---
7 MSB	---

Ejemplos de uso de este comando en BASIC son:

```
mmm  PRINT #1,"I"
ppp  PRINT #1,"AI"        (limpia primero el buffer de movimiento)
```

h. Comando inquiry (J)

El comando **inquiry (J)** dice al controlador que envíe el estado de los microswitches sobre los motores **A** y **B** y el estado de las líneas de entrada **1**, **2**, **3** y **4** del puerto de entrada. El byte de datos leído es de la forma **00BA4321**+32, donde **A** y **B** son los niveles de los switches límites de los motores **A** y **B**, y **4**, **3**, **2** y **1**, son los niveles de las líneas de entrada **4** a la **1**. Al igual que toda la información regresada a la computadora, se añade 32 al valor enviado para asegurar que no se tengan caracteres de control ASCII.

El comando **inquiry (J)** se usa de manera similar al comando **inquiry (I)**.

IV. DESARROLLO

1) Conecte el puerto RS232C de la computadora con el conector identificado con **COMPUTER** sobre el controlador, empleando el cable apropiado.

2) Ejecute la rutina de encendido propuesta en la sección de teoría, al final de ésta se cargará el editor de Basic (**BASICA**) que será el lenguaje a utilizar y el robot deberá estar listo para recibir comandos y ejecutarlos.

3) Ya estando en Basic, pruebe correr el siguiente programa:

```
10   OPEN"COM1:9600,E,7,2,CS,DS,CD",AS#1
20   PRINT#1,"F+50"
30   END
```

Explique su funcionamiento. Ahora, cambie el signo en la línea 20 y vuélvalo a ejecutar. Finalmente, cambie en la misma línea 20 la letra tal que pueda mover a los motores **B** a **F**, uno a la vez. Anote sus observaciones.

4) Cargue y ejecute el siguiente programa con el que se pretende mover al motor **F** 50 pasos en dirección positiva y luego, regresarlo a su posición original:

```
10    OPEN"COM1:9600,E,7,2,CS,DS,CD",AS#1
20    PRINT#1,"F+50"
30    PRINT#1,"F-50"
40    END
```

¿Qué observa?. ¿Se efectúa la función esperada?. Explique a qué se debe la falla en la rutina anterior.

5) Diseñe y pruebe un programa que desempeñe la función esperada en el inciso anterior, para ello, use ahora el comando **?(QUESTION)** de manera que se verifique que el primer movimiento se concluya antes de que se mande ejecutar el otro y que este último se termine antes de finalizar el programa. Se recomienda emplear el siguiente esquema:

```
. . .
. . .
mm    PRINT#1,"F?";:GOSUB NN
. . .
. . .
nn    IF LOC(1)=0 THEN nn ELSE W$=INPUT$(LOC(1),#1)
      W=ASC(W$)-32
      RETURN
. . .
. . .
```

Explique detalladamente su funcionamiento.

6) Debido a problemas de comunicaciones y al tamaño de los registros de error, como se explicó en la sección teórica, el valor máximo que se puede enviar de pasos de movimiento para un motor es 95, esto nos llevaría

a pensar que 95 es el número máximo de pasos que el motor se mueve por cada comando, por lo que se observa una discontinuidad si se requiriera un movimiento mayor; sin embargo, existe la manera de ejecutar movimientos de más magnitud y en forma continua. A continuación se presenta un programa que ordena un movimiento de 500 pasos del motor F y en dirección positiva.

Ejecute el programa y explique detalladamente su funcionamiento.

```
10   OPEN"COM1:9600,E,7,2,CS,DS,CD"AS#1
20   FOR I=1 TO 10
30        PRINT#1,"F+50"
40        PRINT#1,"F?";:GOSUB 80
50        IF W>45 THEN 40
60   NEXT
70   END
80   IF LOC(1)=0 THEN 80 ELSE W$=INPUT$(LOC(1),#1)
90   W=ASC(W$)-32
100  RETURN
```

7) Otra forma de efectuar la misma tarea es con el siguiente programa:

```
10   OPEN"COM1:9600,E,7,2,CS,DS,CD"AS#1
20   N=500
30   H=95
40   IF N<=K GOTO 80
50   PRINT#1,"F+";STR$(H):N=N-H
60   PRINT#1,"F?";:GOSUB 100:H=95-W
70   GOTO 40
80   PRINT#1,"F+";STR$(N)
90   END
100  IF LOC(1) THEN 100 ELSE W$=INPUT$(LOC(1),#1)
110  W=ASC(W$)-32
120  RETURN
```

Ejecute este programa y explique su funcionamiento. Determine las ventajas que este último ofrece respecto al primero.

8) Haga más general el programa anterior de tal forma que el usuario indique el motor, la dirección y la cantidad de pasos a mover. Pruébelo procurando dar valores menores a 200 y letras de la **B** a la **F**.

9) Con la rutina **Go hard home**, el controlador hacía girar a las uniones del brazo hasta que localizaba el cierre de los interruptores colocados en cada una de ellas posicionando, finalmente, a cada una de las articulaciones de tal forma que dichos interruptores se mantuvieran oprimidos, de esta manera se lograba alcanzar una posición física única a la que se le denominó "Hard home".

Usando el comando **J (INQUIRY)** localice la posición de "Hard home" para la rotación de la muñeca (motor **B**).

10) Realice un programa en Basic que haga la transferencia de un objeto de una cierta posición a otra, siempre regresando a una posición original. Se recomienda usar la rutina implementada en el inciso no. 8.

V. CUESTIONARIO

1. ¿Qué ventajas y desventajas ofrece manejar al robot a través de una computadora?

2. ¿Sería posible implementar un programa tal que permitiera que el robot realizara tareas en forma general sin modificarlo? Explique.

3. ¿Por qué se debe limpiar el buffer de movimiento antes de ejecutar algún otro comando?

VI. BIBLIOGRAFIA

1. RHINO Robots Inc. Owner's manual. Champaign Illinois.

ANEXO 1

RESUMEN DE COMANDOS DE CC

CC

```
CC and DIG commands:
  CC>  prompt indicates analog mode (Laplace transforms)
  DIG> prompt indicates digital mode (z transforms)

  ENTER       [[,Gi][,num [,den]]]          Enter transfer function Gi, n>=0
  DELAY       [,time]                       Multiply tf by exp(-sT) delay
  DISPLAY     [,Gi]                         Display tf
  PZF         [,Gi]                         Display tf in pole-zero-form
  TCF         [,Gi]                         Display tf in time constant form
  SHORTHAND   [,Gi]                         Short hand form (complex= zeta,mag)
  SINGLE      [,Gi]                         Display tf as ratio of single polys
  UNITARY     [,Gi]                         Display all polynomials as unitary
  PFE         [,Gi]                         Partial fraction expansion
  ILT         [,Gi][,CAUSAL/,ANTI/,roc]     Inverse Laplace transform
  IZT         [,Gi][,CAUSAL/,ANTI/,roc]     Inverse z transform
  LPDISPLAY   [,Gi][,'message']             Line printer display
  DPDISPLAY   [,Gi]                         Double precision display
  RECALL      [[,Gi][,filename]]            Recall Gi from diskfile
  STORE       [[,Gi][,filename]]            Store Gi on diskfile

Plot commands (parameters shown for automatic scaling):
  CONVERT     [,Gi,Gj,options,time]         Convert Gi(s) to Gj(z), 10 options
  WPLANE      [,Gi]                         Convert Gi(z) to Gi(s)
  STABILITY   [,Gi]                         Unity feedback closed loop stabilit
  ROUTH       [,Gi]                         Range of stable gains
  FREQUENCY   [[,Gi],low,high,#pts,type]    Create frequency file
  FREQUENCY   [[,Gi],SP,low,#pts,#reps,type] Create special digital freq file
  LIST        [,ALL/,INFO/,low,high]        List freq file
  POINT       [[,Gi],freq]                  Compute Gi at freq
  BODE        [,type,AUTO]                  Bode, all plots for Gi used in FREQ
  NYQUIST     [,AUTO]                       Nyquist plot
  INVERSE NYQUIST  [,AUTO]                  Inverse Nyquist
  LOG NYQUIST      [,AUTO]                  Nyquist with log axii
  NICHOLS     [,AUTO]                       Nichols chart
  TIME        [[Gi],type,AUTO]              Time response
  DTIME       [[,Gi],limits,'title',options] Digital response (no auto entry)
  ROOT LOCUS  [,INFO/,AUTO]                 Root locus

Commands with their own subcommand language:
  BUILD          Build up transfer functions
  MACRO          Editor for macros (indirect command files)
  STATE          State space input of transfer functions
  ZEROS          Zeros of polynomials
  MR             Multi-rate frequency plots
  LQG            Linear quadratic Gaussian problem
  MIMO           Transfer function matrices
```

```
Other commands
  ANALOG                                Switch to analog mode
  DIGITAL       [,time]                 Switch to digital mode
  ECHO          [,message]              Print message for macros
  PAUSE         [,TIME,secs]            Pause, option for timed pause
  SETUP                                 Setup of screen and of disk file locations
  HELP                                  Display this help message
  QUIT                                  Quit
  CLS                                   Clear screen
  FILES         [,wildcards]            List disk files
  KILL          [,filename]             Kill disk file
  HARDCOPY                              Screen dump, code must be loaded
  CLOCK         [,ANA/DIG][,COLOR]      Display clock
  CALCULATOR                            HP calculator
  NONSENSE                              Toggles nonsense graphics flag
  TIC-TAC-TOE                           Nonsense, beat the computer!

Plot commands (parameters shown for manual scaling)

BODE [,type,freq,mag,phase[,'title',back,fore]]
     type=1-7 (1=|G|, 2=Arg[G], 3=1&2, 4=|1+G|, 5=1&4, 6=|1+G^-1|, 7=1&6)
     freq=type,low,high,#divs or type,AUTO
          type=0(log10),=1(linear), include #divs only for linear
     mag=type,low,high,#divs or type,AUTO  (skip for Bode type 2)
          type=0(log10),=1(linear),=2(dB), include #divs for linear and dB
     phase=low,high,#divs or AUTO  (include only for Bode 2,3)
     title=ASCII title, enclose in parentheses, default is no title
     back=0(lines),=1(hatch marks)  default is lines
     fore=0(lines),=1(points)  default is lines
NYQUIST [,real,imag[,'title',back,fore]]  (title,fore same as Bode)
     real,imag=low real, high real, #divs, low imag, high imag, # divs
     back=0(lines),=1(hatch marks),=2(circles)
INVERSE NYQUIST [,real,imag[,'title',back,fore]]  (same as NYQUIST)
NICHOLS [,mag,phase[,'title',back,fore]]  (same as BODE)
LOG NYQUIST [,real,imag[,'title',back,fore]] (title,back,fore same as NYQUIST)
     real=r1,r2,r3,i1,i2,i3(all entered log10)  or AUTO
       negative real axis from -10^r1 to -10^r2, positive from 10^r2 to 10^r3
       negative imag axis from -10^i1 to -10^i2, positive from 10^i2 to 10^i3

TIME [[,Gi],type,time,vertical[,'title',back,fore]]
     type=1(c.l. step),=2(c.l. impulse),=3(o.l. step),=4(o.l. impulse
         =5(noncausal o.l. impulse)
     roc (region of convergence) for type=5
     time=low,high,#divs,#points  or  AUTO
     vertical=low,high,#divs  or  AUTO
     title=ASCII title, enclose in quotations, default is no title
     back=0(lines),=1(hatch marks)  default is lines
     fore=0(lines),=2(points)  default is lines
DTIME [[,Gi],type,time,vertical[,'title',back,fore]] (title,back same as TIME
     type=1(c.l. step),=2(c.l. impulse),=3(o.l. step),=4(o.l. impulse)
     time type=0(samples),=1(time)
     samples=high sample,increment for plotting,#axis divs  no AUTO scaling
     time=high time,increment for plotting,# axis divs  no AUTO scaling
     vertical=low,high,#divs  no AUTO scaling
     fore=0(lines),=1(points),=2(spikes)
ROOT LOCUS [[,Gi],real,imag[,'title',back,fore]]  (title,back,fore same as TIME)

     real,imag=low real,high real,#divs,low imag,high imag,#divs  or AUTO
```

Rules for entering commands and parameters:

1. Use small or capital letters, spaces are ignored.
2. Only the minimum number of characters to be non-ambiguous are needed.
3. If no parameters are entered then they are prompted (NOVICES)
4. Parameters can be entered on command line separated by commas (EXPERTS).
5. All parameters except those with defaults must be included.
6. Enclose a parameter in quotes to retain small letters and spaces
7. Each command line may have several commands, separated by &'s.
8. Commands on different command levels can be mixed.
9. The command HELP always lists available commands.
10. The command QUIT always branches to next highest command level
11. Macros are entered by preceding their name with @.
12. Macros can only be started from the CC or DIG command levels.
13. Maximum length of a command line is 256 characters, to enter a transfer function requiring more characters use BUILD to piece it together.

Further remarks:

1. Many temporary files are stored on disk: $$SETUP,$$Gi,$$Hi,$$Pi,$$FUN, $$HIS. They remain after program quits, and are accessed when restarted if available. Default of $$SETUP is created if not available.
2. Once a command starts prompting or executing the only way to abort is to break out of the program. This is not so inelegant as may first appear, because of the temporary files which remain on disk.
3. Other temporary files are deleted before quitting the program, e.g. $$TEMP,$$PZF,$$RES,$$ILT,$$GFREQ. They remain on disk only if execution is broken while open. No attempt is made to access them after restart.
4. Notation used in HELP files:
   Gi = G, G1, G2, ..., where i = integer >=0
   [.] = parameter which is prompted is not included on command line
   (.) = parameter which has a default value
   'message' = parameter which should be enclosed in (double) quotes
   [CAPITAL] = parameter which is entered in ASCII as shown
   usually only the first letter is checked for
   [small] = parameter entered as a number according to the description
   [A/B/small] = enter either A, B, or a number
5. Internal errors (such as disk full, divide by zero, program not present, array dimension out of bounds, and so on) cause a message to be printed. The program will restart at the appropriate command prompt.
6. No hardcopies? Remember to load a screen dump program before starting CC.

Description of CC programs:

CC = root program, initialization, command processor, chains other programs ANALOG, DIGITAL, ECHO, PAUSE, CLS, QUIT, KILL, FILES, NONSENSE, HARDCOPY
CC1 = Macro commands
CC2 = Enter and displays, inverse transforms, disk access
CC3 = Newton-Raphson and partial fraction expansion calculations
CC4 = BUILD commands
CC5 = CONVERT, WPLANE
CC6 = STABILITY, ROUTH, ZERO commands, some BUILD and MR commands
CC7,CC8,CC9 = frequency plots and MR commands
CC10 = TIME, DTIME
CC11 = ROOT LOCUS
CC12 = Help for CC, BUILD, and MR
CC13 = CALCULATOR, CLOCK, TIC-TAC-TOE
CC14-CC16 = LQG commands
CC17-CC19 = MIMO commands
CC20-CC29 = STATE commands

BUILD

```
BUILD commands for creating new transfer functions:
 1POLE        [[,Gi][,a]]                          Single pole
 2POLES       [[,Gi][,wn,zeta]]                    Two poles
 1ZERO        [[,Gi][,a]]                          Single zero
 2ZEROS       [[,Gi][,wn,zeta]]                    Two zeros
 LEADLAG      [[,Gi],[freq,phase]]                 Leadlag, -90 to 90 phase at freq
 NOTCH        [[,Gi],[wn,zetan,zetad,a]]           Notch at wn, depth =,zetan/zetad
 INTEGRATOR   [,Gi]                                Integrator
 BUTTERWORTH  [[,Gi],[freq,order]]                 Butterworth
 BESSEL       [[,Gi],[freq,order]]                 Bessel
 CHEBYSHEV    [[,Gi],[freq,dB ripple,order]]       Chebyshev, 0-3dB passband ripple
 ITAE         [[,Gi],[freq,order]]                 ITAE lowpass filter
 PADEDELAY    [[,Gi],[time,order]]                 Pade approx of exp(-s*time)

BUILD commands for manipulating existing transfer functions:
 CHANGE       [,Gi,parameters] Change coefficients or eliminate polynomials
 CANCEL       [,Gi]            Cancel polys with same coefficients
 SCALE        [,Gi,Gj,alpha]   Gj(.)=Gi(alpha*.)
 SUBSTITUTE   [,Gi,Gj,Gk]      Gk(.)=Gi(Gj(.))
 SF           [,Gi,Gj,POS/NEG[,division]]]   Gj=Spectral Factors of Gi
 CHPZF        [,Gi,Gj]         Gj=Pole Zero Form of Gi
 CHTCF        [,Gi,Gj]         Gj=Time Constant Form of Gi
 CHPFE        [,Gi,Gj]         Gj=Partial Factors of Gi
 CHUNITARY    [,Gi,Gj]         Gj=Unitary Form of Gi
 CHSINGLE     [,Gi,Gj]         Gj=Single polynomial form of Gi
 NUMERATOR    [,Gi,Gj]         Gj=Numerator of Gi
 DENOMINATOR  [,Gi,Gj]         Gj=Denominator of Gi
Interchange and equations:
 Gi # Gj
 Gi = algebraic combination of Gi +-*/^() s and numbers
BUILD commands same as CC commands:
 ENTER      PFE          LPDISPLAY    ECHO     TCF
 DISPLAY    SHORTHAND    DPDISPLAY    PAUSE    UNITARY
 PZF        SINGLE       ANALOG       HELP
```

MACRO

Commands for creating macros:
Use nonambiguous portion, only 1 per command line, parameters must be present

ADD,n	enter edit mode, add lines after n, after end if n not present, exit by a period at beginning of a new line
DELETE,n,m	delete lines from n to m
HELP	list commands
HISTORY	add history file to end of macro
LIST,n,m	list lines n to m, no parameters to list entire macro, just n parameter to list from n to end
NEW	delete all lines, prompts to be sure
QUIT	return to CC command level, prompts if changes not stored
RECALL,filename,n	recalls filename from disk, .MAC added if . not present, added after line n, or after end if n not included
REPLACE,n,m,string1;string2	replace string1 with string2 from line n to m all parameters must be present, note the semi-colon.
SEARCH,n,string	search for string starting at line n, if string missing then previous used, search wraps around
STORE,filename	store resident macro in diskfile, extension .MAC added
TRUNCATE,n	truncate history file to n previous commands

Macro execution instructions:
The command @MACRO,&1,&2,...,&n executes the commands in MACRO.MAC.
Macros are called from CC command level, and must be alone on command line.
They are expanded and parameters are inserted before execution, stopping at 500 lines. Recursive calls are possible, which will use all 500 lines.

MR

MR (Multi-rate) commands:

FUNCTION	[,Fn]	Define function
FDISPLAY	[,Fn/,ALL]	Display function
FREQUENCY	[[,Fn][,low,high freq,#points]]	Create freq file
FREQUENCY	[[,Fn],SPECIAL,low freq,#point,#repeats]	Create special freq file
LIST	[,ALL/,INFO/,low,high freq]	List freq file
POINT	[[,Fn],freq]	Compute function at freq
BODE	\	
NYQUIST	\ Same parameters as in CC command level,	
INVERSE NYQUIST	> but	
LOG NYQUIST	/ uses the function that created freq file	
NICHOLS	/	
SKIP [,Gi,Gj,n]	Skip sampling algorithm, every nth sample chosen	
EXPAND [,Gi,Gj,n]	Samples spaced n apart with zero fill (z^i chgd to z^n*i)	
REPLACE [,Gi,Gj,n]	z^n*i replaced with z^i, other terms assumed zero	

MR commands same as CC commands:

ENTER	PFE	LPDISPLAY	ECHO	CONVERT
DISPLAY	SHORTHAND	DPDISPLAY	PAUSE	UNITARY
PZF	SINGLE	ANALOG	HELP	

ZEROS

```
Zeros commands:
 NUMERATOR    [,G?]        Compute roots of numerator
 DENOMINATOR  [,G?]        Compute roots of denominator
 VERIFY                    Verify roots
 NEWTON RAPHSON            Compute roots by Newton-Raphson iterations (default)
 EIGENVALUE                Set up eigenvalue problem and use QZ algorithm
 HELP [,LP]                List commands, LP option for lineprinter listing
 QUIT                      Return to CC level

 Enter the polynomial: a(n)*s^n + a(n-1)*s^(n-1) +  ... + a(1)*s + a(0)
 as follows: n, a(n), a(n-1), ..., a(1), a(0)
```

STATE

```
STATE commands for the following state space systems:
 Pi = [ A B ]    Hi(s) = C*inv(s*I-A)*B + D
      [ C D ]

STATE commands for entering, displaying, and changing state space systems Pi:
 PENTER      [[Pi,#states,#inputs,#outputs],[coeffs]] Enter system matrix Pi
 PCHANGE     [parameters as prompted]        Change system matrices
 PDISPLAY    [,Pi,[ALL/A/B/C/D/ELEMENTS]]    Display Pi, default is ALL
 PLPDISPLAY  [,Pi,'message'/ELEMENTS]        Line printer display of Pi
 PSTORE      [,Pi,filename]                  Store system matrix Pi on disk
 PRECALL     [,Pi,filename]                  Recall Pi from disk
 DIM         [,Pi]                           Print dimensions of Pi
 EPSILON     [,Pi[,eps]]                     Set coeffs < eps to 0, default=1e-16

STATE Commands for transfer function matrices Hi:
 HDISPLAY    [,Hi]                 Display Hi
 HLPDISPLAY  [,Hi,'message']       Line printer listing of Hi
 HSTORE      [,Hi,filename]        Store Hi on disk
 HRECALL     [,Hi,filename]        Recall Hi from disk
 EXTRACT [,Hi,Gi,(row,col)/ALL] Gi=Hi(row,col) or copy all Hi() to Gi,Gi+1,...

STATE commands for building state space systems:
 IDENTITY          [[,Pi],dim[,alpha]]     Pi = alpha*I
 DIAGONAL          [[,Pi],dim,elements]    Pi = diag[elements]
 HP                [,Hi][,Pi]              Pi = constant tf matrix Hi
 PACK              [,Pi,Pj,Pk,Pl,Pm]       Pi = (Pj,Pk;Pl,Pm)
 UNPACK            [,Pi,Pj,Pk,Pl,Pm]       Pj=A, Pk=B, Pl=C, Pm=D of Pi(A,B;C,D)
 AUGMENT           [,Pi,Pj,Pk,row,col]     Pk = Pi augmented with Pj
 SELECT   [Pi,Pj,S/I/R/O/C,selections]     Pj=selected parts of Pi
 CCF               [,Gi][,Pi]              Pi = controllable canonical form of Gi
 OCF               [,Gi][,Pi]              Pi = observable canonical form of Gi
 DCF               [,Gi][,Pi]              Pi = diagonal canonical form of Gi
 ADD               [,Pi,Pj,Pk]             Pk = Pi+Pj
 SUBTRACT          [,Pi,Pj,Pk]             Pk = Pi-Pj
 MULTIPLY          [,Pi,Pj,Pk]             Pk = Pi*Pj
 SCALE             [,Pi,Pj,alpha]          Pj = alpha*Pi
 FEEDBACK          [,option,Pi,Pj,Pk,n]    Pj or Pk = Pi or Pi,Pj with feedback
                                           6 options
 TRANSPOSE         [,Pi,Pj]                Pj = Pi(A',C';B',D')
 ADJOINT           [,Pi,Pj]                Pj = Pi(-A',C';-B',D')
 SUBSTITUTE        [Pi,Pj,alpha]           Pj = Pi(A/alpha,B/alpha;C,D)
 SIMILARITY        [,Pi,Pj,Pk]             Pk = Pi with similarity transform Pj
 SIMILARITY,EIGENVECTOR [,Pi,Pj]           Pj = Pi with eigenvector sim. trans.
 RESIDUALIZATION [,Pi,Pj,n]                Pj = residualized Pi, keep n states
```

```
Algorithms for the state space matrices Pi:
 BALANCE [,Pi,INFO]    Balance system matrices, INFO prints information
 FADEEVA [,Pi][,Hi]    Compute Hi(s) by Fadeeva's method
 QZ [,Pi,Hi][,thresh] Compute Hi(s) by QZ algorithm (eigenvalue and generalized
                       eigenvalue problems), thresh determines zeros at infinity
 POLES [,Pi]           Poles of Pi = eigenvalues of A of Pi
 ZEROS [,Pi]           Multivariable zeros of Pi, by QZ algorithm
 EIGENVALUE [,Pi,Pj[,Pk]]  Pj=e.values of Pi, Pk=aesociated e.vectors
 CONTROLLABILITY[,Pi] Determine controllability and obsevability
 OBSERVABILITY [,Pi]  Determine controllability and obsevability
 NUMERATOR [,Pi]       Coupling numerators of Pi, by QZ algorithm
 EXPONENTIAL [,Pi,Pj,T[,n]] Pj = discretized Pi using matrix exponential
                       Taylor series with normalizing, n=#terms, default=20
 PEXPONENTIAL [,Pi,Pj,T,[,n]]  Pj=n'th order Pade approx of matrix exp, def=6
 CONVERT [Pi,Pj,option,time]  Pj = discretized Pi using integration techniques
                       Foward and backward rectangle, bilinear, prewarped Tustin
 WPLANE [Pi,Pj]        Pj = W-plane version of digital Pi
 WPRIME [Pi,Pj]        Pj = W-plane version of analog Pi
 INVERSE [Pi,Pj]       Pj = Inverse of Pi, D of Pj must be invertible
 POLE PLACEMENT [Pi,Pj,option]  Pj = result of pole placement for SISO Pi
                       Pj = full state feedback, observer gains, or combination

STATE commands for time responses:
 SIMULATION [,Pi,input,initial condition,time]
    Pi.Y = Output of simulation of system Pi
    input options:
       1,m,height = step input at m'th input
       2,m,area   = impulse at m'th input
       3,m,height,limit = pulse at m'th input
       4,m,sigma,seed = Gaussian white noise input at m'th input
           std.dev.=sigma, random seed uses timer if seed<=0
       5,filename = input from file
       6 = no input
     initial condition options: 1 = zero, 2,Pj = (constant) system Pj
     time = tmax,delta,nskip, where delta=integration step
     6th order Pade approx of matrix exponential used for integration
 DSIMULATION [,Pi,input,initial condition,time], same except for:
     input option: 2,m,height = single pulse at n=0
     time = max sample, nskip, output every nskip samples
 INPUT   creates input file used for simulations, prompts only
 PLOT [,Pi,outputs,limits,'title',options]  Plot of Pi.Y
     outputs = single output or ALL
     limits = tmax,#divs,ymin,ymax,#divs or AUTO
     options = back (0=lines,1=hatches), fore (0=lines,1=points,2=spikes
 PLOT [,Pi,ALL,AUTO]   or  PLOT [,Pi,A,A]  =  short form
```

```
STATE commands for optimal control:
 RICCATI [,Pi,Pj,Pk,Pm]          Pm = Solution of Riccati equation
 LQR     [,option,parameters]    Linear quadratic regulator problem, 4 options
 KBF     [,option,parameters]    Kalman Bucy Filter problem, 4 options
 LQG     [,option,parameters]    Linear quadratic Gaussian problem, 4 options

STATE commands same as BUILD commands, except state space systems are created
 1POLE       [[,Pi][,a]]                       Single pole
 2POLES      [[,Pi][,wn,zeta]]                 Two poles
 LEADLAG     [[,Pi],[freq,phase]]              Leadlag, -90 to 90 phase at freq
 NOTCH       [[,Pi],[wn,zetan,zetad,a]]        Notch at wn, depth = zetan/zetad
 BUTTERWORTH [[,Pi],[freq,order]]              Butterworth
 BESSEL      [[,Pi],[freq,order]]              Bessel
 CHEBYSHEV   [[,Pi],[freq,dB ripple,order]]    Chebyshev, 0-3dB passband ripple
 ITAE        [[,Pi],[freq,order]]              ITAE lowpass filter
 PADEDELAY   [[,Pi],[time,order]]              Pade approx of exp(-s*time)

State commands same as CC commands:
 ENTER     PFE          LPDISPLAY   ECHO    FILES
 DISPLAY   SHORTHAND    DPDISPLAY   PAUSE   KILL
 PZF       SINGLE       ANALOG      HELP
```

ANEXO 2

RESUMEN DE COMANDOS DE SIMNON

COMANDOS DE SIMNON

AREA	ASHOW	AXES	DEFAULT	DISP	EDIT	END
ERROR	FOR	FREE	GE	GIN	GOTO	HCOPY
HELP	IF	INIT	LABEL	LET	LIST	LP
MACRO	MARK	NEXT	PAR	PLOT	PRINT	READ
RESUME	SAVE	SHOW	SIMU	SPLIT	STOP	STORE
SUSPEND	SWITCH	SYST	TEXT	TURN	WRITE	

Library Functions

The following library functions are provided.

ABS(x)	Absolute value of x.
ARCCOS(x)	Arc cosine of x.
ARCSIN(x)	Arc sine of x.
ATAN(x)	Arc tangent of x ($-\pi/2 <$ ATAN $< \pi/2$).
ATAN2(x,y)	Arc tangent of x/y ($-\pi <$ ATAN2 $< \pi$).
COS(x)	Cosine of x (x in radians).
COSH(x)	Hyperbolic cosine of x.
EXP(x)	Exponential function e^x.
INT(x)	Integer part of x.
LN(x)	Natural logarithm of x ($x > 0$).
LOG(x)	Base 10 logarithm of x ($x > 0$).
MAX(x,y)	Maximum of x and y.
MIN(x,y)	Minimum of x and y.
MOD(x,y)	The remainder when dividing x by y.
NORM(x)	Normal distribution with mean value 0 and standard deviation 1.
RECT(x)	Rectangular distribution in the interval [0,1].
SIGN(x)	The sign of x, = −1 for negative numbers, else 1.
SIN(x)	Sine of x (x in radians).
SINH(x)	Hyperbolic sine of x.
SQRT(x)	Square root of x ($x > 0$).
TAN(x)	Tangent of x (x in radians).
TANH(x)	Hyperbolic tangent of x.

Examples

wave=a*SIN(w*b+p)	*Harmonic function*
x=MIN(MAX(x,xmin),xmax)	*Make* $xmin \leq x \leq xmax$

Summary of Simnon Commands

A list of the commands, their syntax, and brief descriptions are given below. Some Intrac commands are also included in the list. More details about the commands are obtained using the command HELP. The list is valid for implementations on VAX 11/780 systems. The following notation is used.

{op1 | . . . | opn}	Defines different alternatives, of which one must be given.
[. . . .]	Parts within squared brackets are optional and can be omitted.
[. . . .]*	A star indicates that the previous part can be repeated.
⟨. . . .⟩	Defines arguments for the commands.

ALGOR {HAMPC|RK|RKFIX|DAS}. To select integration routine.

HAMPC	Hamming predictor corrector (default).
RK	Runge-Kutta variable step size.
RKFIX	Runge-Kutta fixed step size.
DAS	Integration routine for stiff systems.

AREA ‹rows› ‹colums›. To define the plotting area to be used next (see SPLIT).

ASHOW. To plot stored variables with automatic scaling of axes. For syntax, see SHOW.

AXES [‹axis spec› [‹axis spec›]]. To draw axes.

⟨axis spec⟩:= {H|V} ⟨min value⟩ ⟨max value⟩
H, horizontal; V, vertical

DISP [({DIS|TP|LP}{FF|LF})] [‹variable›]*. To display variables.

DIS	Display (default).
TP	Terminal.
LP	Line printer.
FF	Form feed (default when no variables are specified).
LF	Line feed (default when variables are specified).

If no variables are given, all variables are displayed.

EDIT ‹filename›. To edit a file. The editor has two modes, INPUT and EDIT. The mode is changed by entering an empty line. In the edit mode it is possible to change the current file. The following commands are available:

A[PPEND] ⟨string⟩	Append string to line.
B[OTT]	Line pointer to bottom.
C[HANG] /x/y/	Replace string *x* by string *y*.
D[EL] ⟨integer⟩	Delete *n* lines.
DIS [ON|OFF]	Echo check enable/disable.
E[XIT]	Leave editor with update.
F[IND] ⟨string⟩	Find string beginning line.
I[NS] ⟨string⟩	Insert a line below current line.
L[OC] ⟨string⟩	Locate string anywhere in a line.
LEAVE	Leave the editor without update.
N[EXT] ⟨integer⟩	Move line pointer down *n* lines.
O[VERL] ⟨integer⟩	Overlay *n* lines by new text.
P[RINT] ⟨integer⟩	Print *n* lines.
R[ETYP] ⟨string⟩	Retype current line.
T[OP]	Move line pointer to top.

ERROR ‹error bound›. To choose error bound for integration routine. Default value is 0.001.

GET ‹filename›. To get parameter values and initial values from a file that previously has been stored using SAVE.

HCOPY [‹scale factor›] "‹comment›. Make a hard copy of curves on display. The hard copy is scaled with ⟨scale factor⟩, which may be in the interval (0.5, 1.6). The comment is obtained on the hard copy, too.

HELP [{‹command›|EDIT|INTRAC|SIMNON|SYSTEMS}]. To get more information about the commands, the editor, Intrac, Simnon, and the standard systems. A menu of commands is obtained by typing HELP.

HELPDEV=[LP|TP|DIS]. To determine the output device for HELP output (see DISP). Line-printer output is first written onto a print file and printed using the command LP.

INIT ‹state variable›:{‹number›|‹variable›}. To change the initial value of a state variable.

LET ‹variable› = ‹number›. An Intrac command that also is used to set parameters in the standard systems.

LIST [({DIS|TP|LP}[FF|LF])]{‹filename›}*. To list files. The same as DISP but for files—e.g., subsystems.

LP. Initiate listing on line printer of variables and files that have been ordered by DISP, LIST, or PRINT commands.

MARK. To introduce text into a plot. For syntax, type HELP INTRAC MARK.

NEWS. To obtain news about the simulation package.

PAR ‹parameter› : [‹number›|‹variable›]. To change a parameter value.

PLOT [[‹variable›]•[(‹variable›)]]. To select variables to be plotted when making the command SIMU. Examples: PLOT X1 X2 gives X1 and X2 as functions of time, while PLOT X1(X2) gives X1 as function of X2.

PRINT [([DIS|TP|LP] [FF|LF])] ‹filename› [‹lines›] [/‹start time›]. To list file generated with the commands STORE+SIMU.

‹lines› lines starting from ⟨start time⟩ will be printed. The other parameters are the same as for DISP.

SAVE ‹filename› [‹systemid›] [–[PAR|INIT]]. To save parameter values and initial values on a file named ⟨filename⟩ to be used by the command GET. If only parameters or initial values should be saved, the options PAR and INIT, respectively, should be used.

SHOW [[‹start› ‹stop›] [‹variable›]• [(‹variable›)] [–MARK]|–LIST] [/‹filename›]. To plot stored variables from file ⟨filename⟩. To be used with the command STORE. The specified variables are plotted from ⟨start⟩ to ⟨stop⟩ time. If MARK is used, the different variables are numbered on the plot. The LIST option lists the names of all variables.

SIMU [‹start time› ‹stop time› [‹increment›] [–‹option›]
[/‹filename› [‹increment›]]
‹option›:=[MARK|CONT]

To simulate the system from ⟨start time⟩ to ⟨stop time⟩ using the maximum step size ⟨increment⟩ [default (stop time – start time)/100]. Using MARK, the variables defined by PLOT are numbered. With CONT, the simulation is continued with the previously obtained state variables as initial values. When specifying ⟨filename⟩, the plotted variables are stored in ⟨filename⟩ with ⟨increment⟩ as sampling interval.

SPLIT ‹rows› ‹colums›. To split the screen into maximum six plotting areas.

⟨rows⟩:=[1|2|3] default 1
⟨columns⟩:=[1|2] default 1

STATE. Use with the integration routine DAS. For syntax, write HELP STATE.

STOP. To leave Simnon.

STORE [‹variable›]• [–ADD]. Use to select variables to be stored at each simulation. With ADD, new variables can be added to a previously defined list of

variables. The variables can be displayed using ASHOW or SHOW and printed using PRINT.

SWITCH ‹option› {ON|OFF}
‹option›:= {CLOCK|DATE|ECHO|EXEC|LOG|TRACE}
To control the execution of Intrac.

CLOCK	**Adds time to hardcopy output. Default OFF.**
DATE	**Adds date to hardcopy output. Default OFF.**
ECHO	**Macro commands are echoed. Default OFF.**
EXEC	**Commands executed while typed during macro generation. Default OFF.**
LOG	**Executed commands in macro logged on line printer. Default OFF.**
TRACE	**Affects ECHO and LOG.**

SYST [‹subsystem›]+ [−‹option›] [/‹filename›]
‹option›:=[EDIT|EXIT]
To define the system. The subsystems are compiled. If there are several subsystems, the last one has to be a connecting system. EDIT means that the compiler goes into the editor for each file. If filename is specified, the sorted equations are written into a text file.

TEXT '‹any string not containing single quote›'. To include text on paper plot.

TURN ‹option› {ON|OFF}
‹option›:= {S125|DARK|DIS|OVFLO}
To turn on and off switches.

S125	**Selects a limited set of scale factors (1., 2., 5.) when choosing scales on axes. Default OFF.**
DARK	**ON means that plotted curves will not be connected between the sampling instants. Default OFF.**
DIS	**Informs SIMNON if the user has a graphic display. Default ON.**
OVFLO	**ON means that the simulation stops if overflow occurs in the calculations. Default OFF.**

References

ELMQVIST, H. (1975): Simnon-user's guide. Department of Automatic Control, Lund Inst. of Techn., CODEN: LUTFD2/(TFRT-3091)/(1975).

ELMQVIST, H. (1977): SIMNON: An Interactive Simulation Program for Nonlinear Systems. Proceedings Simulation '77, Montreux 1977.

ÅSTRÖM, K. J. (1982): A Simnon Tutorial. Department of Automatic Control, Lund Inst. of Techn., CODEN: LUTFD2/(TFRT-3168)/(1982).

ANEXO 3

RESUMEN DE COMANDOS DE MATLAB

Directorio de archivos M en c:\matlab

acosh	angle	asinh	atanh	bar	bench	bessel
bessela	besselh	besseln	blanks	cdf2rdf	census	cla
compan	computer	cond	conv	conv2	corr	cosh
ctheorem	dato	deci	deconv	demo	demolist	dft
diff	edit	eigmovie	esti	esti3	etime	expm1
expm2	expm3	ferr	feval	fft2	fftshift	fitdemo
fitfun	flipx	flipy	fun	funm	gallery	gamma
gpp	hadamard	hankel	hilb	hist	histogra	humps
idft	ieee	ifft	ifft2	info	ini	int2srt
invhilb	isempty	kron	length	log10	logm	logspace
matdemo	matlab	mean	median	membrane	menu	meshdemo
meshdom	mkpp	movies	nademo	nelder	neldstep	nnls
null	num2srt	ode23	ode45	odedemo	orth	pinv
plan	plotdemo	poly	polyfit	polival	polivalm	ppval
print	quad	quaddemo	quadstep	rank	rat	ratmovie
readme	residue	roots	rot90	rref	rrefmovi	rsf2csf
senc	sinh	spline	sqrtm	square	startup	std
table1	table2	tanh	toeplitz	trace	translat	tril
triu	unmkpp	vdpol	why	wow	zerodemo	zeroin

Funciones interconstruidas de Matlab

intro	<	chol	end	function	ltitr	qr	sort
help	>	clc	eps	global	lu	quit	sprintf
demo	=	clear	error	grid	macro	qz	sqrt
[	&	clg	eval	hess	magic	rand	startup
]	|	clock	exist	hold	max	rcond	string
(	~	conj	exit	home	memory	real	subplot
)	abs	contour	exp	ident	mesh	relop	sum
.	all	cos	expm	if	meta	rem	svd
,	ans	cumprod	eye	imag	min	return	tan
;	any	cumsum	feval	inf	nan	round	text
%	acos	delete	fft	input	nargin	save	title
!	asin	det	filter	inv	norm	schur	type
:	atan	diag	find	isnan	ones	script	what
'	atan2	diary	finite	isstr	pack	semilogx	while
+	axis	dir	fix	keyboard	pause	semilogy	who
-	balance	disp	floor	load	pi	setsrt	xlabel
*	break	echo	flops	log	plot	shg	ylabel
\	casesen	eig	for	loglog	polar	sign	zeros
/	ceil	else	format	logop	prod	sin	
^	chdir	elseif	fprintf	ltifr	prtsc	size	

Directorio de archivos M en C:\matlab\signal

addwopi	bartlett	bilinear	blakman	boxcar	buttap	butter
chebap	chebwin	cheby	cov	decimate	denf	detrend
eqnerr2	fftdemo	filtdemo	fir1	fir2	frecs	freqz
fstab	hamming	hanning	interp	kaiser	numf	readme2
remez	specplot	spectrum	triang	xcorr	xcorr2	yulewalk

Directorio de archivos M en C:\matlab\control

abcdchk	cgivens	dgram	esti	lqr2	obsvf	ss2tf	tzero
acker	connect	dimpulse	esti1	lsim	ord2	ss2zp	zgrid
append	ctrb	dlqe	esti3	lyap	parallel	step	zp2ss
are	ctrbf	dlqr	feedback	margin	place	stepfun	zp2pf
awgmatr	ctrldemo	dlsim	fixphase	minreal	readme3	tf2ss	
balreal	d2c	dlyap	gram	modred	ric	tf2zp	
blkbuil	damp	dmodred	impulse	nargchk	riocus	tio	
bode	dbalreal	dric	lqe	nyquist	shord	tio1	
c2d	dbode	dstep	lqr	obsv	series	tio2	

Directorio de archivos M en C:\matlab\ident

armax	cont	gn	ivar	pe	readme4	selstruc	zpplot
arx	contin	iddemo	ivstruc	pem	resid	sibox	
arxstruc	covf	idsim	ivx	phase	search	spa	
auxvar	covf2	inival	mincrit	polyform	searchax	theta	
bj	freqfunc	iv	mktheta	present	searchbj	trf	
bodeplot	fstap	iv4	oe	pz	searchoe	zp	

ANEXO 4

LISTADO DE LA UNIDAD LABCOM

```
unit labcom;

interface

procedure da_out(y:real;canal:integer);
procedure ad_in(var y:real);
procedure prog_canal(canal:integer;frec:real);
procedure ad_start(canal:integer);

implementation

procedure da_out;
{Rutina que realiza la conversión digital a analógica}
type
   arreglo=record
              h:byte;
              l:byte;
           end;
var
   y2:arreglo;

   procedure dos_compl(y:real;var y2:arreglo);
   {Rutina que obtiene el código en 2's complemento del valor a desplegar}
   begin
      with y2 do
      begin
         if y>=0 then
            Y:=204.7*Y
         else
            y:=204.8*y+4096;
         h:=trunc(y/256);
         l:=trunc(y-256*h);
      end;
   end;

begin
   dos_compl(y,y2);
   port[$711+2*canal]:=y2.h;{Se envían los primeros 4 msb}
   port[$710+2*canal]:=y2.l;{Se envían los restantes 8 lsb}
end;
```

```
procedure prog_canal;
{Rutina que ejecuta la programación del canal de conversión y del
contador}
var
   x:integer;
begin
   x:=port[$710+6];
   port[$710+4]:=128;
   port[$710+9]:=23;
   port[$710+8]:=0;
   port[$710+8]:=128;
   port[$710+9]:=5;
   port[$710+8]:=49;
   port[$710+8]:=trunc(15-ln(frec)/ln(10)+0.5);
   port[$710+8]:=153;
   port[$710+8]:=0;
   port[$710+9]:=112;
   port[$710+4]:=132;
   port[$710+5]:=canal;
end;

procedure ad_in;
{Rutina que hace la conversión analógica a digital; requiere haber
programado previamente al canal y al contador, y se puede usar en ciclos
dentro de un programa}
var
   lo,hi:integer;
begin
   while port[$710+4]<128 do;
   lo:=port[$710+5];
   hi:=port[$710+6];
   y:=256*hi+lo;
   if y>32767 then
      y:=(y-65536)/204.8
   else
      y:=y/204.7;
end;

procedure ad_start;
begin
   port[$710+4]:=128;
   port[$710+5]:=canal;
   port[$710+6]:=0;
   while port[$710+4]<128 do;
end;

end.
```

Esta obra se terminó de imprimir en junio de 1991
en los talleres de Tesis Económicas Profesionales,
Campeche 156, Col. Roma. Tels.: 5643954 y 5848153,
bajo la supervisión en la Sección Editorial de la
Lic. Angélica Monroy López.
El tiraje de esta edición fue de 75 ejemplares.

F-DEPFI/D-95/1991/Ej.6

720894

F-DEPFI
D-95
1991
Ej.6